École Doctorale de Sciences Mathématiques de Paris Centre

Quelques problèmes combinatoires et algorithmiques sur les classes de permutations

THÈSE

présentée et soutenue publiquement le 4 décembre 2009

pour l'obtention du

Doctorat de l'université Paris Diderot – Paris VII

(spécialité informatique)

par

Mathilde Bouvel

Devant le jury composé de

Mme Marilena Barnabei
M. Philippe Duchon
M. Michel Habib
M. Christophe Paul, *rapporteur*
M. Renzo Pinzani, *co-directeur*
M. Simone Rinaldi
M. Dominique Rossin, *directeur*
Mme Michèle Soria

Et après avis de

M. Michael Albert
M. Antonio Machì
M. Christophe Paul

Laboratoire d'Informatique Algorithmique : Fondements et Applications

À mon parrain.

Remerciements

Les remerciements d'une thèse en sont un peu le prologue, mais surtout, et en dépit de leur place dans ce manuscrit, l'épilogue. Et comme toute belle histoire qui se termine, il lui faut un *happy end*. Avant d'aller vers d'autres projets, je voudrais donc remercier les protagonistes de cette histoire de trois années, et ceux des années précédentes, qui m'ont conduite à mener cette thèse à bien.

En premier lieu, c'est avec beaucoup de sincérité que je remercie Dominique, pour m'avoir permis de travailler avec lui pendant ces trois ans. Il me semble qu'il a envisagé ma thèse comme un travail ensemble plutôt que comme un encadrement au sens strict du terme, et je lui en suis très reconnaissante. Malgré ses nombreuses activités en dehors du LIAFA, Dominique a su rester toujours disponible, et nos fréquentes séances de travail, joyeuses et dynamiques, sont parmi les meilleurs souvenirs de ma thèse.

Merci aussi à Renzo d'avoir accepté de co-encadrer ma thèse, et merci à ceux qui l'entourent à Florence et à Sienne : tous m'ont accueillie très chaleureusement, et ont fait de mes quelques séjours de l'autre côté des Alpes des moments agréables et enrichissants, pas seulement du point de vue scientifique.

Je remercie vivement Michael Albert, Toni Machì et Christophe Paul d'avoir accepté la lourde charge de rapporteur de ma thèse. Vos horizons différents m'ont offert trois éclairages complémentaires sur mon travail, ce que j'estime être un grand privilège. À chacun, merci bien sûr pour le temps et l'attention que vous avez accordés à la lecture de mon manuscrit, mais surtout merci pour les ouvertures que vous m'avez proposées : je peux à présent continuer mon chemin et explorer ces pistes, dans un avenir que j'espère proche.

Merci également à celles et ceux qui ont accepté avec enthousiasme de prendre part au jury de soutenance : Marilena Barnabei, Philippe Duchon, Michel Habib, Christophe Paul, Simone Rinaldi et Michèle Soria.

D'autres aussi ont accepté de relire certaines parties de ma thèse, alors en cours de rédaction : Vincent, Marie, Lucas, Anne, Adeline, Enrica, Philippe et Clémence. Je leur adresse de vifs remerciements, tant pour leurs remarques qui ont amélioré le manuscrit que pour m'avoir donné confiance par leurs encouragements. Un merci spécial à Vincent pour sa relecture de grande précision et ses commentaires constructifs, à Philippe pour ses notes de cours et son bibtex (et pour la confiture de rose), et à Enrica pour m'avoir accompagnée dans un thème (ou une version, tout dépend du point de vue) plus difficile que prévu.

Je remercie aussi ceux avec qui j'ai travaillé depuis trois ou quatre ans : mes co-auteurs, ceux (bien plus nombreux) avec qui j'ai échangé des idées au détour d'un couloir ou autour d'un café, et les enseignants-chercheurs avec qui j'ai travaillé pendant le monitorat. Plus généralement, merci aux membres du LIAFA et de l'UFR d'informatique de Paris 7 de faire de notre environnement de travail un lieu agréable et convivial, où l'on aimerait seulement pouvoir pousser les murs.

Merci pour leur solidarité aux stagiaires, aux thésards, et aux récemment docteurs que j'ai côtoyés au LIAFA, à PPS, au DSI de Florence, ou ailleurs, et dont un bon nombre sont devenus des amis. Je remercie en particulier Maddalena, ainsi que sa famille et ses amis, qui m'ont adoptée à chacune de mes visites à Florence. Un merci particulier aussi à Carine, Marie, Vincent et Christophe, à présent tous docteurs, qui m'ont aidée dans les aspects pratiques de la rédaction de thèse et de l'année marathon qui suit.

Toujours dans les aspects pratiques, je remercie Noëlle Delgado et Michèle Wasse, dont l'efficacité à résoudre tous les problèmes d'ordre administratif n'est plus à démontrer, et dont j'ai maintes fois profité des conseils avisés.

Je souhaite remercier les organisateurs successifs des journées ALÉA au CIRM, et les nombreux et joyeux combinatoriciens et probabilistes que j'ai pu y rencontrer. Merci aussi aux membres du projet ANR GAMMA pour nos rendez-vous moins réguliers mais plus fréquents, et tout aussi réjouissants.

Merci aux gâteaux du vendredi et aux croissants du séminaire thésard, et donc bien sûr aussi aux organisateurs, aux cuisiniers, et aux orateurs qui font la réussite de ces rendez-vous du LIAFA et de PPS.

Bien que présentant une thèse d'informatique, je remercie très naturellement mes professeurs de mathématiques, de la sixième aux classes prépa : dans des styles très différents, ils m'ont appris à aimer les maths et à les respecter, sans trop se prendre au sérieux, ce que je continue à faire, du moins me semble-t-il.

Merci ensuite aux enseignants-chercheurs, à l'ENS de Cachan puis au MPRI, qui m'ont communiqué leur passion et m'ont fait bifurquer vers l'informatique théorique. Merci en particulier à Hubert Comon pour son engagement et son investissement dans le département informatique.

Merci enfin à mes professeurs d'italien, qui m'ont appris la langue mais aussi la culture italienne, et sans qui je n'aurais certainement pas envisagé de la même manière la cotutelle que j'ai souhaitée et dont j'ai profité sur bien des plans.

Je remercie les filles de la synchro, pour m'avoir fait garder les pieds sur terre quand la thèse me plongeait la tête dans les nuages : c'est bien le comble pour des Mouettes qui passent leur temps à la piscine !

Merci à mes proches amis qui ont un "vrai" travail, pour tous les bons moments partagés.

J'adresse un merci très ému à ma famille, dont j'ai ressenti la fierté de me voir mener ce projet à bien, et dont le soutien m'a portée pendant les derniers mois, et malgré la douleur partagée.

Merci à Clémence pour les voyages qu'on a faits ensemble, et à Simon pour les dimanches tranquilles dans mon petit appartement. Merci à mes parents qui, sans toujours bien comprendre mes études, en ont toujours été curieux, et fiers aussi. J'ajoute des remerciements d'ensemble à mes parents, mon frère et ma sœur : avec vous, et que l'on soit à Laxou, à Paris ou ailleurs, je trouve des moments de sérénité comme en aucune autre circonstance. Dans le tourbillon d'activités où je me suis embarquée, j'ai mesuré dernièrement à quel point ces moments sont précieux.

And last but not least, je remercie Richard, pour avoir su s'adapter au rythme qui n'en est pas un des derniers moments de rédaction et des semaines qui ont suivi, et pour m'accompagner et me soutenir tant au quotidien que dans les moments de doute inévitables.

Résumé

Ce travail étudie les classes de permutations, avec les deux points de vue combinatoire et algorithmique. Il y est fréquemment fait usage de la décomposition par substitution des permutations, à travers leurs arbres de décomposition. Ces arbres sont le pendant combinatoire des arbres des intervalles communs des permutations utilisés en algorithmique.

D'abord, on s'intéresse à la recherche de motifs dans les permutations : occurrence d'un motif dans une permutation, et recherche de plus grand motif commun à deux (ou à un ensemble de) permutations. Ces problèmes sont NP-difficiles, et en introduisant des restrictions à la classe des permutations séparables, on y apporte dans ce cadre restreint des solutions polynomiales, lorsqu'elles existent.

Ensuite, on analyse combinatoirement certains modèles pour l'évolution des génomes. Dans le modèle de duplication en tandem avec perte aléatoire, on caractérise comme classe de permutations à motifs exclus les permutations obtenues après p étapes d'évolution. L'autre modèle étudié est celui du tri parfait par renversements, où on donne une analyse de complexité fine d'un algorithme préexistant pour le calcul de scénario d'évolution optimal.

Enfin, on étudie la classe des permutations en épingles, qui joue un rôle clé dans un processus permettant d'obtenir de manière systématique les séries génératrices de certaines classes de permutations. On donne une caractérisation des arbres de décomposition des permutations en épingles, la série génératrice rationnelle de cette classe, et un algorithme polynomial pour tester si une classe de permutations contient un nombre fini de permutations en épingles propres.

Abstract

This work is dedicated to the study of permutation classes, with both an algorithmic and a combinatorial point of view. It uses in most of its developments the substitution decomposition of permutations, represented by their decomposition trees. These trees are the combinatorial counterpart of the common interval trees of permutations used in algorithmics.

We first deal with pattern matching problems in permutations : finding an occurrence of a pattern in a permutation, and finding a longest common pattern between two (or among a set of) permutations. These problems are NP-hard, and by imposing some restrictions to the class of separable permutations, we describe in this restricted framework polynomial solutions, when they exist.

Then, we offer a combinatorial analysis of some models of genome evolution. In the tandem duplication-random loss model, we characterize as a pattern-avoiding permutation class the permutations obtained after p steps of evolution. We also study the model of perfect sorting by reversals : in this case we give a precise complexity analysis of a previously known algorithm computing optimal evolution scenarios.

Finally, we study the class of pin-permutations, which plays a key role in a procedure allowing the automatic computation of the generating functions of some permutation classes. We give a characterization of the decomposition trees of pin-permutations, the rational generating function of the class, and a polynomial algorithm to test whether a permutation class contains a finite number of proper pin-permutations.

Riassunto

Breve riassunto

Questo lavoro studia le classi di permutazioni, dal punto di vista combinatorico ed algoritmico. Sarà spesso usata la decomposizione per sostituzione delle permutazioni, utilizzando gli alberi di decomposizione. Questi alberi sono l'equivalente combinatorico degli alberi degli intervalli comuni delle permutazioni usati in algoritmica.

Per cominciare ci si interessa alla ricerca di motivi nelle permutazioni : l'occorrenza d'un motivo in una permutazione e la ricerca d'un più lungo motivo comune tra due (o tra un insieme di) permutazioni. Questi problemi sono NP-difficili, ma concentrandoci sulla classe delle permutazioni separabili, si danno delle soluzioni polinomiali, quando esistono.

In seguito, si propone un'analisi combinatorica di alcuni modelli per l'evoluzione dei genomi. Nel modello di duplicazione tandem con perdita aleatoria, si caratterizzano come classi di permutazioni a motivo escluso le permutazioni ottenute dopo p tappe. L'altro modello studiato è quello dell'ordinamento perfetto per ribaltamenti, per il quale si analizza con esattezza la complessità di un algoritmo preesistente per il calcolo di scenari ottimali.

Infine, si studia la classe delle permutazioni a spilli, che svolge un ruolo centrale nel calcolo sistematico delle funzioni generatrici di alcune classi di permutazioni. Si forniscono una caratterizzazione degli alberi di decomposizione delle permutazioni a spilli, la funzione generatrice razionale di questa classe, e un algoritmo polinomiale per testare se una classe di permutazioni contiene un numero finito di permutazioni a spilli proprie.

Risultati in dettaglio

Gli oggetti studiati in questa tesi sono le permutazioni, considerate principalmente dal punto di vista dei *motivi*. In questo contesto si parla di permutazioni d'interi da 1 a n, essendo n la lunghezza della permutazione. Un motivo in una permutazione σ è in un certo senso una "sotto-permutazione" : è una sotto-sequenza rinormalizzata della permutazione σ. Per esempio, $\sigma = 6\,2\,4\,7\,3\,1\,8\,5$ è una permutazione di lunghezza 8, di cui un motivo è la permutazione $\tau = 3\,2\,1\,4$, ottenuta rinormalizzando la sotto-sequenza $6\,4\,1\,8$ di σ.

Il concetto di motivo nelle permutazioni generalizza ovviamente quello di motivo in un testo, aggiungendogli un'ulteriore dimensione. Questo giustifica il fatto di considerare alcuni problemi algoritmici : in particolare, i problemi di occorrenza di un motivo in una permutazione e di ricerca di motivi comuni a più permutazioni. Al di là dell'interesse di questi problemi, la loro risoluzione ha anche delle conseguenze importanti in combinatoria, com'è spiegato in [BRV08] e nel capitolo 9.

Dal punto di vista combinatorico, il concetto di motivo è usato principalmente per definire le classi di permutazioni : sono degli insiemi chiusi verso il basso rispetto alla relazione di motivo (che è un ordine parziale). Vale a dire che se una permutazione σ appartiene ad una classe $\mathcal{C}$, allora tutti i motivi estratti da σ appartengono anche loro a $\mathcal{C}$. Ogni classe di permutazioni può essere caratterizzata da un insieme di motivi esclusi, chiamato la base di $\mathcal{C}$. Una conoscenza precisa delle proprietà combinatoriche di classi di permutazioni permette di definire particolari sotto-problemi che possono essere risolti da algoritmi più veloci rispetto al problema generale. Questo punto viene

affrontato nei lavori sulla ricerca di motivi nelle permutazioni. Le proprietà combinatoriche di una classe possono anche permettere di analizzare in modo più preciso la complessità di alcuni algoritmi sulle permutazioni (vedere [BBCP07] e il capitolo 7).

La maggior parte dei lavori presentati nella tesi usa la *decomposizione per sostituzione* [MR84] delle permutazioni, rappresentata dai loro *alberi di decomposizione*. Questa struttura è stata definita indipendentemente in algoritmica, dove è chiamata *albero degli intervalli comuni* delle permutazioni [BXHP05, per esempio], e più recentemente in combinatoria [AA05, sulla decomposizione per sostituzione delle permutazioni].

Il problema algoritmico del calcolo dell'albero di decomposizione è legato a quello del calcolo dell'albero di decomposizione modulare dei grafi. È stato motivato dal problema del calcolo di scenari evolutivi, ed è stato risolto con successo [BXHP05, BCMR05, BCMR08]. Fra le sue applicazioni, quest'albero è stato usato in particolare per lo sviluppo di un algoritmo per il calcolo di scenari evolutivi nel modello dell'ordinamento perfetto per ribaltamenti in [BBCP07]. Si è fatta un'analisi più precisa della complessità di quest'algoritmo, studiando proprietà degli alberi di decomposizione. Si sono anche proposti altri usi algoritmici di questi alberi, in particolare per la ricerca di motivi nelle permutazioni [BBL98, Iba97, e i capitoli 4 e 5].

In combinatoria, la decomposizione per sostituzione è stata introdotta di recente, in [AA05], e quindi i risultati che ne fanno uso sono pochi, per ora. La decomposizione per sostituzione permette di avere un punto di vista generale sulle classi di permutazioni, al contrario della maggior parte dei lavori precedenti che si concentrano su una classe o una famiglia di classi. Ad esempio, usando questo strumento sono stati sviluppati metodi algoritmici per automatizzare, sotto certi ipotesi, il calcolo della funzione generatrice di alcune classi di permutazioni [AA05, BRV08, e il capitolo 9].

Nello studio di ogni famiglia di oggetti discreti, e in particolare per le permutazioni, combinatoria e algoritmica propongono due punti di vista complementari su questi oggetti : i risultati combinatorici possono avere ripercussioni algoritmiche, e vice versa alcuni algoritmi hanno conseguenze dal punto di vista combinatorico.

Questa tesi presenta diversi esempi di interazione tra combinatoria e algoritmica.

La tesi è organizzata in quattro parti, abbastanza indipendenti.

La prima parte contiene sia un'introduzione sulle classi di permutazioni e sulla decomposizione per sostituzione, sia una presentazione di alcuni risultati generali ottenuti sugli alberi di decomposizione.

Il capitolo 1 tratta quasi esclusivamente della combinatoria delle classi di permutazioni. Si definiscono in modo formale le permutazioni tramite diverse rappresentazioni (lineare, grafica, ...), il concetto di motivo, le classi di permutazioni, e le loro basi di motivi esclusi. Se ne danno poi alcune proprietà molto note. Si presentano in seguito delle problematiche possibili esaminate in letteratura, sullo studio delle classi di permutazioni. Dal punto di vista combinatorico, i problemi d'enumerazione sono stati quelli più studiati in passato, in particolare con l'interesse sviluppato per la congettura di Stanley-Wilf, dimostrata in [MT04]. Più recentemente, è stato considerato il problema di descrivere una classe di permutazioni senza utilizzare la base dei motivi esclusi [Atk99, AS02, AAA$^+$07, Bri07]. Dal punto di vista algoritmico, sono esaminati i problemi di decisione dell'appartenenza di una permutazione ad una classe, e di sviluppo di algoritmi efficienti [AAAH01]. Questo capitolo si conclude con una presentazione più precisa (benché decisamente non esauriente) sui problemi d'enumerazione associati alle classi di permutazioni.

Il capitolo 2 è dedicato alla decomposizione per sostituzione delle permutazioni, e ai loro alberi di decomposizione. Questo capitolo inizia con la descrizione dei principi generali di decomposizione per sostituzione di oggetti discreti, applicabili non solo nell'ambito delle permutazioni [MR84]. Si presenta in particolare la decomposizione per sostituzione nel caso dei grafi, ricordando le basi della decomposizione modulare [Hab, per esempio], per poi dare le uguaglianze e le differenze tra decomposizione modulare dei grafi e decomposizione per sostituzione delle permutazioni. Dopo questa

introduzione generale, si definisce la decomposizione per sostituzione delle permutazioni [AA05], nella quale interviene una famiglia di permutazioni con un ruolo essenziale : le permutazioni *semplici* [Brib, per una presentazione recente delle conoscenze sull'argomento]. Si definisce poi l'albero degli intervalli comuni delle permutazioni, e se ne danno alcune proprietà, relative in particolare al suo calcolo da un algoritmo di complessità lineare [BXHP05, BCMR05, BCMR08]. Una delle difficoltà principali nel capitolo 2 è stata d'unificare i concetti e i risultati ottenuti nelle comunità algoritmica e in quella combinatorica. Si dimostra in particolare che gli alberi di decomposizione e gli alberi degli intervalli comuni rappresentano la stessa struttura : questo fatto è ammesso da chiunque sia interessato a questi oggetti, ma non ne esiste in letteratura nessuna dimostrazione. Si spiega anche come si può adattare l'algoritmo lineare per calcolare l'albero degli intervalli comuni al fine di aggiungere sulla sua struttura le etichette che lo trasformeranno in albero di decomposizione. Tornando a risultati precedenti a questa tesi, sono ricordate le definizioni del prodotto di sostituzione e delle classi chiuse dal prodotto di sostituzione [AS02], che costituiscono un caso particolare più semplice per quasi tutti i problemi sulle classi di permutazioni risolvibili con l'uso degli alberi di decomposizione. Infine, un altro risultato ottenuto è presentato alla fine di questa prima parte, e descrive la forma "media" (o più precisamente la forma asintotica quasi certa) degli alberi di decomposizione.

La seconda parte porta sullo studio di problemi algoritmici sulla ricerca di motivi nelle permutazioni.

Il capitolo 3 è un'introduzione al problema di ricerca di un'occorrenza di un motivo in una permutazione. Vi è spiegato il lato NP-completo del problema [BBL98], e vi sono descritti casi particolari studiati in letteratura [BBL98, Iba97] che hanno soluzione polinomiale. Ci si interessa ad un caso particolare di questo problema, la ricerca di un'occorrenza di un motivo appartenente alla classe delle permutazioni separabili [Iba97]. Si definisce dunque la classe delle permutazioni separabili, e la struttura d'albero di separazione associata, ovviamente legata agli alberi di decomposizione. Questi alberi di separazione hanno un ruolo fondamentale nei due algoritmi di [BBL98, Iba97], che usano tecniche di programmazione dinamica per risolvere in tempo polinomiale il problema di ricerca di un'occorrenza di un motivo separabile in una permutazione. Inoltre, questi algoritmi costituiscono una base per lo sviluppo degli algoritmi nei due capitoli seguenti.

Il capitolo 4 porta sul problema di ricerca d'un più lungo motivo comune a due permutazioni. Questo problema è NP-difficile, poiché il problema della ricerca dell'occorrenza d'un motivo è NP-completo. Si presenta un primo caso polinomiale studiato in [MR06], per la ricerca d'un più lungo motivo comune a due permutazioni ordinabili da una pila. L'algoritmo proposto utilizza una biezione tra le permutazioni ordinabili da una pila e gli alberi, e su un algoritmo di ricerca d'un più grande sotto-albero comune a due alberi [ZS89]. Si descrive poi un primo risultato ottenuto, un algoritmo polinomiale per la ricerca d'un più lungo motivo comune a due permutazioni, di cui una è separabile. Essendo le permutazioni ordinabili da una pila separabili, questo risultato generalizza il precedente. In questo caso, gli strumenti per lo sviluppo dell'algoritmo sono diversi : qui, si utilizzano gli alberi di separazione e la programmazione dinamica, usando le idee descritte nel capitolo 3. Infine, si mostra come quest'algoritmo può essere adattato per calcolare un più lungo motivo comune a due permutazioni qualsiasi, facendo uso degli alberi di decomposizione sviluppati al posto degli alberi di separazione. Anche se l'algoritmo non è più polinomiale, la complessità ottenuta non è esponenziale rispetto alla lunghezza delle permutazioni, ma solo rispetto ad un parametro più piccolo : il numero massimo di figli di un nodo primo nei loro alberi di decomposizione. Questi lavori sono iniziati durante lo stage di Master a Parigi [Bou06], e sono stati pubblicati in [BR06].

Il capitolo 5 generalizza la ricerca d'un più lungo motivo comune a due permutazioni studiando la ricerca d'un più lungo motivo comune ad un insieme di permutazioni. Anche qui si ha un problema NP-difficile e ci si restringe ad un sotto-problema, imponendo delle condizioni sul motivo cercato piuttosto che sulle permutazioni dati nel problema. Si studia la ricerca d'un più lungo motivo comune separabile ad un insieme di permutazioni, cioè d'un motivo separabile che abbia

un'occorrenza in ognuna delle permutazioni dell'insieme, e che sia di lunghezza massimale possibile. Si dà un algoritmo polinomiale per la ricerca d'un più lungo motivo comune separabile ad un insieme di k permutazioni (con k fissato), e si dimostra che quando la cardinalità dell'insieme di permutazioni dato non è fissata in anticipo, il problema è NP-difficile. A questo punto, è naturale chiedersi se il più lungo motivo comune separabile ad un insieme di permutazioni è un'approssimazione precisa del loro motivo comune di lunghezza massimale (senza condizione di separabilità). Si dimostra che imponendo che il motivo appartenga ad una classe di permutazioni (come le permutazioni separabili), non si può ottenere un'approssimazione più precisa di $\sqrt{Opt}$, Opt essendo la lunghezza massimale d'un motivo comune senza restrizione. Questi risultati sono stati pubblicati in [BRV07].

La terza parte analizza due modelli per l'evoluzione dei genomi, che sono rappresentati da permutazioni (segnate o no) : il modello di duplicazione tandem con perdita aleatoria e il modello dell'ordinamento perfetto per ribaltamenti. L'approccio è principalmente combinatorico, ma anche un po' algoritmico.

Il capitolo 6 considera il modello di duplicazione tandem con perdita aleatoria [CCMR06]. Vi è descritta una caratterizzazione delle permutazioni ottenute in al più p tappe a partire da una permutazione identità in questo modello, in termini di motivi esclusi. Nel caso particolare del modello di duplicazione completa con perdita aleatoria, si descrive completamente la base di motivi esclusi che caratterizza la classe delle permutazioni ottenute in al più p tappe. Si studiano anche i motivi esclusi di questa base, per darne una caratterizzazione locale e ottenere dei risultati d'enumerazione parziali. Nel modello generale, si dimostra che la base della classe delle permutazioni ottenute in al più p tappe è finita, e si dà un limite superiore sulla lunghezza dei motivi che contiene. Si propone anche un algoritmo per il calcolo di scenari evolutivi in questo modello, e si confronta il costo dei scenari calcolati in questo modo con il costo di uno scenario ottimale. I lavori presentati nel capitolo 6 sono oggetto di due articoli [BR09] e [BP08].

Il capitolo 7 tratta di un altro modello per l'evoluzione dei genomi : quello dell'ordinamento perfetto per ribaltamenti [FV04], che lavora con permutazioni segnate. Vi si analizza la complessità di un algoritmo per il calcolo di scenari ottimali estratto da [BBCP07] e che lavora sugli alberi di decomposizione. Si tratta di un algoritmo la cui complessità è parametrizzata, e esponenziale nel caso peggiore. Si dimostra che in pratica quest'algoritmo è polinomiale nel caso medio. In seguito si analizzano alcuni parametri dei scenari ottimali calcolati nel caso delle permutazioni separabili (o commutanti), caso particolarmente pertinente dal punto di vista biologico. Si danno stime asintotiche del numero medio di ribaltamenti in un scenario ottimale, e della lunghezza media di un ribaltamento in un tale scenario : i risultati di quest'analisi sono coerenti con gli esperimenti dei biologi, aggiungendo un punto alla pertinenza del modello. La maggior parte dei risultati del capitolo 7 sono pubblicati in [BCMR], una versione lunga di quest'articolo è stata sottomessa.

La quarta parte esplora metodi automatici per il calcolo delle funzioni generatrici delle classi di permutazioni, o almeno per determinare delle proprietà di queste funzioni generatrici [AA05, BRV08]. Alla base della ricerca delle proprietà delle funzioni generatrici, c'è la ricerca di struttura nelle classi di permutazioni, che sarà presentata in dettaglio nell'introduzione della quarta parte. Questi problemi sono considerati dal punto di vista delle permutazioni a spilli. Questa classe di permutazioni è stata definita in [BRV08] in una procedura di decisione che testa una condizione sufficiente affinché una classe di permutazioni data dalla sua base finita abbia una funzione generatrice algebrica.

Il capitolo 8 è dedicato ad uno studio dettagliato della classe delle permutazioni a spilli. Essendo queste permutazioni descritte solo da una proprietà grafica, se ne dà una caratterizzazione più pratica tanto in combinatoria quanto in algoritmica : si caratterizzano gli alberi di decomposizione che corrispondono alle permutazioni a spilli. La dimostrazione di questo risultato è abbastanza tecnica, e poggia su un'analisi precisa delle rappresentazioni a spilli delle permutazioni. Una prima

conseguenza di questa caratterizzazione delle permutazioni a spilli è che permette il calcolo della funzione generatrice associata a questa classe. Una congettura di [BRV08] è che questa funzione generatrice sia razionale, poiché le permutazioni di questa classe sono codificate da un linguaggio razionale di parole, però senza unicità della codifica. Il risultato ottenuto è una funzione generatrice razionale, confermando questa congettura. Si conclude il capitolo 8 con un'analisi della base della classe delle permutazioni a spilli, e questa volta si dimostra la non validità della congettura che la base di questa classe sia finita. I risultati del capitolo 8 sono presentati in [BBR09].

Il capitolo 9 riprende la procedura di [BRV08] già accennata, che testa una condizione sufficiente per l'algebricità della funzione generatrice di una classe di permutazioni. Dato che avere un numero finito di permutazioni semplici è una condizione sufficiente affinché una classe di permutazioni abbia una funzione generatrice algebrica, la procedura proposta permette di decidere se il numero di permutazioni semplici in una classe $\mathcal{C}$ è finito, supponendo che la classe $\mathcal{C}$ sia data dalla sua base finita di motivi esclusi. Ci si impegna a trasformare questa procedura in un vero algoritmo, di complessità polinomiale. Il principio fondamentale non è cambiato : costruire un automa che riconosce le parole di spilli strette delle permutazioni a spilli appartenenti a $\mathcal{C}$, e testare se il linguaggio riconosciuto da quest'automa è finito. Ma ci sono diversi punti che devono essere analizzati con cura per cercare di mantenerne il carattere polinomiale. Un primo punto ovvio, utilizzando la caratterizzazione del capitolo 8, consiste nel proporre un algoritmo che testa se una permutazione è a spilli oppure no. Le due difficoltà principali sono di descrivere con precisione l'insieme delle parole di spilli associate ad una permutazione a spilli σ, poi di costruire in tempo polinomiale un automa *deterministico* che riconosce le parole di spilli strette associate alle permutazioni a spilli proprie di cui σ è motivo. Per risolvere il primo problema, si usa la caratterizzazione del capitolo 8, e uno studio, ancora una volta abbastanza tecnico, permette di capire come sono formate tutte le parole di spilli associate ad una permutazione a spilli. Per il secondo problema, si costruiscono gli automi cercati da una ricorrenza, usando l'astuzia della lettura da destra a sinistra per ottenere degli automi deterministici. Combinando questi diversi punti, si ottiene un algoritmo che testa in tempo polinomiale se una classe data dalla sua base finita di motivi esclusi contiene un numero finito di permutazioni semplici. L'articolo [BBPR] in preparazione conterrà i risultati del capitolo 9.

L'ultimo capitolo (capitolo 10) in allegato riassume i concetti di base (definizioni e alcune proprietà usate nella tesi) delle funzioni generatrici.

Table des matières

Introduction **1**

I Objets et outils **7**

1 Classes de permutations **9**
1.1 Permutations et leurs différents modes de représentation 10
1.1.1 Définition 10
1.1.2 Représentation sur deux lignes 10
1.1.3 Représentation linéaire 11
1.1.4 Décomposition en produit de cycles 12
1.1.5 Représentation graphique 13
1.1.6 Autres structures représentant des permutations 14
1.2 Motifs dans les permutations et classes de permutations 15
1.2.1 Notion de motif dans les permutations 15
1.2.2 Les classes de permutations et leurs bases 16
1.3 Trois opérations sur les permutations, qui conservent les motifs 18
1.4 Quelques problématiques combinatoires et algorithmiques de l'étude des classes de permutations 20
1.4.1 Problèmes d'énumération et conjecture de Stanley-Wilf 20
1.4.2 Décider l'appartenance à une classe 21
1.4.3 Conception d'algorithmes efficaces 22
1.4.4 Décrire une classe sans connaitre sa base 23
1.5 Étude combinatoire des classes de permutations définies par leur base 24
1.5.1 Motifs exclus de petite taille 25
1.5.2 Généralisations de la notion de motifs exclus 25
1.5.3 Méthodes générales d'énumération 27

2 Décomposition par substitution et arbres de décomposition des permutations **31**
2.1 Décomposition par substitution et décomposition modulaire des graphes 32
2.1.1 Principes généraux de décomposition par substitution 32
2.1.2 Spécialisation sur les graphes : la décomposition modulaire 33
2.2 Décomposition par substitution des permutations 38
2.2.1 Permutations simples 38
2.2.2 Dilatation et théorèmes de substitution 41
2.2.3 Arbres de décomposition 44
2.3 Point de vue algorithmique : l'arbre des intervalles communs 47
2.3.1 Définition et premières propriétés 47
2.3.2 Lien étroit avec les arbres de décomposition 49

2.3.3 Algorithmes de calcul en temps linéaire 52
2.4 Produit de substitution et classes fermées par produit de substitution 56
2.4.1 Définitions et propriétés 56
2.4.2 Exemples d'utilisation des classes fermées par produit de substitution . . 58
2.4.3 Produit de substitution et classes de base finie 59
2.5 Forme en moyenne des arbres de décomposition 59

II Recherche de motifs dans les permutations 63

Généralités sur la recherche de sous-structure dans les objets structurés 65

3 État de l'art : recherche de motifs dans les permutations 67
3.1 Le problème de recherche de motif dans les permutations 68
3.1.1 Présentation du problème 68
3.1.2 Caractère NP-complet du problème général 68
3.1.3 Exemple : recherche de sous-séquence croissante maximale 69
3.1.4 Méthodologie pour des algorithmes efficaces ; application à la recherche de motifs de taille 4 70
3.2 Permutations séparables, arbres de séparation, et généralisations 71
3.2.1 La classe des permutations séparables et leurs arbres de séparation . . . 71
3.2.2 Arbres de séparation, arbres de Schröder, et arbres de décomposition . . 73
3.3 Algorithme de recherche d'un motif séparable dans une permutation 76
3.3.1 Algorithme de Bose, Buss et Lubiw 76
3.3.2 Algorithme d'Ibarra 78

4 Recherche de plus grand motif commun à deux permutations 81
4.1 Le problème de recherche de plus grand motif commun à deux permutations . . 82
4.1.1 Définition du problème, et premières remarques de complexité 82
4.1.2 Une généralisation : recherche de plus grand motif commun entre une permutation et une classe 83
4.1.3 Restriction à des permutations triables par pile 84
4.2 Recherche du plus grand motif commun à deux permutations, dont une séparable 86
4.2.1 Description de l'algorithme 86
4.2.2 Correction et analyse de complexité 88
4.3 Extension à deux permutations quelconques grâce aux arbres de décomposition . 90
4.3.1 Description de l'algorithme 91
4.3.2 Correction et analyse de complexité 92

5 Recherche de plus grand motif commun à un ensemble de permutations 95
5.1 Algorithme pour la recherche d'un plus grand motif séparable commun à K permutations 96
5.1.1 Description de l'algorithme 96
5.1.2 Correction et analyse de complexité 99
5.2 Complexité de la recherche de plus grand motif séparable commun à un ensemble de permutations 101
5.3 Approximation par motif commun restreint à une classe 104

III Analyse combinatoire de modèles pour l'évolution des génomes 107

Introduction aux problèmes de distance dans les modèles pour l'évolution des génomes 109

6 Duplication en tandem avec perte aléatoire 113
6.1 Présentation du modèle "duplication en tandem avec perte aléatoire" 114
6.1.1 Définition du modèle et quelques résultats antérieurs 114
6.1.2 Analyse combinatoire du modèle restreint de "duplication complète avec perte aléatoire" 116
6.2 Permutations minimales avec d descentes 118
6.2.1 Caractérisation 119
6.2.2 Énumeration pour les tailles extrémales 121
6.3 Analyse combinatoire du modèle général de "duplication en tandem avec perte aléatoire" 136
6.3.1 Permutations obtenues en une étape de largeur au plus K 136
6.3.2 Permutations obtenues en p étapes de largeur au plus K 137
6.4 Algorithmes de calcul de scénarios, et étude de complexité 142
6.4.1 Étude algorithmique du modèle de duplication complète avec perte aléatoire 142
6.4.2 Borne supérieure dans le modèle général 143
6.4.3 Borne inférieure dans le modèle général 148

7 Tri par renversement de permutations signées 151
7.1 Généralités sur le tri par renversements des permutations 152
7.2 Tri parfait par renversements des permutations signées 154
7.2.1 Algorithme de tri parfait par renversements 154
7.2.2 Analyse de complexité en moyenne 155
7.3 Le cas particulier des permutations séparables 158
7.3.1 Nombre moyen de renversements 158
7.3.2 Taille moyenne d'un renversement 161

IV Étude de la nature des séries génératrices des classes de permutations 165

Automatisation du calcul des séries génératrices dans des classes de permutations "structurées" 167

8 Arbres de décomposition des permutations en épingles, et série génératrice de la classe 171
8.1 Définitions et propriétés autour des permutations en épingles 172
8.1.1 Représentations en épingles et permutations en épingles 172
8.1.2 Représentations en épingles et mots d'épingles 175
8.1.3 Familles des épis et des quasi-épis. 176
8.2 Caractérisation des arbres de décomposition des permutations en épingles 180
8.2.1 Remarques préliminaires 180
8.2.2 Propriétés des nœuds linéaires 181
8.2.3 Propriétés des nœuds premiers 183
8.2.4 Preuve de la condition nécessaire 185
8.2.5 Preuve de la condition suffisante 188
8.3 Série génératrice des permutations en épingles simples 189

8.3.1 Propriétés des premiers points d'une représentation en épingles d'une permutation en épingles simple . . . 189
8.3.2 Énumération des représentations en épingles simples . . . 195
8.3.3 Énumération des permutations en épingles simples . . . 198
8.4 Série génératrice de la classe des permutations en épingles . . . 199
8.4.1 Une équation définissant les arbres de décomposition des permutations en épingles . . . 199
8.4.2 Séries génératrices simples mises en jeu . . . 200
8.4.3 Série génératrice de la classe . . . 201
8.5 Discussion sur la base de la classe des permutations en épingles . . . 202

9 Algorithme polynomial décidant si une classe contient un nombre fini de permutations simples 207
9.1 Mots d'épingles et première procédure de décision . . . 208
9.1.1 Quelques propriétés des mots d'épingles . . . 208
9.1.2 Les étapes de la procédure de décision de Brignall, Ruškuc et Vatter . . . 211
9.2 Quels mots d'épingles pour quelles permutations en épingles . . . 214
9.2.1 Permutations en épingles de base . . . 215
9.2.2 Permutations en épingles de racine linéaire construites récursivement . . . 220
9.2.3 Permutations en épingles de racine première construites récursivement . . . 227
9.3 Facteurs de mots d'épingles pour identifier les motifs dans les permutations . . . 232
9.4 Algorithme polynomial testant si une classe contient un nombre fini de simples . 236
9.4.1 Les étapes de l'algorithme . . . 236
9.4.2 Construction de l'automate pour les cas de base . . . 238
9.4.3 Construction de l'automate pour les cas récursifs de racine linéaire . . . 240
9.4.4 Construction de l'automate pour les cas récursifs de racine première . . . 241
9.4.5 Récapitulatif de complexité . . . 243
9.4.6 Marquage des états . . . 243
9.4.7 Assemblage de l'automate . . . 244
9.4.8 Cas particulier des classes fermées par produit de substitution . . . 246

Conclusion et perspectives 247

Annexe 251

10 Séries génératrices ordinaires : un outil pour la combinatoire énumérative 253
10.1 Définition et principes fondamentaux . . . 254
10.2 Classes décomposables et méthode symbolique . . . 255
10.3 Analyse de singularité et évaluation asymptotique . . . 258
10.4 Séries bivariées et espérance de paramètres . . . 259

Table des figures 263

Index 267

Bibliographie 269

Introduction

Les travaux présentés dans cette thèse se situent au confluent de deux domaines de l'informatique théorique, que sont d'une part la *combinatoire* et d'autre part l'*algorithmique*.

La combinatoire est une branche des mathématiques discrètes qui étudie des collections d'objets discrets, c'est-à-dire pour lesquels on dispose d'une notion de taille telle que le nombre d'objets de taille n soit fini pour tout n. Certains des buts de la combinatoire sont de caractériser les objets étudiés, d'en décrire des propriétés, de compter pour tout n le nombre d'objets de taille n, de les engendrer exhaustivement ou aléatoirement.

En algorithmique, on cherche plutôt à donner des algorithmes pour résoudre efficacement des problèmes manipulant ces objets combinatoires, problèmes qui sont souvent modélisation d'enjeux réels dans d'autres disciplines scientifiques. Le plus souvent, la complexité d'un algorithme est mesurée relativement à la taille de l'objet en entrée du problème, sauf dans certains cas où il est plus adapté de l'évaluer plutôt par rapport à la taille du résultat calculé.

Les objets formels étudiés en combinatoire et en algorithmique sont toujours modèles d'objets réels, même si la modélisation transparaît de façon plus ou moins immédiate suivant les problèmes étudiés. Les motivations pour l'étude de ces objets proviennent de l'informatique elle-même (modélisation de structures de données, analyse des réseaux, ...), mais aussi de la biologie – en particulier moléculaire et évolutive [Den], de la physique – dont la physique statistique [Bax82], mais pas uniquement [CW07, par exemple], de la chimie [BGK, et ses références, parmi d'autres], ou de la sociologie – à travers l'étude des réseaux sociaux [PR09], parmi d'autres. Lorsque les résultats théoriques obtenus le permettent, ces domaines constituent naturellement un champ d'application privilégié des résultats obtenus. Dans cette thèse, on verra que les motivations les plus immédiates proviennent de la bio-informatique, pour l'étude des modèles d'évolution des génomes.

Dans l'étude d'une famille d'objets discrets, combinatoire et algorithmique offrent deux points de vue sur ces objets, qui sont complémentaires : les résultats combinatoires obtenus peuvent avoir des retombées algorithmiques, et certains algorithmes ont des implications du point de vue combinatoire.

En effet, les propriétés combinatoires des objets, et en particulier les propriétés mettant en évidence une structure dans ces objets, peuvent être mises à profit pour concevoir des algorithmes qui, tirant parti de ces propriétés, sont d'une meilleure complexité qu'un algorithme ne disposant pas de cette connaissance combinatoire *a priori*. Par exemple, les propriétés de la classe des *cographes* autorisent la description d'algorithmes résolvant des problèmes pour cette classe de graphes plus rapidement que sur des graphes quelconques [CPS85, BLS99].

À l'inverse, si la combinatoire permet d'affirmer l'existence de "sur-couches" sur les objets qui sont le reflet de leur structure intrinsèque, elle ne permet pas d'utiliser cette structure à des fins algorithmiques. Il faut pour cela se poser d'abord la question du calcul d'un témoin de cette structure : un algorithme efficace le calculant est un préalable nécessaire à la résolution de tout problème par son utilisation. Un exemple emblématique de telle "sur-couche" est l'arbre de décomposition modulaire des graphes : sa connaissance rend plus aisée la résolution de nombreux problèmes sur

les graphes, et la question du calcul en temps linéaire de l'arbre de décomposition modulaire d'un graphe a reçu et reçoit toujours beaucoup d'attention [Pau, Hab, HP, et leurs références].

On verra dans cette thèse plusieurs exemples d'interaction fructueuse entre combinatoire et algorithmique.

Les objets d'étude privilégiés dans cette thèse sont les *permutations*, envisagées principalement avec le point de vue des *motifs* dans les permutations. Dans ce cadre, on traite des permutations des entiers entre 1 et n, n étant alors la taille de la permutation. Un motif dans une permutation σ est en quelque sorte une "sous-permutation" : c'est une sous-séquence renormalisée extraite de la permutation σ. Par exemple, $\sigma = 6\,2\,4\,7\,3\,1\,8\,5$ est une permutation de taille 8, et un motif en est la permutation $\tau = 3\,2\,1\,4$, obtenue en renormalisant la sous-séquence $6\,4\,1\,8$ extraite de σ.

La notion de motif dans les permutations généralise évidemment celle de motif dans un texte, en lui ajoutant une dimension supplémentaire, et à ce titre certaines questions algorithmiques se doivent d'être posées. Il faut en particulier s'interroger sur les problèmes d'occurrence d'un motif dans une permutation et de recherche de motif commun à plusieurs permutations. En dehors de leur intérêt manifeste, la résolution de problèmes algorithmiques de cette nature sur les permutations peut avoir des retombées en combinatoire, comme on le verra dans [BRV08] et au chapitre 9.

Du point de vue combinatoire, la notion de motif sert principalement à la définition des *classes* de permutations : ce sont des ensembles fermés par le bas pour la relation de motif (qui est une relation d'ordre partiel), c'est-à-dire que si une permutation σ appartient à une classe $\mathcal{C}$, alors tous les motifs extraits de σ appartiennent aussi à $\mathcal{C}$. Une bonne connaissance des propriétés combinatoires des classes de permutations permet de décrire des restrictions pertinentes de problèmes sur les permutations, qui peuvent être résolues par des algorithmes plus rapides que le problème général. Ce point est illustré par [MR06] et la partie II de cette thèse pour les problèmes de recherche de motif dans les permutations. Les propriétés combinatoires d'une classe peuvent aussi permettre d'analyser plus finement la complexité de certains algorithmes sur les permutations (voir [BBCP07, BCP08] et le chapitre 7).

L'essentiel des travaux présentés dans cette thèse utilise l'outil de la *décomposition par substitution* [MR84] des permutations, capturée par leurs *arbres de décomposition*. Cette structure sur les permutations a été définie indépendamment en algorithmique, où elle porte le nom d'*arbre des intervalles communs* [BXHP05, par exemple], et plus récemment en combinatoire [AA05, sur la décomposition par substitution des permutations].

La question algorithmique du calcul de l'arbre de décomposition est liée à celle du calcul de l'arbre de décomposition modulaire des graphes. Elle a été introduite avec des motivations de calcul de scénarios pour l'évolution des génomes, et a été résolue avec succès [BXHP05, BCMR05, BCMR08]. On présentera un algorithme de calcul de scénario d'évolution à partir de l'arbre de décomposition extrait de [BBCP07], et on verra aussi d'autres utilisations algorithmiques de ces arbres, en particulier pour la recherche de motifs dans les permutations [BBL98, Iba97, et les chapitres 4 et 5].

Du point de vue combinatoire, l'utilisation de la décomposition par substitution a émergé très récemment dans [AA05], si bien que peu de résultats l'utilisant sont connus à ce jour. La décomposition par substitution permet d'avoir un point de vue général sur les classes de permutations, au contraire de la plupart des travaux antérieurs qui se concentrent sur une classe ou une famille de classes donnée. C'est par exemple en utilisant cet outil que sont développées des méthodes algorithmiques pour automatiser, sous certaines hypothèses, le calcul de la série génératrice des classes de permutations [AA05, BRV08, et le chapitre 9].

Plan détaillé et contributions Cette thèse est organisée en quatre parties relativement indépendantes, chaque partie à l'exception de la première étant précédée d'une introduction.

La partie I constitue essentiellement un état de l'art sur les classes de permutations et sur les arbres de décomposition. Il faut cependant noter que quelques résultats obtenus dans mon travail de thèse, parce qu'ils traitent des arbres de décomposition de manière assez générale, sont intégrés à cette partie.

Le chapitre 1 traite presque exclusivement de la combinatoire des classes de permutations. On y définit formellement les permutations, à travers leurs différentes représentations (linéaire, graphique, ...), la notion de motif dans les permutations, les classes de permutations, et leurs bases de motifs exclus. On donne aussi quelques propriétés très classiques de ces objets. On présente ensuite plusieurs problématiques envisageables, et envisagées dans la littérature, pour l'étude des classes de permutations. Combinatoirement, les problèmes d'énumération ont été les plus étudiés par le passé, en particulier à cause de l'intérêt pour la conjecture de Stanley-Wilf, aujourd'hui démontrée [MT04]. Plus récemment, on s'est posé la question de la description d'une classe de permutations autrement que par sa base de motifs interdits [Atk99, AS02, AAA$^+$07, Bri07]. D'un point de vue algorithmique, on envisagera les problèmes de décision de l'appartenance d'une permutation à une classe, et de conception d'algorithmes efficaces [AAAH01]. Ce chapitre s'achève par un état de l'art plus précis (bien que résolument non exhaustif) sur les problèmes d'énumération associés aux classes de permutations.

Le chapitre 2 est consacré à la décomposition par substitution des permutations, et à leurs arbres de décomposition. Ce chapitre s'ouvre sur des principes généraux de décomposition par substitution d'objets discrets, applicables dans d'autres contextes que celui des permutations [MR84]. On présente en particulier la spécialisation de la décomposition par substitution au cas des graphes, en rappelant les bases de la décomposition modulaire [Hab, par exemple], pour nous permettre de souligner dans la suite les points communs et (surtout) les différences entre décomposition modulaire des graphes et décomposition par substitution des permutations. Après cette introduction générale, on définit la décomposition par substitution des permutations [AA05], qui fait intervenir une famille de permutations y jouant un rôle fondamental : les permutations *simples* [Brib, pour un survol récent sur ce thème]. On définit ensuite l'arbre des intervalles communs des permutations, et on en donne quelques propriétés, en particulier relatives à son calcul par un algorithme en temps linéaire [BXHP05, BCMR05, BCMR08]. Une des difficultés principales dans le chapitre 2 a été d'unifier les concepts et résultats obtenus dans les communautés d'algorithmique des graphes et de combinatoire. On donnera en particulier une preuve que les arbres de décomposition et les arbres des intervalles communs représentent la même structure, ce qui est admis par quiconque s'intéresse à ces objets, mais dont aucune preuve à ma connaissance ne figure dans la littérature. On expliquera aussi comment adapter l'algorithme linéaire de calcul de l'arbre des intervalles communs pour ajouter sur sa structure les étiquettes qui vont le transformer en arbre de décomposition. Revenant à des résultats antérieurs à cette thèse, on définit le produit de substitution des permutations, et les classes fermées par produit de substitution [AS02], qui constituent un cas particulier plus simple pour presque toutes les questions sur les classes de permutations qui peuvent être résolues par l'utilisation des arbres de décomposition. Enfin, un autre résultat nouveau figure à la fin du chapitre 2, et décrit la forme "en moyenne" (ou plus précisément, la forme asymptotique presque sûre) des arbres de décomposition.

La partie II s'intéresse à l'étude de problèmes algorithmiques sur la recherche de motifs dans les permutations.

Le chapitre 3 présente un état de l'art sur le problème de la recherche d'une occurrence d'un motif dans une permutation. On y explique le caractère NP-complet du problème [BBL98], et on en décrit des cas particuliers étudiés dans la littérature [BBL98, Iba97], et qui admettent une solution polynomiale. La restriction du problème général qui va nous intéresser tout particulièrement est la

recherche d'une occurrence d'un motif appartenant à la classe des permutations séparables [Iba97]. On définit donc la classe des permutations séparables, et la structure d'arbre de séparation qui leur est associée et qui n'est pas sans lien avec les arbres de décomposition. Ces arbres de séparation jouent un rôle clé dans les deux algorithmes de [BBL98, Iba97], qui, grâce à des techniques de programmation dynamique, résolvent en temps polynomial le problème de recherche d'occurrence d'un motif séparable dans une permutation. Ces algorithmes constituent en outre une base de travail pour la conception des algorithmes dans les chapitres suivants.

Le chapitre 4 considère le problème de recherche de plus grand motif commun à deux permutations. Ce problème est NP-difficile, de par la NP-complétude de la recherche d'occurrence de motif. On présente un premier cas polynomial étudié dans [MR06], pour la recherche de plus grand motif commun à deux permutations triables par pile. L'algorithme proposé repose sur une bijection entre les permutations triables par pile et les arbres, et sur un algorithme de recherche de plus grand sous-arbre commun à deux arbres [ZS89]. On décrira ensuite une première contribution qui consiste en un algorithme polynomial pour la recherche de plus grand motif commun entre deux permutations, dont une est séparable. Les permutations triables par pile étant séparables, ceci généralise le résultat précédent. La conception de l'algorithme repose cependant sur des outils différents, à savoir les arbres de séparation et la programmation dynamique, en s'appuyant sur les idées décrites dans le chapitre 3. Enfin, on verra que cet algorithme peut être adapté pour calculer un plus grand motif commun à deux permutations quelconques, en utilisant des arbres de décomposition développés à la place des arbres de séparation. Même si on perd bien sûr le caractère polynomial de l'algorithme, la complexité obtenue n'est pas exponentielle en la taille des permutations en entrée, mais seulement en un paramètre plus petit : l'arité maximale d'un nœud premier dans leur arbre de décomposition. J'ai commencé ces travaux pendant mon stage de Master [Bou06], et ils ont fait l'objet de la publication [BR06].

Le chapitre 5 généralise la recherche de plus grand motif commun à deux permutations, et étudie la recherche de plus grand motif commun à un ensemble de permutations. Là encore, cette question étant NP-difficile, on se restreint à un sous-problème, mais en imposant cette fois des conditions sur le motif cherché plutôt que sur les permutations en entrée. On s'intéresse à la recherche de plus grand motif séparable commun à un ensemble de permutations, c'est-à-dire d'un motif séparable ayant une occurrence dans chacune des permutations de l'ensemble, et qui soit le plus long pour ce critère. On décrira un algorithme polynomial pour la recherche de plus grand motif séparable commun à un ensemble de k permutations (k étant fixé), et on verra que quand le cardinal de l'ensemble de permutations en entrée n'est pas connu à l'avance, le problème est NP-difficile. Il est naturel alors de se demander si le plus grand motif séparable commun à un ensemble de permutations est une bonne approximation de leur plus grand motif commun (sans contrainte de séparabilité). On verra qu'en contraignant le motif à appartenir à une classe de permutations (comme les séparables, entre autres), on ne peut pas faire mieux qu'obtenir une approximation en $\sqrt{Opt}$, Opt désignant la taille du plus grand motif commun non contraint. Ces résultats ont été publiés dans [BRV07].

La partie III analyse deux modèles pour l'évolution des génomes, où ceux-ci sont représentés par des permutations (signées ou non) : le modèle de duplication en tandem avec perte aléatoire, et le modèle du tri parfait par renversements. Le point de vue adopté est essentiellement combinatoire, mais l'algorithmique trouve aussi sa place dans cette partie.

Le chapitre 6 envisage le modèle de duplication en tandem avec perte aléatoire [CCMR06]. On décrit une caractérisation des permutations obtenues en au plus p étapes à partir d'une permutation identité dans ce modèle, en termes de motifs exclus. Dans le cas particulier du modèle de duplication complète avec perte aléatoire, on décrit complètement la base de motifs exclus qui caractérise la classe des permutations obtenues en au plus p étapes. On étudie aussi les motifs exclus qui apparaissent dans cette base, pour en donner une caractérisation locale, et obtenir des résultats d'énumération (même s'ils restent partiels). Dans le modèle général, on montre que la base de la classe des permutations obtenues en au plus p étapes est finie, et on donne une borne sur la taille

des motifs qu'elle contient. On propose aussi un algorithme de calcul de scénario d'évolution dans ce modèle, et on compare le coût des scénarios ainsi calculés au coût d'un scénario optimal. Les travaux présentés au chapitre 6 ont fait l'objet des deux articles [BR09] et [BP08].

Le chapitre 7 s'intéresse à un autre modèle pour l'évolution des génomes : celui du tri parfait par renversements [FV04], qui travaille sur des permutations signées. On y analyse la complexité d'un algorithme de calcul de scénario optimal tiré de [BBCP07] et travaillant sur les arbres de décomposition. Il s'agit d'un algorithme de complexité paramétrée, qui est exponentiel dans le pire des cas. On démontre qu'il est en fait polynomial en moyenne. On analyse ensuite certains paramètres des scénarios optimaux calculés, pour le cadre restreint des permutations séparables (ou commutantes), particulièrement pertinent d'un point de vue biologique. On donne des estimations asymptotiques du nombre moyen de renversements dans un scénario optimal, et de la taille moyenne d'un renversement dans un tel scénario : les résultats de cette analyse sont cohérents avec les observations biologiques, ce qui vient conforter la pertinence de ce modèle. Une grande partie des résultats du chapitre 7 sont publiés dans [BCMR], dont une version étendue a été soumise.

La partie IV explore des méthodes automatiques pour le calcul des séries génératrices des classes de permutations, ou tout au moins pour donner des propriétés de ces séries génératrices [AA05, BRV08]. Sous-jacente à cette recherche de propriétés des séries génératrices, on voit poindre l'idée de recherche de structure dans les classes de permutations, qui sera présentée plus en détails comme introduction à la partie IV. Ces questions seront abordées sous l'angle des permutations en épingles. Cette classe de permutations a été introduite dans [BRV08] dans une procédure de décision testant une condition suffisante pour qu'une classe de permutations donnée par sa base finie ait une série génératrice algébrique.

Le chapitre 8 est consacré à une étude approfondie de la classe des permutations en épingles. Ces permutations n'étant décrites que par une propriété graphique, on en donne une caractérisation plus utilisable aussi bien combinatoirement qu'algorithmiquement : on caractérise les arbres de décomposition qui correspondent aux permutations en épingles. La preuve de ce résultat est plutôt technique, et repose sur une analyse fine des représentations en épingles des permutations. Une première conséquence de cette caractérisation des permutations en épingles est qu'elle permet le calcul de la série génératrice associée à cette classe. Il était conjecturé par [BRV08] que la série génératrice des permutations en épingles était rationnelle, puisque ces permutations peuvent être codées par un langage rationnel de mots, mais sans qu'il y ait unicité du codage. Le résultat obtenu est une série génératrice rationnelle, répondant ainsi par l'affirmative à cette conjecture. On termine le chapitre 8 en s'intéressant à la base de la classe des permutations en épingles, cette fois-ci en infirmant la conjecture selon laquelle la base de cette classe est finie. Les résultats du chapitre 8 sont présentés dans [BBR09].

Le chapitre 9 reprend la procédure de [BRV08] mentionnée plus haut, testant une condition suffisante sur l'algébricité de la série génératrice d'une classe de permutations. Sachant que contenir un nombre fini de permutations simples est une condition suffisante pour qu'une classe de permutations ait une série génératrice algébrique, la procédure proposée permet de décider si le nombre de permutations simples dans une classe $\mathcal{C}$ est fini, la classe $\mathcal{C}$ étant supposée donnée par sa base finie de motifs exclus. On s'attache à transformer cette procédure en un véritable algorithme, de complexité polynomiale. Le principe de fonctionnement reste fondamentalement inchangé : construire un automate qui accepte les mots d'épingles stricts des permutations en épingles propres appartenant à $\mathcal{C}$, et tester si le langage accepté par cet automate est fini. Mais il y a plusieurs points qu'il faut analyser avec attention pour éviter une explosion combinatoire, et conserver le caractère polynomial. Un premier point, évident avec la caractérisation du chapitre 8, consiste à proposer un algorithme testant si une permutation donnée est ou non une permutation en épingles. Les deux principales difficultés sont de savoir décrire précisément l'ensemble des mots d'épingles associés à une permutation en épingles σ, puis de construire en temps polynomial un automate *déterministe* qui accepte les mots d'épingles stricts associés aux permutations en épingles propres

dont σ est motif. Pour résoudre la première, on utilise la caractérisation du chapitre 8, et une étude là encore assez technique permet de comprendre comment sont formés tous les mots d'épingles associés à une permutation en épingles. Pour la seconde, on construira les automates cherchés par récurrence, en utilisant l'astuce d'une lecture de droite à gauche des mots d'épingles pour obtenir des automates qui soient déterministes. En mettant bout à bout ces différents points, on obtient un algorithme testant en temps polynomial si une classe donnée par sa base finie de motifs exclus contient un nombre fini de permutations simples. L'article [BBPR] en préparation reprendra le contenu du chapitre 9.

Dans le chapitre 10 en annexe, on résume, sous la forme d'un bref tutoriel, le formalisme des séries génératrices et les quelques résultats les concernant qui sont utilisés dans cette thèse.

Première partie

Objets et outils

« On numérote
On énumère
Beaucoup trop
Pour nos petits cerveaux »

Bénabar, *Les Numéros*

Chapitre 1

Classes de permutations

Sommaire

1.1 Permutations et leurs différents modes de représentation 10
1.1.1 Définition 10
1.1.2 Représentation sur deux lignes 10
1.1.3 Représentation linéaire 11
1.1.4 Décomposition en produit de cycles 12
1.1.5 Représentation graphique 13
1.1.6 Autres structures représentant des permutations 14
1.2 Motifs dans les permutations et classes de permutations 15
1.2.1 Notion de motif dans les permutations 15
1.2.2 Les classes de permutations et leurs bases 16
1.3 Trois opérations sur les permutations, qui conservent les motifs . . 18
1.4 Quelques problématiques combinatoires et algorithmiques de l'étude des classes de permutations 20
1.4.1 Problèmes d'énumération et conjecture de Stanley-Wilf 20
1.4.2 Décider l'appartenance à une classe 21
1.4.3 Conception d'algorithmes efficaces 22
1.4.4 Décrire une classe sans connaitre sa base 23
1.5 Étude combinatoire des classes de permutations définies par leur base 24
1.5.1 Motifs exclus de petite taille 25
1.5.2 Généralisations de la notion de motifs exclus 25
1.5.3 Méthodes générales d'énumération 27

Les travaux exposés dans cette thèse portent sur les *classes de permutations*. On présentera ces classes dès le paragraphe 1.2, mais il nous faut évidemment commencer par définir les objets de base sur lesquels portent cette étude, à savoir les *permutations*.

1.1 Permutations et leurs différents modes de représentation

1.1.1 Définition

La désormais célèbre encyclopédie libre WIKIPÉDIA ouvre son article consacré aux permutations par la définition suivante :

> *En mathématiques, la notion de permutation exprime l'idée de réarrangement d'objets discernables. Une permutation de n objets distincts rangés dans un certain ordre, correspond à un changement de l'ordre de succession de ces n objets.*

Cette définition est immédiatement suivie par un ancrage des permutations dans le contexte où elles sont étudiées :

> *La permutation est une des notions fondamentales en combinatoire, c'est-à-dire pour des problèmes de dénombrement et de probabilités discrètes.*

Dans le PETIT LAROUSSE 2009, une définition plus formelle a été choisie :

> [MATH.] *Bijection d'un ensemble sur lui-même. (Le nombre de permutations d'un ensemble de m objets est m! [factorielle m].)*

Dans cette définition, pourtant très concise, le choix est fait de donner le *nombre* des permutations d'un ensemble de cardinal m. Cela suggère que les principaux problèmes concernant les permutations s'attachent à des questions d'énumération, et plus généralement de combinatoire.

Dans cette thèse, on s'intéresse aux permutations d'un intervalle d'entiers, commençant le plus souvent en 1.

Définition 1.1. Une *permutation* d'un intervalle I de $\mathbb{N}$ est une bijection de I sur lui-même. La *taille* d'une permutation désigne le nombre d'éléments de I. La taille d'une permutation σ est notée $|\sigma|$.

On note $\mathcal{S}_n$ l'ensemble des permutations de l'intervalle $[1..n]$, et $\mathcal{S} = \cup_{n\geq 1}\mathcal{S}_n$.

Sauf mention ponctuelle du contraire, on considère toujours des permutations de $\mathcal{S}$. Ainsi, lorsque l'on considère des *permutations de taille n*, sans plus de précisions, il s'agit de permutations appartenant à $\mathcal{S}_n$.

Enfin, on utilisera la notation Id_n (ou simplement Id lorsque la valeur de n est claire dans le contexte) pour désigner la permutation identité de taille n : pour tout $i \in [1..n], Id_n(i) = i$.

Il existe de nombreuses manières de représenter les permutations. Cette thèse utilise principalement deux d'entre elles : la représentation linéaire et la représentation graphique des permutations. On les présente ci-après, de même que quelques autres, en illustrant l'intérêt de ces différentes représentations par des exemples.

1.1.2 Représentation sur deux lignes

La représentation des permutations la plus fréquemment rencontrée (et ce dès les cours de mathématiques du secondaire) est celle que l'on appelle la *représentation sur deux lignes*. Pour une permutation de taille n, elle fait figurer sur une première ligne les entiers entre 1 et n en ordre croissant, et en-dessous leurs images.

Exemple 1.2. La permutation σ de taille 9 qui à 1 associe 8, à 2 associe 3, à 3, 1, à 4, 9, ... est notée

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 8 & 3 & 1 & 9 & 2 & 6 & 4 & 5 & 7 \end{pmatrix}$$

dans sa représentation en deux lignes.

Une variante de cette représentation consiste à ajouter des segments reliant l'occurrence de tout entier sur la première ligne avec son occurrence sur la deuxième ligne. Le résultat obtenu sur l'exemple précédent est donné par la figure 1.1.

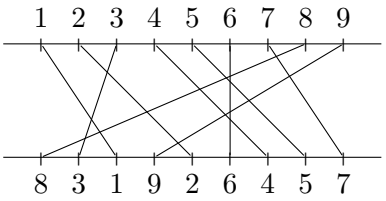


FIG. 1.1 – Représentation sur deux lignes d'une permutation.

Cette représentation est celle qui permet le mieux de visualiser les *inversions* dans une permutation σ, c'est-à-dire les paires d'entiers (i, j) telles que $i < j$ et $\sigma(i) > \sigma(j)$: les inversions correspondent à deux segments qui s'intersectent. C'est avec cette représentation qu'il est le plus aisé de définir le *graphe des inversions* d'une permutation, appelé plus simplement graphe de permutation (voir page 70).

1.1.3 Représentation linéaire

La représentation sur deux lignes d'une permutation (de taille n) a invariablement pour première ligne $1\ 2 \dots n$. En conservant seulement sa deuxième ligne, on obtient donc une autre manière de décrire complètement une permutation, que l'on appellera *représentation linéaire*.

Définition 1.3. La *représentation linéaire* d'une permutation σ de taille n est le mot de n lettres $\sigma(1)\ \sigma(2) \dots \sigma(n)$.

On notera souvent σ_i l'image $\sigma(i)$ de i par σ. L'entier σ_i sera désigné par le terme *élément* (ou parfois *lettre*) de la permutation σ. L'indice i est appelé *position* de l'élément σ_i, et pour éviter les confusions, on dira que σ_i est l'élément de *valeur* σ_i dans σ. Enfin, l'élément à la position i pourra être appelé plus simplement i-ème élément d'une permutation.

Exemple 1.4. La représentation linéaire de la permutation donnée à l'exemple 1.2 est $\sigma = 8\,3\,1\,9\,2\,6\,4\,5\,7$. L'élément à la position 5 a pour valeur 2.

Les mots qui représentent des permutations sont très particuliers : les mots de taille n qui conviennent sont ceux sur l'alphabet $[1..n]$ qui contiennent exactement une fois chaque lettre. Malgré cette particularité, la représentation linéaire permet de définir des concepts sur les permutations empruntés à la théorie des langages, et d'envisager des problèmes directement hérités de l'algorithmique du texte. La notion de *motif* dans les permutations transporte dans ce contexte celle de *sous-mot*, au sens classique du terme. En particulier, cela autorise à étudier, pour les permutations, les problèmes de recherche de motif (ou *pattern matching*), largement étudiés dans le cadre des mots. La partie II de cette thèse est consacrée aux problèmes de recherche de motifs dans les permutations.

La représentation linéaire des permutations étant la plus compacte, c'est aussi celle que l'on utilisera par défaut dans la suite.

1.1.4 Décomposition en produit de cycles

Cette représentation des permutations ne sera pas utilisée dans le travail rapporté ici, mais elle est extrêmement importante dans d'autres contextes, ce qui justifie de la présenter brièvement malgré tout.

Les permutations d'un ensemble X forment un groupe, au sens algébrique du terme, avec pour opération de produit la composition de fonctions. Ce groupe est appelé le *groupe symétrique* sur l'ensemble X. Dans le groupe symétrique (sans perte de généralité, sur $X = [1..n]$), on démontre que les permutations peuvent être décomposées de manière unique en produit de *cycles*, comme énoncé dans le théorème 1.6.

Définition 1.5. Pour tout k-uplet $i_1, \ldots, i_k$ d'entiers distincts compris entre 1 et n, le *cycle* $(i_1\ i_2 \ldots i_k)$ est la permutation σ de $\mathcal{S}_n$ définie par $\sigma(i_p) = i_{p+1}$ si $p \in [1..(k-1)]$, $\sigma(i_k) = i_1$, et $\sigma(j) = j$ si $j \notin \{i_1, \ldots, i_k\}$. Le *support* du cycle est l'ensemble $\{i_1, \ldots, i_k\}$.

On remarque immédiatement que deux cycles de supports disjoints commutent (pour l'opération de composition), ce qui autorise à énoncer le théorème suivant :

Théorème 1.6. *Toute permutation de $\mathcal{S}_n$ se décompose de manière unique (à l'ordre des facteurs près) en produit de cycles à supports disjoints.*

La preuve de ce théorème est un grand classique des premières années d'études supérieures en mathématiques, et est omise ici.

La représentation des permutations en produit de cycles est très utilisée en combinatoire dans le cadre de l'étude des *cartes*. Une carte à n arêtes peut être codée par la donnée de deux permutations α et σ de taille $2n$, α étant une involution sans point fixe (c'est-à-dire que α a n cycles, tous de taille 2). Chaque élément dans $[1..2n]$ représente une demi-arête (ou *brin*), les cycles de α indiquent les appariements entre deux brins d'une même arête, et les cycles de σ correspondent aux sommets de la carte, en indiquant l'ordre circulaire dans lequel les brins se répartissent autour de chaque sommet. Sur cette représentation des cartes, définie dans [Edm60], on pourra consulter les travaux fondateurs [CM92] pour plus de détails, [Sch98, chapitre 1] pour une présentation concise, ou [Mie] pour une approche didactique. Une très vaste littérature, dont ces ouvrages rendent compte, témoigne de l'importance du codage des cartes combinatoires par deux permutations.

Mais la représentation en produit de cycles des permutations n'est pas utilisée seulement pour l'étude des cartes. Un exemple d'utilisation de cette décomposition en cycles est la *transformation de Foata*, qui permet entre autre de démontrer que le nombre de minima partiels et le nombre de cycles suivent la même distribution.

Définition 1.7. Un *minimum partiel* dans une permutation $\sigma \in \mathcal{S}_n$ est un élément σ_i tel que pour tout $j < i$, $\sigma_j > \sigma_i$.

Par exemple, la permutation $\sigma = \mathbf{9}\,\mathbf{3}\,7\,6\,\mathbf{2}\,4\,5\,\mathbf{1}\,8$ a quatre minima partiels, à savoir les éléments 9, 3, 2, et 1, indiqués en gras.

La transformation de Foata est une bijection de $\mathcal{S}_n$, qui fonctionne comme suit. Partant d'une permutation σ, on écrit sa décomposition en cycles, en faisant commencer chaque cycle par son plus petit élément, et en ordonnant les cycles par ordre décroissant de leurs plus petits éléments. En effaçant les parenthèses, on obtient une permutation de $\mathcal{S}_n$ dont le nombre de minima partiels est égal au nombre de cycles de σ.

Par exemple, la permutation $\sigma = 8\,4\,7\,5\,2\,3\,6\,1\,9$ se décompose en produit de cycles en $\sigma = (9)(3\,7\,6)(2\,4\,5)(1\,8)$, avec des cycles commençant par leurs plus petits éléments, en ordre décroissant. L'image de σ par la bijection de Foata est la permutation $9\,3\,7\,6\,2\,4\,5\,1\,8$ qui a quatre minima partiels.

1.1.5 Représentation graphique

Dans cette thèse, et en particulier dans la partie IV, les permutations seront souvent représentées de manière graphique, par ce qui est parfois appelé leur *diagramme*. Ici, on utilisera plutôt le terme de *représentation graphique*.

Dans cette représentation, une permutation σ de taille n est donnée par une grille carrée de côté n, où se répartissent n points (représentant les éléments de la permutation), de sorte à avoir exactement un point par ligne et par colonne. Le point dans la i-ème colonne en partant de la gauche est dans la $\sigma(i)$-ème ligne en partant du bas, et représente le i-ème élément de σ. En d'autres termes, les points de la représentation graphique de σ ont pour coordonnées $(i, \sigma(i))$ pour $i \in [1..n]$.

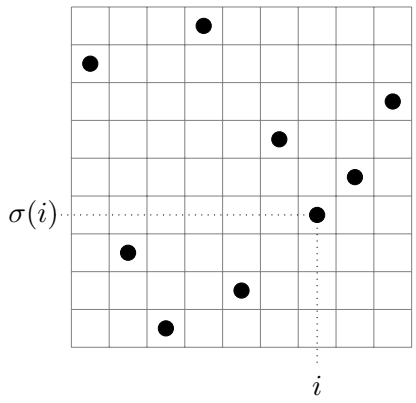


FIG. 1.2 – Représentation graphique de la permutation $\sigma = 8\,3\,1\,9\,2\,6\,4\,5\,7$.

Il a souvent, et depuis longtemps, été fait usage de cette représentation des permutations, en particulier par X. G. Viennot. Cependant, elle apparaît en tant que telle essentiellement dans des articles récents, ce qui s'explique par le développement des outils graphiques. Parmi les problématiques rendues plus abordables par la représentation graphique des permutations, ou pour lesquelles cette représentation permet d'obtenir des résultats complémentaires de ceux obtenus en utilisant leur représentation linéaire, on peut citer, sans définir formellement les objets mis en jeu :

- l'estimation du taux de croissance des classes de permutations [Atk], et en particulier la preuve la conjecture de Stanley-Wilf [MT04],
- l'étude des classes de permutations à croissance polynomiale [AAB07, BMMR],
- la définition et l'étude de polyominos particuliers, décrits par une paire de permutations, et appelés *permutominos* [BDPR07, BLRS08, FFG$^+$06],
- l'étude des permutations triables par plusieurs piles, pour laquelle des avancées ont été obtenues, même s'il reste de nombreuses questions ouvertes dans ce domaine [CKS08],
- la définition de classes de permutations autrement que par leur base, en l'occurrence par des propriétés graphiques (par exemple, deux familles de classes appelées *merged classes* et *grid classes* [Bria, HV06], ou la classe des permutations en épingles [BBR09, BHV08a, BRV08]),
- la construction d'antichaînes infinies de permutations [Bria],
- l'étude des bases (finies ou infinies) des classes de permutations [AA05, BHV08b, Bri07],
- l'étude des permutations simples et des classes fermées par produit de substitution [Atk, AA05],
- la description de conditions suffisantes pour qu'une classe de permutations ait une série génératrice algébrique, et de méthodes pour calculer ces séries [Atk, AA05, BHV08b].

On reviendra dans la partie IV sur les cinq derniers points ci-dessus, à travers l'étude de la classe des permutations en épingles, pour donner un algorithme polynomial testant une condition suffisante d'algébricité.

1.1.6 Autres structures représentant des permutations

Les représentations sur deux lignes, linéaire ou graphique des permutations décrites ci-dessus, même si elles ont chacune leur spécificité, ont en commun de conserver la permutation comme unique objet d'étude : elles ne greffent pas de structure supplémentaire sur ces objets. La représentation en cycles permet au contraire de souligner une structure sous-jacente dans la permutation. C'est cette structure qui est exploitée pour le codage des cartes combinatoires.

Mais la structure cyclique n'est pas la seule que l'on peut mettre en évidence dans une permutation. Il existe en fait d'autres manières de représenter les permutations qui mettent en évidence certaines structures implicites dans ces objets. Cette thèse utilisera deux représentations de ce type.

Comme on l'a déjà mentionné au paragraphe 1.1.2, on peut associer à toute permutation son *graphe d'inversions*, aussi appelé plus simplement *graphe de permutation*. Cette structure sera définie page 70, et elle sera utilisée ponctuellement dans ce manuscrit, le plus souvent pour mettre en parallèle des problèmes sur les graphes avec leur pendant dans le contexte des permutations.

Il faut remarquer qu'il est impropre de parler de *représentation* d'une permutation par son graphe de permutation, car plusieurs permutations peuvent correspondre au même graphe. Pour corriger le caractère non injectif de la construction du graphe de permutation, on peut choisir d'étiqueter chaque sommet du graphe par la valeur de l'élément de la permutation qu'il code. La difficulté est alors reportée sur la détermination des étiquetages admissibles pour tout graphe de permutation.

La représentation structurée des permutations qui est au cœur du travail présenté ici est la structure d'*arbre de décomposition* des permutations. Elle est utilisée dans chaque chapitre, à l'exception du présent chapitre et du chapitre 6.

Les arbres de décomposition seront définis en détail au chapitre 2. Cependant, comme cette thèse met en jeu plusieurs familles d'arbres, dans différents paragraphes, on rappelle dès à présent la définition d'un arbre, formulée de manière récursive.

Définition 1.8. Un *arbre* $\mathcal{T}$ est soit réduit à une *feuille*, soit la donnée d'une *racine* r et de $k \geq 1$ arbres $\mathcal{T}_1, \ldots, \mathcal{T}_k$. Si $\mathcal{T}$ est réduit à une feuille, celle-ci est aussi appelée racine de $\mathcal{T}$.

Les *nœuds* de $\mathcal{T}$ sont sa racine et les nœuds des arbres $\mathcal{T}_1, \ldots, \mathcal{T}_k$ s'ils existent. Un nœud de $\mathcal{T}$, s'il n'est pas une feuille, est appelé *nœud interne* de $\mathcal{T}$.

Les racines r_i des arbres $\mathcal{T}_i$ sont appelés les *fils* de r, et r est le *père* de chaque r_i. Les *descendants* d'un nœud v de $\mathcal{T}$ sont obtenus en itérant la relation fils à partir de v.

Le *sous-arbre* de $\mathcal{T}$ enraciné en r_i est $\mathcal{T}_i$, et récursivement, pour tout nœud v de $\mathcal{T}$, le sous-arbre enraciné en v est la restriction de $\mathcal{T}$ aux descendants de v.

Le nombre k de fils d'un nœud v de l'arbre $\mathcal{T}$ est appelé l'*arité* de v. Une feuille est donc un nœud d'arité 0.

On distinguera les arbres *plans* des arbres *non ordonnés* : un arbre $\mathcal{T}$ est plan si, pour tout nœud interne de $\mathcal{T}$, ses fils sont totalement ordonnés (de gauche à droite) ; au contraire, un arbre est non ordonné si pour tout nœud interne v, il n'y a pas d'ordre entre les fils de v.

Sauf mention du contraire, on considère des arbres finis, c'est-à-dire ayant un nombre fini de nœuds.

La structure d'arbre de décomposition pour représenter les permutations a été introduite indépendamment dans un contexte algorithmique (sous le nom d'*arbre des intervalles communs*), et dans le cadre de l'étude combinatoire des permutations (à travers la *décomposition par substitution*). Le chapitre 2 est exclusivement consacré à cette représentation des permutations et aux concepts qui s'y rattachent. On tente d'y unifier les connaissances sur ces objets obtenues dans les deux communautés algorithmique et combinatoire.

Comme pour les graphes de permutation, les arbres de décomposition, s'ils ne sont pas étiquetés, ne sont pas une *représentation* des permutations, au sens où plusieurs permutations ont le même

arbre de décomposition. Cependant, il est immédiat de décrire les étiquetages admissibles d'un arbre de décomposition, et la correspondance entre permutations et arbres de décomposition étiquetés est bijective. Cela fait de ces objets une représentation des permutations, qui est en outre utilisable en pratique.

1.2 Motifs dans les permutations et classes de permutations

Plus qu'aux permutations en tant qu'objets, cette thèse s'intéresse à des *classes* de permutations, c'est-à-dire à des ensembles de permutations qui sont fermés par le bas pour la relation de *motif*. On définit ici ces notions, et on donne quelques unes de leurs propriétés essentielles.

1.2.1 Notion de motif dans les permutations

La notion de motif dans les permutations est définie de sorte à transporter dans ce contexte la notion de sous-mot dans les mots.

Définition 1.9. Un *sous-mot* d'un mot $w = w_1 w_2 \dots w_n$ sur un alphabet fini Σ est un mot $v = v_1 v_2 \dots v_k$ sur le même alphabet, tel qu'il existe des entiers $1 \leq i_1 < i_2 < \dots < i_k \leq n$ vérifiant que pour tout $j \in [1..k]$, $w_{i_j} = v_j$.

Dans le cas des mots, toute sous-séquence $w_{i_1} w_{i_2} \dots w_{i_k}$ d'un mot w sur l'alphabet Σ est un mot sur le même alphabet. On souhaite que de la même manière un motif d'une permutation de $\mathcal{S}$ soit aussi une permutation de $\mathcal{S}$. Si on définissait un motif d'une permutation, en copiant la définition de sous-mot, comme une sous-séquence extraite, ça ne serait pas le cas : les entiers de la sous-séquence extraite de la permutation n'aurait aucune raison de former un intervalle du type $[1..k]$. Il faut donc disposer d'une opération qui transforme toute sous-séquence extraite d'une permutation de $\mathcal{S}$ en une permutation de $\mathcal{S}$.

Définition 1.10. Deux séquences de k entiers distincts $\sigma = \sigma_1 \sigma_2 \dots \sigma_k$ et $\tau = \tau_1 \tau_2 \dots \tau_k$ sont *isomorphes en ordre* si pour tout $i, j \in [1..k]$, on a $\sigma_i \leq \sigma_j$ si et seulement si $\tau_i \leq \tau_j$.

Exemple 1.11. Par exemple, les deux séquences 5 3 6 7 4 9 et 4 1 5 7 2 8 sont isomorphes en ordre.

Pour toute séquence d'entiers distincts, parmi toutes celles qui lui sont isomorphes en ordre, on peut en distinguer une particulière, définie de manière unique, et que l'on peut voir comme un représentant d'un ensemble de séquences d'entiers distincts toutes isomorphes en ordre.

Définition 1.12. Pour toute séquence $\sigma = \sigma_1 \sigma_2 \dots \sigma_k$ de k entiers distincts, la *normalisation* de σ est la *permutation* de $\mathcal{S}_k$ qui est isomorphe en ordre à σ. En d'autres termes, dans la normalisation de σ, le i-ème plus petit élément de σ est remplacé par i.

Exemple 1.13. Les deux séquences 5 3 6 7 4 9 et 4 1 5 7 2 8 ont la même normalisation : la permutation 3 1 4 5 2 6 $\in \mathcal{S}_6$.

Cette opération de normalisation transforme donc de manière canonique toute séquence d'entiers distincts en une permutation de même taille. Elle permet de définir les motifs dans les permutations :

Définition 1.14. Une permutation $\sigma \in \mathcal{S}_k$ est *motif* d'une permutation $\tau \in \mathcal{S}_n$, ce que l'on note $\sigma \preceq \tau$, s'il existe des entiers $1 \leq i_1 < i_2 < \dots < i_k \leq n$ tels que la séquence d'entiers distincts $\tau_{i_1} \tau_{i_2} \dots \tau_{i_k}$ est isomorphe en ordre à σ. La séquence $\tau_{i_1} \tau_{i_2} \dots \tau_{i_k}$ est appelée une *occurrence* de σ dans τ.

Lorsque $\sigma \preceq \tau$, on dira aussi que τ *contient* σ. Dans le cas contraire, on dira que τ *évite* σ.

Exemple 1.15. La permutation $\sigma = 3\,1\,2$ est un motif de $\tau = 4\,2\,6\,1\,5\,3$, et les 4 occurrences de σ dans τ sont $4\,2\,3$, $4\,1\,3$, $6\,1\,5$ et $6\,1\,3$. La permutation τ contient donc σ. En revanche, elle évite le motif $4\,2\,3\,1$.

Remarque 1.16. On remarque immédiatement que la relation de motif $\preceq$ est réflexive, transitive et antisymétrique : il s'agit donc d'une relation d'ordre partiel sur $\mathcal{S}$. On utilisera parfois la relation d'ordre strict associée : on notera $\sigma \prec \tau$ lorsque $\sigma \preceq \tau$ et $\sigma \neq \tau$.

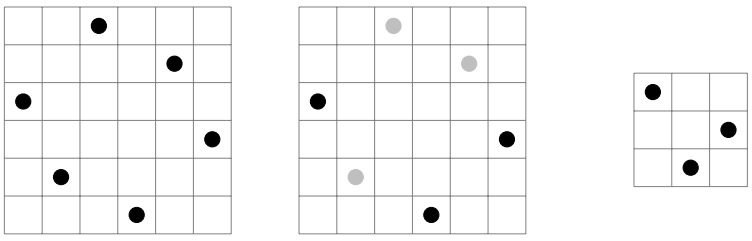
Dans la définition 1.14, certains éléments (ici, $\tau_{i_1}\tau_{i_2}\dots\tau_{i_k}$) de la permutation τ sont sélectionnés, et la normalisation de la séquence associée à ces éléments définit un motif de τ. Mais on peut avoir le point de vue inverse, et voir un motif comme obtenu après la suppression de certains éléments de τ (en l'occurrence, tous sauf $\tau_{i_1}\tau_{i_2}\dots\tau_{i_k}$). On définit ci-après la suppression d'un élément dans une permutation :

Définition 1.17. La *suppression* d'un élément j dans une permutation σ de taille k est la permutation obtenue après normalisation de la sous-séquence extraite de σ où seul l'élément de valeur j n'est pas conservé.

Exemple 1.18. La suppression de l'élément 4 dans la permutation $3\,1\,4\,5\,2\,6 \in \mathcal{S}_6$ donne la permutation $3\,1\,4\,2\,5 \in \mathcal{S}_5$.

La suppression de plusieurs éléments peut être définie de manière analogue, avec une unique étape de normalisation à la fin. Mais on s'aperçoit aisément que le résultat obtenu en procédant à des suppressions successives d'un élément (chacune associée à une normalisation) est identique. En particulier, la permutation obtenue ne dépend pas de l'ordre dans lequel les éléments sont supprimés. On propose seulement la définition de la suppression d'un unique élément, car c'est de cette opération dont il sera fait usage dans le paragraphe 6.3.2 du chapitre 6.

La notion de motif se visualise aussi très bien sur la représentation graphique des permutations. Pour décrire un motif d'une permutation σ, on efface une partie des points sur la représentation graphique de σ, et on supprime de la grille les colonnes et les lignes correspondantes. On obtient ainsi la représentation graphique d'un motif de σ.



La permutation $\tau = 4\,2\,6\,1\,5\,3$. Suppression de certains points de τ. $\sigma = 3\,1\,2$ est motif de τ.

FIG. 1.3 – La notion de motif sur la représentation graphique des permutations.

1.2.2 Les classes de permutations et leurs bases

Muni de la notion de motif, on peut à présent définir les classes de permutations :

Définition 1.19. Une *classe* $\mathcal{C}$ de permutations est un ensemble de permutations stable (ou fermé par le bas) pour la relation de motif $\preceq$. C'est-à-dire que pour toute permutation $\tau \in \mathcal{C}$, et pour tout motif σ de τ, σ appartient aussi à la classe $\mathcal{C}$.

On peut définir des classes de permutations par motifs exclus.

Définition 1.20. Fixant un ensemble $\mathcal{B}$ de permutations (les motifs exclus), on considère les permutations qui ne contiennent aucune occurrence d'aucun motif $\sigma \in \mathcal{B}$. L'ensemble de ces permutations est noté $\mathcal{S}(\mathcal{B})$.

Proposition 1.21. *Pour tout ensemble $\mathcal{B}$, $\mathcal{S}(\mathcal{B})$ est une classe de permutations.*

Démonstration. Il faut vérifier que $\mathcal{S}(\mathcal{B})$ est bien un ensemble stable pour la relation de motif. Soit donc $\tau \in \mathcal{S}(\mathcal{B})$ et $\sigma \preceq \tau$. Par transitivité, si σ contient un motif de $\mathcal{B}$, alors il en est de même pour τ, ce qui est interdit par la définition de $\mathcal{S}(\mathcal{B})$. Donc σ ne contient aucun motif de $\mathcal{B}$, démontrant le résultat annoncé. □

Pour permettre de définir avec unicité l'ensemble $\mathcal{B}$ de motifs exclus qui caractérise la classe de permutations $\mathcal{S}(\mathcal{B})$, il faut s'assurer qu'il n'y a aucune relation de motif entre les éléments de $\mathcal{B}$: si deux motifs distincts σ et τ appartiennent à $\mathcal{B}$, alors on ne doit avoir ni $\sigma \preceq \tau$, ni $\tau \preceq \sigma$. En effet, dans l'éventualité où on choisirait un ensemble $\mathcal{B}$ contenant deux permutations σ et τ telles que $\sigma \prec \tau$, les permutations évitant les motifs de $\mathcal{B}$ seraient les mêmes que celles évitant les motifs de $\mathcal{B} \setminus \{\tau\}$, puisque dans toute occurrence de τ apparaît nécessairement une occurrence de σ.

Les ensembles de permutations deux à deux incomparables (pour $\preceq$) sont appelés des *antichaînes* de permutations. Dans la définition 1.20, la classe de permutations $\mathcal{S}(\mathcal{B})$ ne peut donc être caractérisée par l'ensemble $\mathcal{B}$ de motifs exclus que dans le cas où $\mathcal{B}$ est une antichaîne de permutations. La proposition suivante montre que cette condition sur $\mathcal{B}$ est aussi suffisante.

Proposition 1.22. *Étant donnée une classe de permutations $\mathcal{C} = \mathcal{S}(\mathcal{B}_1) = \mathcal{S}(\mathcal{B}_2)$ décrite par deux ensembles $\mathcal{B}_1$ et $\mathcal{B}_2$ de motifs exclus, si $\mathcal{B}_1$ et $\mathcal{B}_2$ sont des antichaînes, alors $\mathcal{B}_1 = \mathcal{B}_2$.*

Démonstration. On montre que $\mathcal{B}_1 \subseteq \mathcal{B}_2$, l'autre inclusion se démontrant de la même manière. Soit $\sigma \in \mathcal{B}_1$. Alors $\sigma \notin \mathcal{C}$, de sorte que σ doit contenir une occurrence d'un motif de $\mathcal{B}_2$. Il existe donc $\tau \in \mathcal{B}_2$ tel que $\tau \preceq \sigma$. De la même manière, comme $\tau \in \mathcal{B}_2$, on a $\tau \notin \mathcal{C}$, et donc il existe un motif ρ de $\mathcal{B}_1$ tel que $\rho \preceq \tau$. Les deux motifs ρ et σ appartiennent donc à $\mathcal{B}_1$ et satisfont $\rho \preceq \tau \preceq \sigma$. Comme $\mathcal{B}_1$ est une antichaîne, on déduit que $\rho = \sigma$, et donc $\rho = \sigma = \tau$. Le motif τ appartenant à $\mathcal{B}_2$, on conclut que $\sigma \in \mathcal{B}_2$. □

La proposition 1.22 autorise donc la définition suivante :

Définition 1.23. Lorsque c'est une antichaîne, l'ensemble $\mathcal{B}$ de motifs exclus décrivant la classe $\mathcal{S}(\mathcal{B})$ est unique. Il est appelé la *base* de la classe de permutations $\mathcal{S}(\mathcal{B})$.

Remarque 1.24. Dans toute cette thèse, lorsqu'on considère une classe de permutations $\mathcal{S}(\mathcal{B})$, on sous-entendra toujours que $\mathcal{B}$ est la base de cette classe.

La terminologie de *base* peut prêter à confusion : contrairement aux bases des espaces vectoriels par exemple, les éléments de la base d'une classe de permutations $\mathcal{S}(\mathcal{B})$ *n'appartiennent pas* à la classe.

A priori, il peut sembler que les classes $\mathcal{S}(\mathcal{B})$ données par leur base $\mathcal{B}$ de motifs exclus sont plus restreintes que les classes de permutations, en toute généralité. En étudiant cette question, on s'aperçoit que c'est exactement le contraire qui se produit : toute classe de permutations, même si elle n'est pas explicitement donnée sous la forme $\mathcal{S}(\mathcal{B})$, peut être décrite par sa base de motifs exclus. Cette base est formée des éléments minimaux (au sens de $\preceq$) qui n'appartiennent pas à la classe.

Théorème 1.25. *Tout classe $\mathcal{C}$ de permutations peut être caractérisée par sa base $\mathcal{B}$ de motifs exclus, qui est unique. $\mathcal{B}$ est l'ensemble des permutations n'appartenant pas à $\mathcal{C}$ et minimales (au sens de $\preceq$) pour ce critère. Plus formellement,*

$$\mathcal{B} = \{\tau \notin \mathcal{C} : \forall \sigma \prec \tau, \sigma \in \mathcal{C}\}.$$

Démonstration. On considère une classe $\mathcal{C}$ de permutations, c'est-à-dire un ensemble stable pour la relation de motif $\preceq$. On pose $\mathcal{B} = \{\tau \notin \mathcal{C} : \forall \sigma \prec \tau, \sigma \in \mathcal{C}\}$. Il est clair de $\mathcal{B}$ est une antichaîne, et on prouve que $\mathcal{C} = \mathcal{S}(\mathcal{B})$.

On considère d'abord une permutation $\pi \notin \mathcal{S}(\mathcal{B})$. Il existe alors un motif $\tau \in \mathcal{B}$ tel que $\tau \preceq \pi$. Puisque $\tau \in \mathcal{B}$, on déduit que $\tau \notin \mathcal{C}$, et sachant que $\mathcal{C}$ est stable pour la relation $\preceq$, puisque $\tau \preceq \pi$, on obtient que $\pi \notin \mathcal{C}$.

À l'inverse, si $\pi \notin \mathcal{C}$, alors soit $\pi \in \mathcal{B}$ (et par conséquent $\pi \notin \mathcal{S}(\mathcal{B})$), soit il existe un motif $\sigma \prec \pi$ tel que $\sigma \notin \mathcal{C}$. Dans ce second cas, on procède par récurrence pour se ramener au premier cas, et ainsi prouver que $\pi \notin \mathcal{S}(\mathcal{B})$.

L'unicité de la base est assurée par la proposition 1.22. □

On peut remarquer que la base d'une classe de permutations n'a *a priori* aucune raison d'être finie. Bien que les bases des classes de permutations considérées dans la littérature soient souvent finies (voir le paragraphe 1.5), il existe des classes dont on sait que la base est infinie. C'est par exemple le cas pour la classe des permutations en épingles qu'on étudiera dans le chapitre 8, paragraphe 8.5. On verra qu'on peut construire une antichaîne infinie de permutations qui ne sont pas en épingles, mais telles que tous leurs motifs stricts le soient.

1.3 Trois opérations sur les permutations, qui conservent les motifs

Il est classique de définir trois opérations sur les permutations, qui se comportent bien vis-à-vis de la relation de motif : il s'agit des opérations de *miroir*, de *complément* et d'*inverse*.

Définition 1.26. Pour toute permutation $\sigma \in \mathcal{S}_n$, son *miroir* (ou renversé) σ^r, son *complément* σ^c, et son *inverse* σ^{-1} sont définies comme suit :
- pour tout $i \in [1..n], \sigma^r(i) = \sigma(n+1-i)$,
- pour tout $i \in [1..n], \sigma^c(i) = n+1-\sigma(i)$,
- pour tout $i \in [1..n], \sigma^{-1}(i) = j$ où j est l'entier tel que $\sigma(j) = i$.

L'inverse telle que définie dans la définition 1.26 correspond à la fonction réciproque de σ dans le groupe symétrique, c'est-à-dire qu'il s'agit bien de l'inverse de σ au sens de la théorie des groupes. Le miroir correspond à une lecture de droite à gauche de la permutation.

Les opérations de miroir, de complément et d'inverse correspondent, sur la représentation graphique des permutations, à des symétries du carré, comme illustré sur la figure 1.4. Le miroir correspond à une symétrie d'axe vertical, le complément à une symétrie d'axe horizontal, et l'inverse à une symétrie par rapport à la première diagonale.

En composant ces trois opérations, on peut obtenir un ensemble de permutations à partir d'une permutation donnée σ. Ce sont celles dont la représentation graphique peut être obtenue à partir de celle de σ en appliquant des symétries du carré. L'ensemble de permutations obtenues est cependant limité par les relations qui lient ces transformations (visualisées comme symétries du carré ou comme opérations sur les permutations) les unes aux autres.

Proposition 1.27. *Pour toute permutation σ, on a les identités suivantes :*

$$\begin{aligned} \sigma^{rc} &= \sigma^{cr} & \qquad \sigma^{rr} &= \sigma \\ \left(\sigma^r\right)^{-1} &= \left(\sigma^{-1}\right)^c & \qquad \sigma^{cc} &= \sigma \\ \left(\sigma^c\right)^{-1} &= \left(\sigma^{-1}\right)^r & \qquad \left(\sigma^{-1}\right)^{-1} &= \sigma \end{aligned}$$

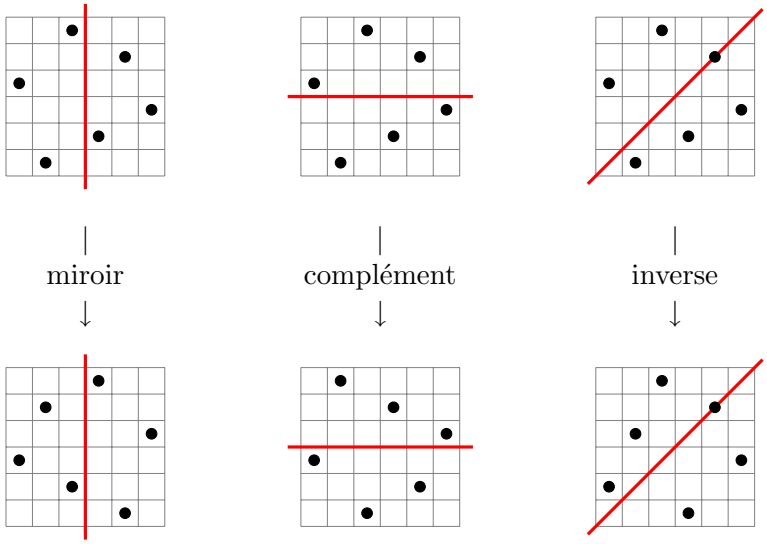


FIG. 1.4 – Les opérations de miroir, de complément et d'inverse, sur la représentation graphique des permutations.

Démonstration. Chacune de ces identités se vérifie sans difficulté, par exemple sur la représentation standard en une ligne des permutations, ou plus facilement encore sur leur représentation graphique. □

Il est connu que le groupe des symétries du carré (ou groupe diédral D_4) contient 8 éléments. En appliquant les 8 transformations correspondantes à une permutation σ, on obtient un ensemble d'au plus 8 permutations, appelé *classe de symétrie* de σ. Une manière de décrire cet ensemble de permutations est la suivante :

Définition 1.28. La *classe de symétrie* de σ est l'ensemble de permutations

$$\{\sigma, \sigma^r, \sigma^c, \sigma^{-1}, \sigma^{rc}, (\sigma^r)^{-1}, (\sigma^c)^{-1}, (\sigma^{rc})^{-1}\}.$$

Avec la proposition 1.27, on peut vérifier que l'application d'une opération (miroir, complément ou inverse) sur l'un des 8 éléments de cet ensemble donne bien un élément de l'ensemble. On peut aussi vérifier que ces 8 éléments sont *a priori* distincts. Mais il est possible que dans certains cas, la même permutation soit représentée plusieurs fois dans cet ensemble. Dans ce cas, la classe de symétrie considérée est de cardinal 1, 2 ou 4.

Exemple 1.29.
L'unique permutation dont la classe de symétrie est de taille 1 est la permutation $1 \in \mathcal{S}_1$.
La classe de symétrie de 2 4 1 3 est $\{2\,4\,1\,3, 3\,1\,4\,2\}$, et a donc cardinal 2.
La classe de symétrie de 2 4 1 6 3 5, de cardinal 4, est $\{2\,4\,1\,6\,3\,5, 5\,3\,6\,1\,4\,2, 3\,1\,5\,2\,6\,4, 4\,6\,2\,5\,1\,3\}$.
Une classe de symétrie de cardinal 8 est celle de 5 2 4 1 6 3, qui est formée de l'ensemble des 8 permutations suivantes : $\{5\,2\,4\,1\,6\,3, 3\,6\,1\,4\,2\,5, 2\,5\,3\,6\,1\,4, 4\,2\,6\,3\,1\,5, 4\,1\,6\,3\,5\,2, 3\,5\,1\,4\,6\,2, 5\,1\,3\,6\,2\,4, 2\,6\,4\,1\,5\,3\}$.

Les opérations de miroir, de complément et d'inverse ont de bonnes propriétés vis-à-vis de la relation de motif :

Proposition 1.30 ([SS85]). *Pour toutes permutations σ et τ, si $\sigma \preceq \tau$, alors* $\begin{cases} \sigma^r \preceq \tau^r, \\ \sigma^c \preceq \tau^c, \\ \sigma^{-1} \preceq \tau^{-1}. \end{cases}$

Démonstration. La propriété 1.30 est évidente sur la représentation graphique des permutations. On peut cependant aussi la démontrer facilement sur leur représentation en une ligne.

Par définition, comme $\sigma \preceq \tau$, en notant k et n les tailles respectives de σ et τ, il existe des entiers $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ tels que $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$ soit isomorphe en ordre à σ. On démontre alors sans difficulté que $\begin{cases} \tau^c_{i_1}\tau^c_{i_2}\ldots\tau^c_{i_k} \text{ est isomorphe en ordre à } \sigma^c, \\ \tau^r_{n+1-i_k}\ldots\tau^r_{n+1-i_2}\tau^r_{n+1-i_1} \text{ est isomorphe en ordre à } \sigma^r, \\ \tau^{-1}_{j_1}\tau^{-1}_{j_2}\ldots\tau^{-1}_{j_k} \text{ est isomorphe en ordre à } \sigma^{-1}, \end{cases}$
où les entiers $1 \leq j_1 < j_2 < \ldots < j_k \leq n$ sont définis par $\{i_1, i_2, \ldots, i_k\} = \{\tau^{-1}_{j_1}, \tau^{-1}_{j_2}, \ldots, \tau^{-1}_{j_k}\}$. □

La stabilité de la notion de motif à travers les trois opérations de miroir, de complément et d'inverse permet de définir des classes dites *symétriques* d'une classe de permutations $\mathcal{C}$, et de décrire leurs bases à partir de la base de $\mathcal{C}$.

Définition 1.31. Étant donnée une classe $\mathcal{C}$ de permutations, ses classes *symétriques* sont les classes obtenues en appliquant une même transformation (miroir, complément, inverse, ou une transformation composée à partir de celles-ci) sur chaque permutation de la classe. Par exemple, la classe miroir de $\mathcal{C}$ est $\mathcal{C}^r = \{\sigma^r : \sigma \in \mathcal{C}\}$.

De même que la classe de symétrie d'une permutation donnée contient au plus 8 éléments, chaque classe de permutations a au plus 8 classes symétriques.

On peut en outre caractériser la base des classes symétriques de $\mathcal{C}$ à partir de la base de la classe $\mathcal{C}$:

Proposition 1.32. *On considère une classe $\mathcal{C} = \mathcal{S}(\mathcal{B})$, et une de ses classes symétriques $\mathcal{C}^t$ obtenue par application d'une transformation t à toutes les permutations de $\mathcal{C}$. Alors $\mathcal{C}^t = \mathcal{S}(\mathcal{B}^t)$, où $\mathcal{B}^t = \{\sigma^t : \sigma \in \mathcal{B}\}$.*

Démonstration. Pour démontrer ce résultat, il suffit de s'apercevoir que la proposition 1.30, associée au fait que les transformations de miroir, complément et inverse satisfont $\left(\sigma^r\right)^r = \left(\sigma^c\right)^c = \left(\sigma^{-1}\right)^{-1} = \sigma$, donne en fait une équivalence entre les quatre propriétés $\sigma \preceq \tau$, $\sigma^r \preceq \tau^r$, $\sigma^c \preceq \tau^c$ et $\sigma^{-1} \preceq \tau^{-1}$. □

Cela implique que l'étude de deux classes symétriques, en particulier du point de vue énumératif, est redondante : les propriétés d'une classe se transportent directement à ses classes symétriques, via les opérations de miroir, complément, inverse et leurs composées. On reviendra sur les transports de propriétés énumératives d'une classe à ses symétriques dans le paragraphe 1.5.

1.4 Quelques problématiques combinatoires et algorithmiques de l'étude des classes de permutations

On présente maintenant quelques problématiques générales, adoptées dans la littérature et dans cette thèse, pour l'étude des classes de permutations.

1.4.1 Problèmes d'énumération et conjecture de Stanley-Wilf

Étant donnée une classe $\mathcal{C}$ de permutations, une première question qu'il est naturel de se poser est celle de l'énumération de la classe $\mathcal{C}$: pour tout $n \in \mathbb{N}$, combien y a-t-il de permutations de taille n dans la classe $\mathcal{C}$?

Pour toute classe $\mathcal{C}$ de permutations, on notera $\mathcal{C}_n = \mathcal{C} \cap \mathcal{S}_n$ l'ensemble des permutations de taille n dans $\mathcal{C}$. On s'intéresse donc à estimer le cardinal de $\mathcal{C}_n$ en fonction de n, pour toute classe $\mathcal{C}$. Cette question a été étudiée pour de nombreuses classes $\mathcal{C} = \mathcal{S}(\mathcal{B})$ données par leur base, comme on le verra au paragraphe 1.5.

Avec une vision plus globale, R. P. Stanley et H. S. Wilf ont conjecturé au début des années 1990 que, pour toute classe $\mathcal{C}$, il existe une constante c telle que pour tout $n \in \mathbb{N}, |\mathcal{C}_n| \leq c^n$. Il existe en fait une conjecture plus forte (c'est même cette version forte qui a été formulée au départ par R. P. Stanley et H. S. Wilf) qui propose que pour tout motif σ, en notant $\mathcal{C} = \mathcal{S}(\sigma)$, la suite $|\mathcal{C}_n|^{\frac{1}{n}}$ a une limite, qui est finie, lorsque $n \to \infty$. R. Arratia a démontré dans [Arr99] que ces deux versions de la conjecture sont en fait équivalentes lorsque la base $\mathcal{B}$ de la classe contient un unique motif. Le problème de l'existence de la limite de $|\mathcal{C}_n|^{\frac{1}{n}}$ pour les classes $\mathcal{C}$ dont la base n'est pas réduite à un motif reste ouvert.

Parallèlement, des avancées dans la preuve de cette conjecture ont été obtenues par M. Bóna, qui l'a démontrée dans [Bón99] pour des motifs exclus restreints (les *layered permutations*), ou de taille 4 [Bón97]. Mais la clé qui a permis de démontrer la conjecture dans le cas général a été trouvée par M. Klazar, qui montre dans [Kla00] que la conjecture de Füredi-Hajnal sur les matrices 0/1 implique la conjecture de Stanley-Wilf.

La démonstration de la conjecture de Füredi-Hajnal a été donnée quatre années plus tard, par A. Marcus et G. Tardos dans [MT04], et est étonnamment simple pour une preuve recherchée pendant 15 ans. Néanmoins, le résultat obtenu est sans aucun doute le résultat majeur de ces dernières années dans l'étude combinatoire des classes de permutations :

Théorème 1.33 (Conjecture de Stanley-Wilf, Théorème de Marcus-Tardos)**.**
Pour toute classe $\mathcal{C}$ de permutations, il existe une constante c telle que pour tout $n \in \mathbb{N}, |\mathcal{C}_n| \leq c^n$.

En particulier, en comparaison du nombre $n!$ de permutations de taille n, toute classe de permutations contient une fraction asymptotiquement négligeable de l'ensemble de toutes les permutations.

Le sujet n'est pas clos pour autant. Les recherches sur l'énumération des classes de permutations dans leur ensemble se portent maintenant sur les estimations de la valeur de la constante c du théorème 1.33, en particulier quand la classe $\mathcal{C}$ est définie par un unique motif exclu σ. Lorsque $\mathcal{C} = \mathcal{S}(\sigma)$, on cherche une borne sur $c = c(\sigma)$ en fonction de la taille k de σ.

Dans la preuve de A. Marcus et G. Tardos, la borne obtenue est très large : $c(\sigma) \leq 15^{2k^4\binom{k^2}{k}}$. Cette borne a été améliorée par M. Bóna dans [Bón04b], puis par J. Cibulka dans [Cib09], pour atteindre $b^{\mathcal{O}(k \log k)}$ pour une constante b. Les valeurs obtenues restent cependant exponentielles en k, et sont donc encore bien au delà des bornes polynomiales (et même quadratiques) en k qui sont conjecturées.

La conjecture la plus fameuse dans ce domaine est due à R. Arratia, et prétend que $c(\sigma) \leq (k-1)^2$. Cette conjecture est entre autre étayée par le fait que $c(12\dots k) = (k-1)^2$ [Reg81]. Les efforts déployés pour démontrer cette conjecture n'ont jamais abouti, et pour cause. La conjecture d'Arratia a été infirmée par M.H. Albert, M. Elder, A. Rechnitzer, P. Westcott et M. Zabrocki dans [AER+06], où il est démontré que $c(4231) \geq 9.47$.

1.4.2 Décider l'appartenance à une classe

Dans une étude des classes de permutations d'un point de vue algorithmique, un des premiers problèmes à envisager est celui du test d'appartenance à une classe. Étant donnée une classe $\mathcal{C}$ de permutations, on veut pouvoir décider, pour toute permutation τ, si $\tau \in \mathcal{C}$, et dans la mesure du possible le décider efficacement.

On peut d'abord remarquer que, lorsqu'une classe $\mathcal{C}$ est décrite par sa base $\mathcal{B}$ de motifs exclus, et qu'en outre cette base est finie, un algorithme naïf de test d'appartenance à la classe $\mathcal{C}$ est obtenu

en effectuant plusieurs tests d'occurrence de motif. En effet, pour tester si une permutation $\tau \in \mathcal{C}$, il suffit de vérifier, pour chaque motif de $\mathcal{B}$, que ce motif ne possède aucune occurrence dans τ.

Toujours naïvement, pour tester si un motif σ de taille k possède une occurrence dans une permutation τ de taille n, il suffit d'examiner chaque sous-séquence de k éléments extraite de τ, et de tester si elle est isomorphe en ordre à σ.

Avec ces deux idées, lorsqu'une classe $\mathcal{C}$ est donnée par sa base finie $\mathcal{B}$ de motifs exclus, on obtient donc un algorithme de test d'appartenance à $\mathcal{C}$ de complexité $\mathcal{O}(m \cdot k \cdot n^k)$, les paramètres k, m et n désignant respectivement la taille maximale d'un élément de $\mathcal{B}$, le nombre d'éléments de $\mathcal{B}$, et la taille de la permutation τ dont on veut tester l'appartenance à $\mathcal{C}$. Le facteur k est dû à la normalisation de chaque sous-séquence extraite de τ, comme cela sera expliqué plus en détails page 68. On verra aussi dans le paragraphe 3.1.4 une méthode proposée dans [AAAH01] pour accélérer la recherche d'occurrence d'un motif dans une permutation.

La procédure naïve décrite ici est d'une complexité élevée, et donc peu satisfaisante. Pour certaines classes de permutations, des algorithmes de test d'appartenance spécifiques ont été développés, et permettent de résoudre ce problème beaucoup plus efficacement. On peut citer par exemple la classe $\mathcal{S}(231)$ des permutations triables par pile, ou la classe $\mathcal{S}(2413, 3142)$ des permutations séparables. Pour ces deux classes, les algorithmes *ad hoc* de test d'appartenance ont une complexité en $\mathcal{O}(n)$, à comparer à $\mathcal{O}(n^3)$ ou $\mathcal{O}(n^4)$ pour l'algorithme naïf.

Les algorithmes linéaires de reconnaissance des permutations triables par pile et des permutations séparables sont brièvement présentés dans cette thèse, respectivement dans les paragraphes 4.1.3 et 3.2.1. De plus amples détails peuvent être trouvés dans [Knu73, BBL98].

Un défaut majeur de l'algorithme naïf de test d'appartenance présenté ci-dessus est qu'il ne s'applique qu'à des classes dont on connaît la base de motifs exclus, et dans le cas où cette base est finie. On peut donc se demander s'il est possible d'identifier les classes qui ont une base finie, et le cas échéant de calculer cette base.

Une première ébauche de caractérisation a été proposée dans [AMR02], où M. D. Atkinson, M. Murphy et N. Ruškuc s'intéressent aux classes possédant la propriété de base finie forte (*strongly finitely based classes* en anglais). Ces classes sont celles possédant une base finie, et dont toutes les sous-classes ont aussi une base finie.

Plus récemment, M. H. Albert et M.D. Atkinson ont donné une condition suffisante pour qu'une classe ait une base finie [AA05] : il suffit qu'elle contienne un nombre fini de permutations simples. On peut remarquer que cette condition est aussi suffisante pour que la classe de permutations ait une série génératrice algébrique.

Suite à ce résultat, R. Brignall, N. Ruškuc et V. Vatter ont décrit dans [BRV08] une procédure décidant si une classe contient un nombre fini de permutations simples. Malheureusement, cette procédure suppose que la classe de permutations soit donnée par sa base finie. Elle fournit donc un moyen effectif de tester une condition suffisante pour qu'une classe de permutation, donnée par sa base finie, ait une série génératrice algébrique, mais ne permet pas de tester que sa base soit finie.

Ces conditions suffisantes d'algébricité et les procédures effectives pour les tester seront étudiées plus en détails dans la partie IV de la thèse.

1.4.3 Conception d'algorithmes efficaces

Comme tous les champs de l'algorithmique, celle des permutations regorge de problèmes de complexités variées, et les problèmes qui nous intéressent principalement dans ce paragraphe sont les problèmes NP-difficiles sur les permutations. Il est communément admis qu'il n'existe pas d'algorithmes polynomiaux résolvant ces problèmes en toute généralité, mais on peut imaginer les restreindre grâce à des classes de permutations, de sorte à rendre les sous-problèmes ainsi définis résolubles en temps polynomial.

Des exemples de tels problèmes seront étudiés dans la partie II de cette thèse : il s'agit des problèmes de recherche d'occurrence d'un motif dans une permutation, et de recherche de plus

grand motif commun à un ensemble de permutations. La recherche d'occurrence de motif est un problème NP-complet [BBL98], ce qui implique que la recherche de plus grand motif commun est NP-difficile. Cependant, on verra qu'en restreignant les permutations en entrée ou le motif recherché à la classe des permutations séparables, on peut donner des solutions par des algorithmes polynomiaux aux sous-problèmes obtenus. Les restrictions peuvent porter aussi bien sur les entrées du problème (recherche d'occurrence d'un motif séparable dans une permutation quelconque, au paragraphe 3.3) que sur le résultat attendu en sortie (recherche de plus grand motif séparable commun à deux permutations quelconques, au paragraphe 5.1).

La restriction de problèmes NP-difficiles à des classes de permutations, dans le but d'obtenir des sous-problèmes polynomiaux, peut paraître artificielle et injustifiée. Dans certains contextes, ces restrictions sont au contraire tout à fait pertinentes. C'est en particulier le cas lorsque les permutations modélisent d'autres objets, et que la classe de permutations considérée correspond à une contrainte sur les objets qui s'avère naturelle dans le modèle étudié.

Par exemple, le chapitre 7 s'intéresse au problème de calcul de scénarios parfaits de tri par renversements de permutations signées. Ce problème est NP-difficile en général [FV04], mais l'algorithme FPT proposé par S. Bérard, A. Bergeron, C. Chauve et C. Paul dans [BBCP07] est polynomial lorsque les entrées du problème sont restreintes à la classe des permutations séparables (signées). Or le problème du tri parfait par renversements trouve ses motivations dans l'étude de l'évolution des génomes, et dans ce contexte, la plupart des données biologiques observées correspondent à des permutations séparables. Ainsi, la restriction du problème à la classe des permutations séparables est très judicieuse ici : elle est parfaitement justifiée par le modèle et permet d'obtenir un algorithme polynomial, pour un problème qui resterait NP-difficile si l'on n'introduisait pas cette restriction.

1.4.4 Décrire une classe sans connaitre sa base

Hormis quelques travaux récents, la plupart des recherches sur les classes de permutations portent sur des classes décrites par une base finie de motifs exclus. Mais cette description (quand elle est possible, la base d'une classe pouvant être infinie) n'est pas toujours la plus naturelle pour définir une classe de permutations. On rappelle qu'une classe de permutations n'est pas autre chose qu'un ensemble de permutations stable pour la relation de motif. Il suffit donc pour décrire une classe de permutations de se donner une propriété stable pour la relation de motif, et de définir la classe comme l'ensemble des permutations satisfaisant cette propriété.

Le premier exemple de classe décrite de cette façon est dû à D. E. Knuth, qui définit dans [Knu73] les permutations triables par une pile. La propriété d'être triable par une pile se transmet naturellement d'une permutation à tous les motifs qu'elle contient, permettant de parler de classe de permutations. Cette classe sera présentée plus en détails dans le paragraphe 4.1.3.

Plus récemment, d'autres classes définies par une propriété ont été considérées. Les permutations séparables peuvent être décrites comme celles qui possèdent un arbre de séparation, cette propriété passant directement aux motifs : on reviendra abondamment sur la classe des permutations séparables dans la partie II de ce travail.

L'idée de D. E. Knuth de considérer les permutations triables par une pile peut être renversée, pour étudier les permutations obtenues après un passage d'une permutation identité par une pile. En s'inspirant de cette idée, mais de manière plus générale, M. H. Albert, R. E. L. Aldred, M. D. Atkinson, H. P. van Ditmarsch, C. C. Handley, D. A. Hotlon et D. J. McCaughan se sont intéressés aux permutations obtenues en sortie d'un dispositif de mélange, prenant en entrée une permutation identité. Le formalisme des *permuting machines* a été introduit dans [AAA$^+$07] pour mener une étude des classes de permutations ainsi définies.

Des classes de permutations ont aussi été définies en se donnant des classes plus simples, et un moyen de les mélanger. Par exemple, M. D. Atkinson examine dans [Atk99] les permutations

qui sont le "mélange" de deux permutations croissantes, ou d'une permutation croissante et d'une décroissante. Toujours dans l'idée de construire des classes au moyen de classes plus petites, des efforts ont été menés pour l'étude du *produit de substitution* (ou *produit en couronne*) sur les classes de permutations [AS02, Bri07, par exemple]. Une présentation formelle de cette opération sur les classes de permutations sera donnée au paragraphe 2.4.

Avec tout l'intérêt qui est porté actuellement à l'étude des permutations représentées graphiquement, il est naturel aussi que des classes de permutations aient été définies par des propriétés de leurs représentations graphiques. C'est le cas par exemple de la classe des permutations en épingles, qui est au centre de l'étude menée dans la partie IV de cette thèse.

Il existe donc de nombreuses façons de définir une classe de permutations autrement qu'en se donnant un ensemble de motifs exclus. À l'heure actuelle, on dispose de peu d'outils pour l'étude de ces classes de permutations, définies ainsi de manière "non standard". Pour pouvoir étudier ces classes, une première étape consiste souvent à donner une caractérisation des permutations de la classe, autrement que par la propriété qui la définit. On cherchera des caractérisations peut-être moins naturelles au départ, mais qui se prêtent mieux à des manipulations combinatoires.

Dans certains cas, il est possible de calculer la base de la classe à partir de la propriété qui la définit, et ainsi de disposer de toutes les techniques pour l'étude des permutations à motifs exclus. C'est par exemple ce qui se passe pour les permutations triables par pile (évitant 231) ou séparables (évitant 2413 et 3142). Cependant, les situations où ce type de caractérisation aboutit sont rares, et limitées théoriquement par le fait que les classes de permutations peuvent avoir des bases infinies de motifs exclus. Une question cruciale en ce sens est de pouvoir déterminer si une classe possède ou non une base finie. Ce problème n'est pas complètement résolu, mais des outils ou critères de décision ont été proposés dans [AS02, AA05].

Dans d'autres cas, on peut proposer une description récursive des permutations appartenant à la classe. Les objets définis récursivement sont très répandus en combinatoire, et si l'on trouve une telle caractérisation des permutations de la classe, on peut profiter de techniques éprouvées pour en mener l'étude. C'est par exemple de cette manière qu'on pourra calculer la série génératrice d'énumération des permutations en épingles dans la partie IV.

L'étude des classes de permutations dont la base n'est pas connue est en tout cas un domaine de recherche très actuel, et on peut s'attendre à ce que soient développées dans les années à venir de nouvelles techniques pour mener cette étude.

1.5 Étude combinatoire des classes de permutations définies par leur base : quelques pointeurs bibliographiques

On revient dans ce paragraphe à une approche plus classique d'étude de classes de permutations définies par leur base finie de motifs exclus. Le but est de proposer au lecteur un panorama des résultats existants, absolument non exhaustif, mais qui soit dans une certaine mesure représentatif des types de résultats attendus, et des techniques utilisées pour y parvenir.

La thèse de O. Guibert [Gui95] présente un panorama beaucoup plus détaillé de l'état des connaissances en combinatoire sur les permutations à motifs exclus en 1995. Outre les travaux de O. Guibert sur certaines classes de permutations énumérées par des suites classiques (nombres de Catalan, de Motzkin, de Pell, de Schröder, ...), et les bijections qu'il donne entre des classes de permutations et d'autres objets classiques de combinatoire (certaines familles de cartes planaires, tableaux de Young, combinaisons de mots de parenthèses, arbres $1-2$, permutations de Baxter, permutations vexillaires, ...), cette thèse propose en annexe un catalogue qui répertorie tous les résultats d'énumération connus à sa date de publication sur les permutations à motifs exclus, ou en tout cas bon nombre d'entre eux.

Devant l'intérêt croissant pour les permutations et leurs motifs, S. Kitaev et T. Mansour ont proposé dans [KM03] un survol des résultats connus dans ce domaine en 2003. On y trouve non

seulement de nombreux éléments pour l'étude des permutations évitant des motifs, mais aussi des aspects de l'analyse des permutations qui contiennent un nombre donné d'occurrences d'un motif, et un parallèle entre motifs dans les permutations et motifs dans les mots.

Même s'il manque tous les résultats récents dans ces références, et en particulier la preuve de la conjecture de Stanley-Wilf, elles constituent une bonne première approche du domaine.

1.5.1 Motifs exclus de petite taille

Suite aux travaux de D. E. Knuth en 1973 caractérisant les permutations triables par pile comme celles évitant le motif 231, un examen plus ou moins systématique des classes de permutations définies par un ou plusieurs motif(s) exclu(s) de taille 3 ou 4 a été entrepris. Grâce aux transferts de propriétés d'une classe de permutations à une autre, quand sont appliquées sur les motifs de sa base des opérations composées à partir du miroir, du complément et de l'inverse, cette tâche se révèle moins étendue que ce qu'on pourrait penser.

Les permutations de taille 3 se répartissent en deux classes de symétries : $\{123, 321\}$ et $\{132, 213, 231, 312\}$. Il suffit donc, modulo symétrie, d'étudier deux classes de permutations définies par exclusion d'un seul motif de taille 3. Les premiers travaux sur $\mathcal{S}(123)$ remontent à P. A. MacMahon en 1915-1916 [Mac16], et comme on l'a déjà mentionné plusieurs fois, c'est D. E. Knuth en 1973 dans [Knu73] qui s'est intéressé le premier à la classe $\mathcal{S}(231)$. Dans les deux cas, la séquence d'énumération des permutations de ces classes est donnée par les nombres de Catalan[a]. On mentionne aussi qu'il existe des bijections permettant de passer de permutations évitant un motif $\sigma \in \{123, 321\}$ à celles évitant un motif $\tau \in \{132, 213,\ 231, 312\}$. Par exemple, les bijections entre $\mathcal{S}(321)$ et $\mathcal{S}(132)$ sont rassemblées dans [CK08].

Les classes de permutations $\mathcal{S}(Q)$ pour $Q \subseteq \mathcal{S}_3$ ont été étudiées exhaustivement par R. Simion et F. Schmidt [SS85], d'un point de vue énumératif. Il est rendu compte de tous les résultats obtenus en ce sens dans [KM03]. On peut remarquer que pour deux motifs de taille 3 évités, il y a seulement cinq séquences d'énumération possibles.

Des résultats supplémentaires sur les motifs exclus de taille 3 ont été obtenus par V. Vatter dans [Vat03], où des contraintes viennent s'ajouter à l'exclusion d'un ou de plusieurs motifs de taille 3. Deux possibilités en ce sens sont envisagées : l'exclusion d'un motif supplémentaire, de taille arbitraire, ou l'obligation de contenir exactement une occurrence d'un motif, également de taille arbitraire.

Sont considérées ensuite les classes $\mathcal{S}(\sigma)$ pour $\sigma \in \mathcal{S}_4$. Il y a sept classes de symétrie pour les motifs de taille 4, mais il suffit en fait d'étudier trois motifs de taille 4, les autres donnant les mêmes séquences d'énumération. Pour une démonstration de ce résultat, et une étude énumérative des classes correspondantes, on renvoie le lecteur à [Bón97] et ses références. Les classes de permutations définies par exclusion d'un motif de taille 4 sont dans l'ensemble bien comprises, à une exception notable près. L'énumération des permutations évitant le motif 4231 (ou 1324) reste ouverte, et est un sujet de recherche actif. Récemment, M. H. Albert, M. Elder, A. Rechnitzer, P. Westcott et M. Zabrocki ont réfuté dans [AER$^+$06] la conjecture d'Arratia sur la valeur de c dans le théorème 1.33, en donnant une borne inférieure sur le taux de croissance de la classe $\mathcal{S}(4231)$.

1.5.2 Généralisations de la notion de motifs exclus

Tout comme les permutations triables par une pile ont motivé l'étude des permutations à motifs exclus, c'est l'étude des permutations triables par plusieurs piles qui a été à l'origine de la généralisation de la notion de motif dans les permutations.

Dans sa thèse [Wes90], J. West a défini les motifs *barrés* pour caractériser les permutations triables par deux piles.

[a] Les nombres de Catalan seront définis au chapitre 10, pour illustrer la définition et l'utilisation des séries génératrices.

Définition 1.34. Un *motif barré* σ de $\overline{\mathcal{S}_n}$ est une permutation de $\mathcal{S}_n$ dont certains éléments sont surmontés d'une barre.

Un tel motif code de manière compacte deux permutations $\underline{\sigma}$ et $\overline{\sigma}$, avec $\underline{\sigma} \preceq \overline{\sigma}$: $\underline{\sigma}$ est la permutation obtenue après normalisation en supprimant les éléments barrés de σ, et $\overline{\sigma}$ est la permutation σ où toutes les barres ont été effacées.

Exemple 1.35. Par exemple, $\sigma = 2\,1\,\overline{3}\,5\,4$ est un motif barré, qui code les deux permutations $\underline{\sigma} = 2\,1\,4\,3$ et $\overline{\sigma} = 2\,1\,3\,5\,4$.

Définition 1.36. Une permutation $\tau \in \mathcal{S}$ contient un motif barré σ si toute occurrence (au sens classique) de $\underline{\sigma}$ dans τ peut être étendue en une occurrence de $\overline{\sigma}$.

Exemple 1.37. Par exemple, la permutation $\tau = 2\,1\,4\,3\,6\,5$ ne contient pas le motif barré $\sigma = 2\,1\,\overline{3}\,5\,4$: bien que l'occurrence $2\,1\,6\,5$ de $\underline{\sigma} = 2\,1\,4\,3$ dans τ puisse être étendue en une occurrence de $\overline{\sigma} = 2\,1\,3\,5\,4$ (par exemple, $2\,1\,3\,6\,5$), les deux autres occurrences de $\underline{\sigma}$ dans τ, à savoir $2\,1\,4\,3$ et $4\,3\,6\,5$ ne peuvent pas être étendues en occurrences de $\overline{\sigma}$.

Il faut bien remarquer que dans la définition 1.36 seul le motif a des éléments barrés. La permutation τ est une permutation de $\mathcal{S}$, sans barres.

Les motifs barrés ont donc été introduits au départ pour caractériser les permutations triables par deux piles, et J. West démontre dans [Wes90] que ces permutations sont celles évitant les motifs $2\,3\,4\,1$ et $3\,\overline{5}\,2\,4\,1$. Depuis, de plus amples investigations sur les permutations évitant des motifs barrés ont été entreprises. Pour s'en faire une idée, on pourra par exemple consulter [Pud], qui regroupe tous les résultats d'énumération connus à ce jour pour les motifs barrés de taille au plus 4, mais aussi pour certains ensembles de motifs barrés, et pour des motifs barrés de taille supérieure à 4.

Une autre généralisation de la notion de motif est apparue au début des années 2000, pour tenir compte des positions, contigües ou non, des éléments utilisées par une occurrence d'un motif dans une permutation. Ces motifs qui introduisent des contraintes d'adjacence entre éléments sont les *motifs à tirets*.

Définition 1.38. Un *motif à tirets* de taille k est une permutation de $\mathcal{S}_k$ dans laquelle entre chaque paire d'éléments à des positions consécutives peut figurer un tiret $-$.

Exemple 1.39. Un motif à tirets de taille 8 est $6\,3\,5 - 1 - 2\,8 - 4\,7$.

Définition 1.40. Un motif à tirets σ de taille k possède une occurrence dans la permutation $\tau \in \mathcal{S}_n$ s'il possède une occurrence dans τ au sens classique (il existe des entiers $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ tels que la séquence d'entiers distincts $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$ est isomorphe en ordre au motif σ où les tirets ont été effacés), avec la contrainte supplémentaire que lorsque σ_j et σ_{j+1} ne sont pas séparés par un tiret dans σ, alors on doit avoir $i_{j+1} = i_j + 1$.

Formulé autrement, cela signifie que les éléments qui sont contigus (c'est-à-dire qui ne sont pas séparés par un tiret) dans le motif σ doivent aussi être contigus dans une occurrence de σ.

Exemple 1.41. La permutation $3\,4\,1\,5\,2$ contient le motif $\sigma = 2\,3 - 1$, comme en témoigne l'occurrence $3\,4\,2$. En revanche, la permutation $\tau = 3\,1\,5\,4\,2$ ne contient pas le motif σ : les deux seules occurrences de $2\,3\,1$ dans τ sont $3\,5\,2$ et $3\,4\,2$, et dans les deux cas, les éléments correspondant aux valeurs 2 et 3 de $2\,3\,1$ ne sont pas à des positions consécutives dans τ.

La notation avec des tirets pour indiquer les éléments qui ne sont pas nécessairement adjacents peut prêter à confusion par rapport aux motifs standards. En effet, dans le contexte des motifs à tirets, un motif σ écrit sans aucun tiret indique que les occurrences de ce motif dans une permutation τ doivent correspondre à des séquences d'entiers qui se trouvent à des positions consécutives dans

τ. Pour retrouver la notion d'occurrence d'un motif au sens classique, on doit faire figurer un tiret entre chaque paire d'éléments de σ à des positions consécutives.

Les motifs à tirets ont été définis par E. Babson et E. Steingrímsson dans [BS00a] pour étudier les différents comportements possibles d'une famille de statistiques sur les permutations, dites statistiques mahoniennes. Ces statistiques sont celles ayant la même distribution que le nombre d'inversions. Plusieurs statistiques mahoniennes avaient préalablement émergé dans la littérature : le *major index*, connu dès P. A. MacMahon, de nombreuses statistiques basées sur les descentes des permutations (entre autre la statistique MAK de Foata et Zeilberger), la statistique de Denert et les autres statistiques basées sur les excédances des permutations (c'est-à-dire les positions i telles que $\sigma_i > i$)...

La démarche de E. Babson et E. Steingrímsson a consisté à proposer une classification de ces statistiques, en quatorze groupes, qui permettent d'expliquer les différences de comportement de la plupart des statistiques mahoniennes connues.

Très rapidement après la définition des motifs à tirets, les questions d'énumération des permutations qui les évitent ont été envisagées. A. Claesson a résolu l'énumération des permutations évitant un motif à tirets de taille 3 dans [Cla01], puis A. Claesson et T. Mansour ont donné l'énumération des permutations évitant deux tels motifs dans [CM05]. Les permutations évitant un ensemble de trois motifs à tirets de taille 3 ont même été considérées, et leur énumération est un résultat de A. Bernini, L. Ferrari et R. Pinzani dans [BFP05].

Il a aussi été envisagé d'énumérer les permutations évitant des motifs à tirets en fonction de leur taille et de la valeur d'un autre paramètre. J'ai obtenu des résultats en ce sens en collaboration avec Antonio Bernini et Luca Ferrari, publiés dans [BBF07].

Un article récent de R. Brignall, S. B. Ekhad, R. Smith et V. Vatter a remis au goût du jour une autre manière de généraliser les permutations évitant des motifs. La définition classique de motif est conservée, mais on étudie les permutations qui évitent *presque* les motifs considérés.

Plusieurs notions de "quasi-évitement" de motifs ont été proposées dans la littérature. Dans [Noo96], J. Noonan étudie les permutations qui contiennent au plus un certain nombre r d'occurrences d'un motif exclu [Noo96]. Ces classes de permutations sont notées $\mathcal{S}(\sigma^{\leq r})$. Dans [BESV], R. Brignall, S. B. Ekhad, R. Smith et V. Vatter ont une définition différente du quasi-évitement : ils étudient les permutations qui évitent un motif, à la suppression de r éléments de la permutation près. La notation adoptée pour les classes ainsi définies est $\mathcal{S}(\sigma)^{+r}$. Bien que le point de vue soit tout à fait différent, la deuxième notion de quasi-évitement est en fait une généralisation de la première : l'inclusion $\mathcal{S}(\sigma^{\leq 1}) \subsetneq \mathcal{S}(\sigma)^{+1}$ est en effet démontrée dans [BESV].

Cette dernière généralisation présentée est pour le moment beaucoup moins étudiée que les deux précédentes, et il est fort probable que bien des problématiques soulevées par son étude soient encore à découvrir.

1.5.3 Méthodes générales d'énumération

Beaucoup de résultats d'énumération concernant les classes de permutations ont été obtenus par des preuves élégantes mettant en bijection des permutations à motifs exclus avec d'autres objets de la combinatoire. Parmi ces objets, on peut citer les chemins de Dyck [Eli04], les tableaux de Young [Gui95], les cartes planaires [Gui95], ou encore les chemins de Motzkin, les involutions et les partitions d'ensembles [KM03, et ses références].

Outre leur élégance, l'intérêt des preuves bijectives d'énumération est qu'elles transportent des paramètres d'une classe d'objets à une autre, autorisant des énumérations fines, selon la taille des objets mais aussi la valeur d'un ou de plusieurs paramètres. La thèse de S. Elizalde [Eli04] est très riche d'énumérations de ce type.

Mais lorsque l'on cherche seulement à énumérer les objets d'une classe combinatoire, sans raffinement selon un quelconque paramètre, on pourrait souhaiter disposer d'outils génériques, moins

élégants mais permettant d'obtenir de manière systématique les résultats d'énumération cherchés. Pour les permutations à motifs exclus, quatre méthodes de ce type existent. Elles sont toutes présentées dans [Vat08], et on les décrit brièvement ici.

La technique d'énumération grâce aux *arbres de génération* a été développée par J. West [Wes95], puis reprise et généralisée sous le nom de *méthode ECO* par E. Barcucci, A. Del Lungo, E. Pergola et R. Pinzani dans [BDLPP99]. Appliqué à une classe $\mathcal{C}$ de permutations, le principe de cette méthode est de décrire une règle de production de permutations de $\mathcal{C}_{n+1}$ à partir des permutations de $\mathcal{C}_n$ (plus généralement des objets combinatoires considérés ayant taille $n+1$ à partir de ceux ayant taille n). Cette règle doit permettre de produire tous les éléments de $\mathcal{C}_{n+1}$ à partir de ceux de $\mathcal{C}_n$, sans jamais produire de doublon.

Des étiquettes sont ensuite associées aux permutations, de sorte à pouvoir coder la règle de production trouvée en une règle de réécriture sur les étiquettes. L'arbre de génération est alors l'arbre (*a priori* de profondeur infinie) dont la racine correspond à la permutation 1, et où les fils d'un nœud quelconque de l'arbre, correspondant à une permutation $\sigma \in \mathcal{C}_n$, sont les permutations de $\mathcal{C}_{n+1}$ produites à partir de σ. Les nœuds de l'arbre ne sont pas étiquetés par les permutations elles-même, mais par leur étiquettes utilisées dans la règle de réécriture. Un exemple sera donné au chapitre 6, paragraphe 6.2.2.1. La règle de réécriture ou l'arbre de génération permettent souvent de calculer la séquence d'énumération de la classe de permutations considérée, ou au moins de donner des équations qu'elle satisfait.

La deuxième méthode générique pour obtenir des propriétés énumératives des classes de permutations est associée à la *décomposition par substitution* des permutations. On reviendra en détails au chapitre 2 sur la décomposition par substitution, qui joue un rôle clé dans cette thèse. Pour décrire en quelques mots comment obtenir des résultats d'énumération grâce à la décomposition par substitution des permutations, on peut d'ores et déjà annoncer que les permutations peuvent être représentées par une structure d'arbre, donc naturellement récursive. Grâce à cette structure d'arbre, M. H. Albert et M. D. Atkinson expliquent dans [AA05] comment on peut calculer la série génératrice d'énumération d'une classe de permutations à partir de la série génératrice des permutations *simples* (voir définition page 40) appartenant à la classe. En particulier, lorsqu'une classe contient un nombre fini de permutations simples, leur méthode permet non seulement de calculer effectivement la série génératrice de la classe, mais aussi d'assurer que cette série génératrice est algébrique.

L'*insertion encoding* a été défini par M. H. Albert, S. Linton et N. Ruškuc dans [ALR05] et propose de coder les permutations par des mots et les classes par des langages, et ainsi de disposer de tous les outils de la théorie des langages pour étudier les classes de permutations, en particulier du point de vue énumératif. Le mot associé à une permutation $\sigma \in \mathcal{S}_n$ décrit un chemin d'insertions des éléments $1, 2, \ldots, n$ par ordre croissant, qui aboutit à la permutation σ. Les insertions sont effectuées dans des "trous" (ou *slots*) numérotés de gauche à droite. Les cas les plus favorables dans ce modèle correspondent aux permutations qui peuvent être décrites avec un nombre fini de trous. Dans ce cas, le langage associé à une classe dont la base est finie est un langage régulier, et il est possible, à partir de ce langage, de calculer la série génératrice de la classe.

La dernière méthode "automatique" d'énumération des classes de permutations décrite dans [Vat08] est celle des *enumeration schemes*. Cette méthode a été proposée par D. Zeilberger dans [Zei98], et implémentée dans le package Wilf pour Maple. Cette méthode repose sur la technique du diviser-pour-régner, découpant une classe de permutations en éléments plus petits, et permettant ainsi de trouver des équations de récurrence sur leur séquences d'énumération. Dans [Vat08], V. Vatter a généralisé cette méthode pour qu'elle s'adapte à un plus grand nombre de classes de permutations. Cette généralisation a été implémentée en Maple dans le package WilfPlus.

Concernant l'énumération des classes de permutations selon plusieurs paramètres, des méthodes automatiques de ce type n'ont pas été développées à ce jour. Cependant, des principes généraux

pour des énumérations prenant en compte des paramètres en plus de la taille ont été décrits par M. Bousquet-Mélou. Elle présente dans [BM02] quatre méthodes adaptées à de nombreuses classes de permutations, chacune de ces méthodes étant illustrée par un exemple.

Chapitre 2

Décomposition par substitution et arbres de décomposition des permutations

Sommaire

2.1 Décomposition par substitution et décomposition modulaire des graphes 32
2.1.1 Principes généraux de décomposition par substitution 32
2.1.2 Spécialisation sur les graphes : la décomposition modulaire 33
2.2 Décomposition par substitution des permutations 38
2.2.1 Permutations simples 38
2.2.2 Dilatation et théorèmes de substitution 41
2.2.3 Arbres de décomposition 44
2.3 Point de vue algorithmique : l'arbre des intervalles communs . . . 47
2.3.1 Définition et premières propriétés 47
2.3.2 Lien étroit avec les arbres de décomposition 49
2.3.3 Algorithmes de calcul en temps linéaire 52
2.4 Produit de substitution et classes fermées par produit de substitution 56
2.4.1 Définitions et propriétés 56
2.4.2 Exemples d'utilisation des classes fermées par produit de substitution . 58
2.4.3 Produit de substitution et classes de base finie 59
2.5 Forme en moyenne des arbres de décomposition 59

2.1 Décomposition par substitution et décomposition modulaire des graphes

2.1.1 Principes généraux de décomposition par substitution

La décomposition par substitution est un principe général qui n'est pas associé aux seules permutations, mais qui au contraire a été défini dans un cadre plus large, dont rend compte l'article [MR84] de R. H. Möhring et F. J. Radermacher. Parmi les objets combinatoires dont l'étude a bénéficié des apports de la décomposition par substitution, on peut citer les relations [Fra53, Gal67, MP01, MR84] et les graphes [BLS99, Spi92], mais aussi les tournois [CDM99] ou les ensembles partiellement ordonnés [BM83], parmi d'autres encore [ST93].

L'idée de la décomposition par substitution, développée par R. H. Möhring et F. J. Radermacher dans [MR84], est de permettre de décomposer des objets discrets structurés (appelés plus simplement *structures*) en structures *premières*, dites aussi *simples* ou *indécomposables*. Il faut y voir une volonté de *factoriser* ces structures, comme on peut factoriser un nombre entier en produit de facteurs premiers. Le but de cette décomposition est d'offrir une compréhension approfondie de la structure intrinsèque des objets étudiés.

Les objets (ou structures) étudiés se divisent donc en deux catégories : les structures *premières*, que l'on ne peut pas décomposer, et les structures *décomposables*. Une structure S est décomposable si elle peut être décrite par *substitution*, dans une structure S', d'autres structures plus petites, notées ici S_β. La structure S' est alors appelée *structure quotient* de S. L'opération de substitution de structures dans une autre, notée $S'[S_\beta : \beta \in A']$ est définie pour chaque famille de structures étudiée, mais le principe de substitution reste le même dans tous les cas. La structure S' étant un objet discret, elle est construite sur un ensemble d'éléments A' (les sommets dans un graphe par exemple). La structure $S = S'[S_\beta : \beta \in A']$ est construite à partir de S' en remplaçant chaque élément β de A' par une structure S_β.

R. H. Möhring et F. J. Radermacher ont étudié différentes familles d'objets (relations, graphes, fonctions booléennes, ...) pour démontrer des propriétés du même type dans tous les cas, avec des variantes plus ou moins fortes des résultats selon les objets étudiés. Les relations, et plus encore les graphes (qui ne sont pas autre chose que des relations binaires, symétriques et sans "boucles" (x, x)), sont les objets qui possèdent les propriétés les plus fortes vis-à-vis de la décomposition par substitution.

Parmi les résultats principaux de [MR84], généralisant dans un contexte plus large des travaux antérieurs dont on trouvera les références dans la bibliographie de [MR84], R. H. Möhring et F. J. Radermacher démontrent qu'il existe en toute généralité trois types de structures particulières, à partir desquelles toutes les structures peuvent être construite par substitution : il s'agit des structures premières (de type P), des structures linéaires (L) ou dégénérées (D), et des structures de type infini. Les structures de type infini peuvent être décomposées en sous-structures, qui sont elles-mêmes décomposables, et cette décomposition se propage récursivement dans des sous-structures, sans jamais s'arrêter.

Dans le cas particulier des relations, ils démontrent qu'en outre il n'y a aucune structure de type infini. Ainsi, il suffit des structures de types P, D et L pour construire toutes les relations. Dans le cas encore plus spécifique des graphes (non orientés), il n'y a même que des structures de types P et D qui sont nécessaires (des structures de type L apparaissent dans la décomposition des graphes orientés).

Une autre propriété des relations, et donc des graphes, est que la décomposition des structures en structures de type P, D et L est unique. Dès qu'une famille de structure satisfait cette propriété d'unicité de la factorisation (éventuellement, en autorisant non seulement des structures de type P, D et L, mais aussi des structures de type infini), cela autorise à définir les *arbres de composition*. L'arbre de composition d'une structure S, si elle n'est pas de type infini, est construit récursivement.

Par hypothèse, il existe une unique décomposition de S en $S'[S_\beta : \beta \in A']$ avec la contrainte que S' est de type P, D ou L (comme S n'est pas de type infini, ça n'est pas non plus le cas de S'). L'arbre de composition de S est alors formé d'une racine d'étiquette P, D ou L, selon le type de S', sous laquelle se greffent les arbres de composition des structures S_β pour $\beta \in A'$. Dans le cas où S est de type infini, son arbre de composition est réduit à un nœud racine d'étiquette ∞.

R. H. Möhring et F. J. Radermacher mettent en évidence que les arbres de composition d'une structure décrivent d'une certaine manière toutes les décompositions essentielles de cette structure. Elles sont ainsi stockées dans cet arbre, qui contient seulement un nombre linéaire de nœuds. Möhring et Radermacher soulignent que cette représentation compacte des décompositions d'une structure est bien adaptée à des manipulations algorithmiques. Les aspects algorithmiques sont cependant peu développés dans [MR84], et des travaux postérieurs à la publication de cet article ont largement amélioré les résultats algorithmiques décrits (contrairement aux résultats combinatoires qui sont toujours d'actualité, même si de nombreuses généralisations ont été envisagées depuis).

Les aspects algorithmiques de la décomposition par substitution ont été principalement étudiés dans le cadre des graphes. Le paragraphe 2.1.2 est entièrement consacré à la décomposition par substitution des graphes, appelée *décomposition modulaire* dans ce contexte. On reporte donc au paragraphe 2.1.2 la présentation des résultats algorithmiques principaux relatifs à la décomposition par substitution.

Enfin, on peut remarquer que la décomposition par substitution des permutations n'est pas envisagée dans [MR84]. Il faut attendre les travaux de M. H. Albert et M. D. Atkinson [AA05], exploitant plus avant les idées principales qui transparaissent déjà dans l'article [AS02] de M. D. Atkinson et T. Stitt, pour que cette décomposition des permutations soit définie et utilisée à des fins combinatoires. Pour un résumé très succinct de la décomposition par substitution spécialisée au cas des permutations, on pourra consulter la section 4 de [Vat08]. Nous reviendrons en détails sur la décomposition par substitution des permutations dans le paragraphe 2.2.

2.1.2 Spécialisation sur les graphes : la décomposition modulaire

La spécialisation de la décomposition par substitution qui a été la plus étudiée est certainement la décomposition modulaire des graphes. Elle a été utilisée pour démontrer des résultats combinatoires majeurs, et a été mise à profit pour concevoir des algorithmes efficaces. Après les définitions nécessaires, ce paragraphe résume quelques uns de ces résultats. Pour un panorama plus large, on pourra consulter par exemple [Pau, Hab, HP].

Un avertissement au lecteur s'impose ici : cette thèse ne prétend en aucune façon traiter de la décomposition modulaire des graphes. Ce paragraphe a pour but d'en présenter seulement quelques aspects, qui permettront de souligner, dans la suite du manuscrit, les points communs mais aussi les différences entre décomposition modulaire des graphes et décomposition par substitution des permutations.

Vocabulaire de base de théorie des graphes

Définition 2.1. Un *graphe* $G = (V, E)$ est défini par un ensemble fini de sommets V et un ensemble E d'arêtes, qui relient chacune deux sommets distincts.

On considèrera des graphes non orientés, c'est-à-dire que les arêtes sont des *paires* $\{x, y\}$ de sommets distincts. Les graphes *orientés* sont définis en considérant au contraire des *couples* (x, y) de sommets distincts pour définir les arêtes orientées.

Remarquons que les graphes considérés sont *simples* (il n'y a pas d'arêtes multiples entre deux sommets) et *sans boucles* (il n'y a pas d'arête reliant un sommet à lui-même).

Une paire $\{x, y\}$ de sommets distincts qui n'est pas une arête est appelée une *non arête*.

Définition 2.2. Deux sommets reliés par une arête sont dits *adjacents*, et les sommets adjacents à un sommet x sont les *voisins* de x. L'ensemble des voisins de x s'appelle aussi voisinage de x et est noté $N(x)$.

Deux sommets x et y sont *connectés* s'il existe une suite de sommets $x = v_0, v_1, \ldots, v_k = y$ telle que pour tout $i \in [0..(k-1)]$, v_i et v_{i+1} sont adjacents. En particulier, pour $k = 0$, x est connecté à lui-même.

Un graphe est *connexe* si pour tous sommets x et y, ceux-ci sont connectés. La relation de *connexité* est la relation binaire symétrique composée de toutes les paires de sommets connectés. C'est une relation d'équivalence, et les classes d'équivalence pour cette relation sont appelées *composantes connexes* du graphe sous-jacent.

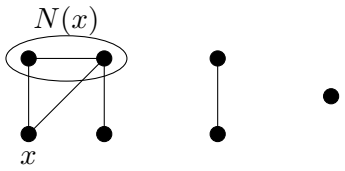


FIG. 2.1 – Un graphe G possédant trois composantes connexes, et où le sommet x a deux voisins.

Définition 2.3. Pour tout graphe $G = (V, E)$, on peut définir son graphe *complémentaire* $\overline{G}$ en transformant les arêtes de G en non arêtes et vice-versa. Plus formellement, en notant abusivement V^2 l'ensemble des paires (et non des couples) d'éléments de V, le complémentaire de G est $\overline{G} = (V, \overline{E})$, avec $\overline{E} = V^2 \setminus \big(E \cup \{\{x, x\} : x \in V\}\big)$.

Définition 2.4. Dans un graphe $G = (V, E)$, le *sous-graphe induit* par un ensemble de sommets $S \subseteq V$ est le graphe d'ensemble de sommets S, où deux sommets sont adjacents si et seulement s'ils étaient adjacents dans G.

Parmi tous les graphes, on distingue deux familles particulières, dont l'ensemble des arêtes est défini le plus simplement possible. On verra que ces graphes correspondent en fait aux structures dégénérées de R. H. Möhring et F. J. Radermacher.

Définition 2.5. Le *stable* à n sommets est le graphe $G = (V, E)$, où V est un ensemble de taille n et $E = \emptyset$. La *clique* à n sommets est le graphe $G = (V, E)$, où V est un ensemble de taille n et $E = V^2 \setminus \{\{x, x\} : x \in V\}$.

Ici encore V^2 désigne abusivement l'ensemble des paires de sommets de V.

On remarque que cliques et stables sont des graphes complémentaires.

Modules et décomposition modulaire

Définition 2.6. Un *module* dans un graphe $G = (V, E)$ est un ensemble $M \subseteq V$ de sommets de G qui sont indistinguables par les sommets extérieurs. En d'autres termes, pour tout sommet $x \in V \setminus M$, $\begin{cases} \text{soit } M \subseteq N(x) \text{ (tous les sommets de } M \text{ sont adjacents à } x), \\ \text{soit } M \cap N(x) = \emptyset \text{ (aucun sommet de } M \text{ n'est adjacent à } x). \end{cases}$

Les sommets d'un module jouent donc tous le même rôle vis-à-vis des sommets extérieurs au module. On pourra ainsi les contracter en un seul sommet sans bouleverser la structure profonde du graphe. Il est aussi important de remarquer que par définition, les modules d'un graphe G et de son complémentaire $\overline{G}$ sont les mêmes.

Dans tout graphe G, les ensembles V et $\{x\}$ pour tout $x \in V$ sont des modules. Ces modules sont appelés les modules *triviaux*.

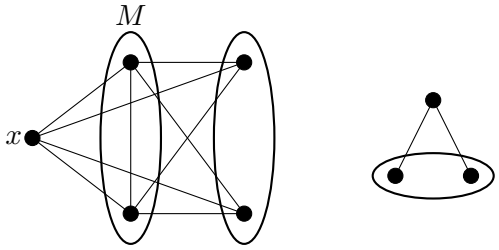


FIG. 2.2 – Exemples de modules dans un graphe. Chacun des trois ensembles de sommets entourés forme un module de même que l'ensemble $M \cup \{x\}$ par exemple.

Définition 2.7. Un graphe G est un graphe *premier* si tous ses modules sont des modules triviaux.

Il n'y a pas de graphe premier à 3 sommets. On considère en général que les graphes à 1 ou 2 sommets ne sont pas premiers (bien qu'ils satisfassent la définition 2.7). Parmi les graphes à 4 sommets, il n'y a qu'un graphe premier : c'est le chemin à 4 sommets •—•—•—•, noté en général P_4, et qui est donc le plus petit graphe premier. Comme la terminologie pouvait le laisser supposer, les graphes premiers correspondent aux structures premières de R. H. Möhring et F. J. Radermacher. Les graphes premiers possèdent de fortes propriétés structurelles. On en donne seulement une ici, démontrée dans le cadre plus général des structures premières par J. H. Schmerl et W. T. Trotter [ST93] :

Théorème 2.8. *Tout graphe premier G à $n \geq 5$ sommets contient un graphe premier à $n-1$ ou $n-2$ sommets, c'est-à-dire qu'il existe un sous-graphe induit de G qui a $n-1$ ou $n-2$ sommets et qui est premier.*

Parmi tous les modules d'un graphe, certains jouent un rôle plus crucial que les autres. Il s'agit des modules forts.

Définition 2.9. Deux modules M et M' d'un graphe $G = (V, E)$ se chevauchent si $M \cap M' \neq \emptyset$, sans qu'on ait pour autant $M \subseteq M'$ ou $M' \subseteq M$.
Un module *fort* de G est un module qui ne chevauche aucun autre module.
Un module fort d'un graphe G est *maximal* lorsque le seul module fort de G qui le contient est V.

Définition 2.10. Une *partition modulaire* de $G = (V, E)$ est une partition des sommets de V telle que chaque partie soit un module de G. Étant donnée une partition modulaire $\mathcal{P}$ de G, le *graphe quotient* $G_{/\mathcal{P}}$ est obtenu en contractant chaque partie de $\mathcal{P}$ en un seul sommet.

Les parties de $\mathcal{P}$ étant des modules, pour deux parties M et M' de $\mathcal{P}$, on est dans l'une des deux situations suivantes : $\begin{cases} \text{soit } \forall x \in M, \forall x' \in M', x \text{ est adjacent à } x', \\ \text{soit } \forall x \in M, \forall x' \in M', x \text{ n'est pas adjacent à } x'. \end{cases}$
Ceci définit sans ambiguïté les arêtes de $G_{/\mathcal{P}}$: des sommets correspondant à deux modules M et M' sont reliés si et seulement si on est dans le premier des deux cas ci-dessus.

On dispose à présent de toute la terminologie nécessaire à énoncer le théorème de décomposition modulaire des graphes :

Théorème 2.11. *Soit un graphe G. Notons $\mathcal{M}(G)$ la partition modulaire de G en modules forts maximaux. Alors une et une seule des trois propositions suivantes est satisfaite :*
- *le graphe G n'est pas connexe, c'est-à-dire que $G_{/\mathcal{M}(G)}$ est un stable (cas* parallèle*),*
- *le complémentaire $\overline{G}$ du graphe G n'est pas connexe, c'est-à-dire que $G_{/\mathcal{M}(G)}$ est une clique (cas* série*),*

– *le graphe $G_{/\mathcal{M}(G)}$ est un graphe premier (donc de taille au moins 4) (cas* premier*).*

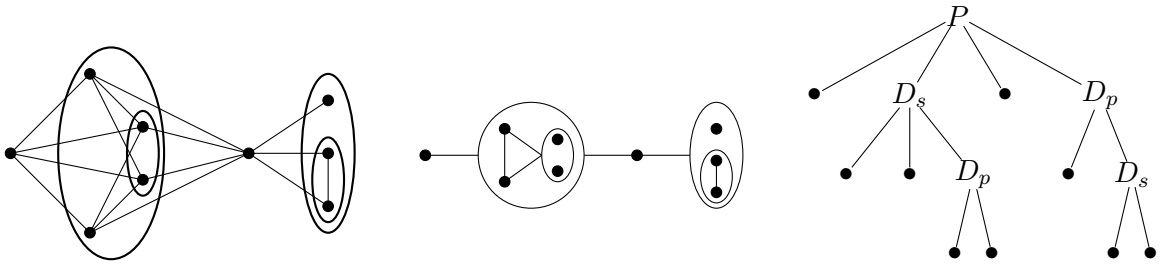
Ce théorème est une spécialisation de la décomposition par substitution de R. H. Möhring et F. J. Radermacher au cadre des graphes. Les cas parallèle et série correspondent aux structures dégénérées (de type D), et le cas premier aux structures premières (de type P). Comme annoncé, la décomposition modulaire d'un graphe est unique, et ne fait pas intervenir de structures linéaires (de type L).

Une fois obtenu ce théorème de décomposition, il est naturel de suivre le cheminement de R. H. Möhring et F. J. Radermacher et de décomposer récursivement les graphes induits par chaque partie de $\mathcal{M}(G)$, pour construire l'*arbre de décomposition modulaire*, qui est la version des arbres de composition restreinte aux graphes.

Arbre de décomposition modulaire, et algorithmes de calcul de cet arbre

Définition 2.12. L'*arbre de décomposition modulaire* d'un graphe $G = (V, E)$ est l'arbre tel que
- la racine correspond à V,
- les feuilles correspondent aux sommets de G,
- chaque nœud de l'arbre correspond à un module fort de G,
- un nœud correspondant à un module M' est le fils d'un nœud correspondant à un module M si et seulement si M est le plus petit module fort qui contient M' strictement.

D'après le théorème 2.11, les nœuds de l'arbre de décomposition modulaire de G sont de trois types : parallèle, série et premier.



Un graphe G et ses modules forts. | Décomposition récursive de G en modules forts. | Arbre de décomposition modulaire de G.

FIG. 2.3 – Décomposition modulaire d'un graphe et arbre de décomposition modulaire. Les nœuds premiers sont étiquetés P, et les nœuds dégénérés séries et parallèles D_s et D_p respectivement.

Remarque 2.13. On remarque au passage que les arbres de décomposition modulaire ne sont pas des arbres plans, c'est-à-dire que les fils d'un nœud ne sont pas ordonnés. Cela se comprend très facilement pour les nœuds dégénérés, puisque tous les sommets d'un stable (ou d'une clique) jouent le même rôle. On verra qu'au contraire les arbres de décomposition des permutations sont plans (les fils de tout nœud interne sont totalement ordonnés), ce qui en facilite grandement l'étude combinatoire.

Il existe différents algorithmes pour calculer l'arbre de décomposition modulaire d'un graphe, qui reposent sur différentes approches. Un aperçu de la variété des techniques utilisées est donné dans [Hab], où sont décrites les idées principales des différentes familles d'algorithmes. Dans [Pau], C. Paul présente en détails et de façon unifiée les deux familles d'algorithmes qui fournissent des moyens de calculer l'arbre de décomposition d'un graphe en temps linéaire.

La première famille d'algorithmes présentée repose sur un principe développé par A. Ehrenfeucht, H. N. Gabow, R. M. McConnell et S. J. Sullivan en 1994 : il s'agit de calculer un chemin de

la racine à une feuille dans l'arbre, puis de calculer récursivement les sous-arbres qui se branchent sur cette colonne vertébrale. La première étape revient à calculer l'arbre de décomposition modulaire dans un graphe où tous les modules forts sont imbriqués les uns dans les autres. C'est cette étape qui concentre la principale difficulté du problème. Elle est résolue en temps quadratique dans l'algorithme d'origine de Ehrenfeucht *et al.*, dont la complexité totale est quadratique.

Une manière de construire des algorithmes sous-quadratiques, et même linéaires, à partir de l'idée de A. Ehrenfeucht *et al.* est d'utiliser des structures de données avancées, qui permettent d'obtenir des algorithmes plus complexes mais de complexité plus faible pour résoudre la première étape. Cette amélioration est due à E. Dahlhaus, J. Gustedt et R. M. McConnell en 2001. Une autre méthode donnant une solution sous-quadratique à la première étape est l'utilisation d'affinage de partition ordonnée. Cette idée a été exploitée par R. M. McConnell et J. Spinrad en 2000, pour obtenir une version sous-quadratique (mais sur-linéaire) de l'algorithme de A. Ehrenfeucht *et al.*

La seconde famille d'algorithmes permettant d'aboutir à un calcul en temps linéaire de l'arbre de décomposition modulaire est celle à base de *permutations factorisantes*. Les algorithmes de cette famille sont construits en deux étapes : d'abord le calcul d'une permutation factorisante du graphe [TCHP08] (faisant suite à [HMP04]), puis le calcul de l'arbre de décomposition modulaire à partir de cette permutation factorisante [CHM02].

Pour la première étape, M. Habib, C. Paul et L. Viennot ont proposé un algorithme sous-quadratique en 1998-99, à partir d'une technique d'affinage de partition. Cet algorithme a été amélioré par M. Habib, F. de Montgolfier et C. Paul en 2004, puis par M. Tedder, D. G. Corneil, M. Habib et C. Paul en 2008, jusqu'à obtenir une complexité linéaire, mais au prix d'une grande complication de l'algorithme.

La deuxième étape consiste à calculer l'arbre de décomposition modulaire étant donnée une permutation factorisante. Un premier algorithme linéaire pour résoudre ce problème est dû à C. Capelle, M. Habib et F. de Montgolfier en 2002. Il est important de remarquer que le calcul de l'arbre de décomposition d'un graphe dont est donnée une permutation factorisante est très similaire au calcul de l'arbre des intervalles communs des permutations. À partir de cette remarque, deux équipes – d'une part B.-M. Bui Xuan, M. Habib et C. Paul [BXHP05], et d'autre part A. Bergeron, C. Chauve, F. de Montgolfier et M. Raffinot [BCMR05, BCMR08]) – ont adapté en 2005 un algorithme de T. Uno et M. Yagiura [UY00] calculant les intervalles communs à un ensemble de permutations, afin de décrire d'autres algorithmes linéaires de calcul de l'arbre de décomposition modulaire d'un graphe étant donnée une permutation factorisante. Nous reviendrons plus longuement au paragraphe 2.3 sur les arbres des intervalles communs des permutations.

Du point de vue de la complexité, il faut donc retenir que :

Théorème 2.14. *Il existe des algorithmes linéaires calculant, étant donné un graphe G, son arbre de décomposition modulaire.*

Il faut cependant pondérer ce résultat par le fait que les algorithmes de complexité linéaire connus utilisent des concepts difficiles (même aux dires des spécialistes), et demandent une analyse de complexité très fine pour prouver leur linéarité. La description d'un algorithme simple et linéaire pour la décomposition modulaire des graphes reste à ce jour un problème ouvert.

Quelques exemples d'application de la décomposition modulaire

Les succès de la décomposition modulaire des graphes, aussi bien d'un point de vue algorithmique que combinatoire, ne sont plus à démontrer (voir par exemple [BLS99, CH94, Ill97, Spi92, Sum71]).

Le théorème des mineurs de Robertson et Seymour, qui est très certainement l'un des (si ce n'est le) résultat(s) majeur(s) de combinatoire des graphes de ces dernières années, a par exemple été obtenu en utilisant une décomposition arborescente des graphes, même s'il ne s'agit pas exactement

de la décomposition modulaire. Une conséquence importante de ce théorème est que toute classe de graphes fermée par le bas pour la relation de mineur est caractérisée par un nombre fini de mineurs exclus. On peut comparer ce résultat avec ce qui se produit dans le cadre des classes de permutations, où l'on a vu que certaines classes pouvaient avoir une base infinie de motifs exclus.

Toujours d'un point de vue combinatoire, la décomposition modulaire est un outil qui permet de mieux comprendre la structure de certaines classes de graphes. C'est le cas par exemple pour les graphes de comparabilité [Gal67]. Elle offre aussi un codage compact des graphes, si on garde le graphe quotient dans tout nœud premier de l'arbre de décomposition modulaire.

En algorithmique, la décomposition modulaire est un bon point de départ pour appliquer des stratégies du type diviser pour régner, et peut donc être utilisée pour concevoir des algorithmes efficaces. Elle permet aussi d'avoir une meilleure compréhension des algorithmes existants, et des structures de données qu'ils mettent en jeu.

2.2 Décomposition par substitution des permutations

Bien qu'elles soient absentes du travail fondateur de R. H. Möhring et F. J. Radermacher [MR84], les permutations peuvent être étudiées à travers le prisme de la décomposition par substitution. Les récentes et nombreuses publications consacrées aux permutations simples [AA05, AAK03, Brib, BHV08a, BHV08b, BRV08, CLP06], qui correspondent aux structures premières de [MR84], témoignent d'un intérêt nouveau pour cette manière d'appréhender l'étude des classes de permutations.

Comme dans le cadre général, les permutations simples sont envisagées comme les "briques de base" à partir desquelles toutes les permutations sont construites, au moyen de la *substitution*, aussi appelée *dilatation* (en anglais *inflation*) dans ce contexte. C'est ce que formalisent les théorèmes de *décomposition par substitution*.

C'est d'abord dans le champ algorithmique qu'on devine la décomposition par substitution des permutations, de manière implicite dans les travaux de S. Heber et J. Stoye en 2001 [HS01]. En combinatoire, les prémisses de cette décomposition des permutations figurent dans les travaux de M. D. Atkinson et T. Stitt en 2002 [AS02], mais les arguments ne sont pas poussé assez loin pour faire apparaitre les permutations simples comme clé de décomposition. Trois ans plus tard, M. H. Albert et M. D. Atkinson [AA05] posent les bases de la décomposition par substitution pour les permutations, et démontrent les premiers résultats combinatoires sur les classes de permutations utilisant cet outil.

2.2.1 Permutations simples

Les permutations simples et les intervalles dans les permutations (en particulier les intervalles forts) sont deux concepts essentiels pour la décomposition par substitution des permutations. Outre les deux articles [AAK03] et [AA05] auxquels on fera référence, le lecteur pourra consulter l'article de survol de R. Brignall [Brib] pour un panorama plus complet des problématiques combinatoires (et dans une moindre mesure algorithmiques) envisagées autour des permutations simples.

Les intervalles des permutations

Définition 2.15. Un *intervalle* ou *bloc* dans une permutation σ est un ensemble d'éléments de σ dont les positions et les valeurs forment deux intervalles de $\mathbb{N}$. Un intervalle sera identifié par les valeurs de ses éléments.

On utilisera essentiellement la terminologie d'*intervalles* en association avec la représentation en une ligne des permutations, et on parlera plutôt de *blocs* lorsqu'on travaille avec leur représentation graphique.

Exemple 2.16. Les intervalles de la permutation $\sigma = 4\,3\,1\,2\,6\,5\,8\,7$ sont $\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{1,2\}, \{3,4\}, \{5,6\}, \{7,8\}, \{1,2,3\}, \{1,2,3,4\}, \{5,6,7,8\}, \{1,2,3,4,5,6\}$ et $\{1,2,3,4,5,6,7,8\}$. La figure 2.4 (figure de gauche) permet de visualiser ces intervalles sur la représentation graphique de σ.

Remarque 2.17. La partie droite de la figure 2.4 met en évidence sur un exemple qu'un bloc sur la représentation graphique d'une permutation est un carré qui n'admet aucun point sur ses côtés, c'est-à-dire tel qu'aucun point de la permutation ne se trouve au-dessus, en-dessous, à gauche ou à droite de ce carré. D'après la définition 2.15, cette propriété est générale, et caractérise les blocs d'une permutation.

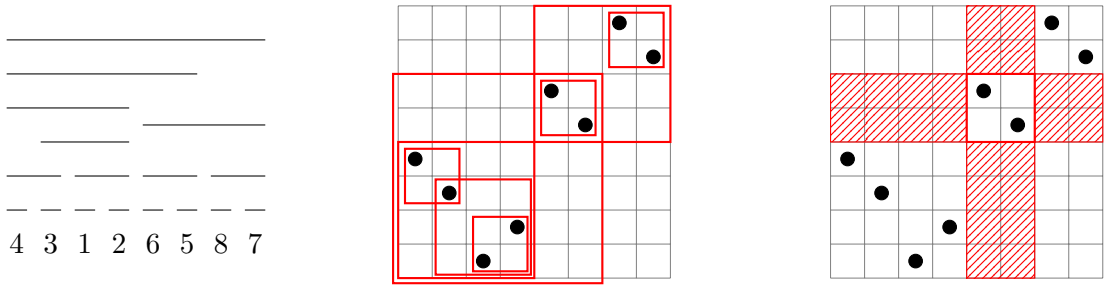


FIG. 2.4 – À gauche, les intervalles d'une permutation, sur sa représentation en une ligne. Au centre, ces mêmes blocs sur sa représentation graphique. À droite, un bloc est un carré dans la représentation graphique qui n'admet aucun point de la permutation sur ses côtés.

Parmi les intervalles d'une permutation, on peut en distinguer certains, qui satisfont des propriétés supplémentaires :

Définition 2.18. Un intervalle I dans une permutation est dit *minimal* lorsque I n'est pas un singleton mais que tout intervalle strictement inclus dans I est un singleton.

Définition 2.19. Deux intervalles I et J d'une permutation σ se chevauchent si $I \cap J \neq \emptyset$, sans qu'on ait pour autant $I \subseteq J$ ou $J \subseteq I$.
Un intervalle d'une permutation σ est dit *fort* lorsqu'il ne chevauche aucun intervalle de σ.

Définition 2.20. Un intervalle fort d'une permutation $\sigma \in \mathcal{S}_n$ est dit *maximal* lorsque le seul intervalle fort de σ qui le contient est $[1..n]$.

Exemple 2.21. Sur la permutation $\sigma = 4\,3\,1\,2\,6\,5\,8\,7$ de l'exemple 2.16, les intervalles $\{3,4\}$ et $\{1,2,3\}$ se chevauchent, de même que les intervalles $\{5,6,7,8\}$ et $\{1,2,3,4,5,6\}$. On obtient ainsi que les intervalles forts de σ sont $\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{1,2\}, \{5,6\}, \{7,8\}, \{1,2,3,4\}$ et $\{1,2,3,4,5,6,7,8\}$, et ses intervalles forts maximaux sont $\{5,6\}, \{7,8\}$ et $\{1,2,3,4\}$. Les intervalles minimaux de σ sont ici tous de taille 2 ($\{1,2\}, \{3,4\}, \{5,6\}$ et $\{7,8\}$), mais ça n'est pas toujours le cas. Les intervalles minimaux de la permutation $4\,3\,1\,2\,6\,8\,5\,7$, obtenue en échangeant 5 et 8 dans σ, sont $\{1,2\}, \{3,4\}$ et $\{5,6,7,8\}$.

On verra que les intervalles minimaux jouent un rôle particulier pour démontrer certaines propriétés énumératives (voir le lemme 2.71). Quant aux intervalles forts, ils permettent de définir la structure d'*arbre des intervalles communs* sur les permutations, qui représente dans un espace linéaire l'ensemble des intervalles d'une permutation (qui peut contenir jusqu'à un nombre quadratique d'intervalles). Le paragraphe 2.3 est consacré à la définition et à quelques propriétés algorithmiques de l'arbre des intervalles communs.

On mentionne ici un travail concernant les intervalles dans les permutations, sans connexion apparente avec leur décomposition par substitution. Motivé par des applications en génomique, S. Corteel, G. Louchard et R. Pemantle s'intéressent dans [CLP06] à la distribution du nombre d'intervalles dans une permutation aléatoire.

Définition des permutations simples

Certains intervalles apparaissent dans toutes les permutations. Pour une permutation de taille n, il s'agit des singletons $\{1\}, \{2\}, \ldots, \{n\}$ et de l'intervalle $[1..n]$. Ces intervalles s'appellent les intervalles *triviaux*.

Définition 2.22. Un intervalle *trivial* d'une permutation $\sigma \in \mathcal{S}_n$ est soit $[1..n]$ soit un singleton $\{i\}$, avec $1 \leq i \leq n$.

Définition 2.23. Les permutations *simples* sont les permutations dont tous les intervalles sont triviaux. On exclura cependant de l'ensemble des permutations simples les permutations 1, 1 2 et 2 1.

Exemple 2.24. La permutation 3 5 1 6 2 4 est simple. En revanche, cela n'est pas le cas de la permutation 3 5 1 2 6 4, qui contient l'intervalle $\{1, 2\}$. Les représentations graphiques des ces permutations sont données en figure 2.5.

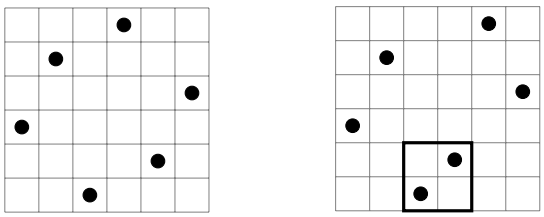


FIG. 2.5 – Une permutation simple (à gauche), et une permutation contenant un intervalle de taille 2 (à droite).

De même que 1 n'est pas un nombre premier dans le contexte de la factorisation des entiers, dans une optique de décomposition de permutations, il est naturel de ne pas considérer la permutation 1 parmi les permutations simples. Pour les permutations de taille 2, cela est discutable. Pour le travail présenté dans cette thèse, il est plus adapté d'exclure 1 2 et 2 1 de l'ensemble des permutations simples. On verra une première illustration de ce choix dans les théorèmes 2.33 et 2.35 de décomposition des permutations.

Remarque 2.25. On vérifie immédiatement que chacune des six permutations de taille 3 contient un intervalle de taille 2. Ainsi, une permutation simple a taille supérieure ou égale à 4. En outre, il y a deux permutations simples de taille 4, à savoir 2 4 1 3 et 3 1 4 2.

Quelques propriétés des permutations simples

Dans [AAK03], M. H. Albert, M. D. Atkinson et M. Klazar s'intéressent à l'énumération des permutations simples. En particulier, ils étudient la relation entre la série génératrice des permutations simples et celle des nombres de Comtet. Cela leur permet de démontrer que la séquence d'énumération des permutations simples n'est pas P-récursive (ou que la série génératrice associée n'est pas D-finie), c'est-à-dire qu'elle n'est pas solution d'une récurrence linéaire à coefficients polynomiaux. En pratique, cela implique qu'il est très improbable (sinon impossible) de trouver

une formule close donnant le nombre de permutations simples de taille n. Malgré ces obstacles théoriques à leur énumération, M. H. Albert, M. D. Atkinson et M. Klazar donnent un équivalent asymptotique assez fin du nombre de permutations simples :

Théorème 2.26. *Un équivalent asymptotique du nombre s_n de permutations simples de taille n est*

$$s_n = \frac{n!}{e^2}(1 - \frac{4}{n} + \frac{2}{n(n-1)} + \mathcal{O}(\frac{1}{n^3})) \text{ quand } n \to \infty.$$

La méthode employée pour démontrer le théorème 2.26 permet (en théorie et au prix de lourds calculs) d'obtenir autant de termes que souhaité dans le développement asymptotique de s_n.

Les permutations simples comme les graphes premiers correspondent aux structures premières de R. H. Möhring et F. J. Radermacher. De même qu'on avait le théorème 2.8 dans le cadre des graphes, les travaux de J. H. Schmerl et W. T. Trotter assurent que :

Théorème 2.27 ([ST93], repris dans [AA05])**.** *Toute permutation simple de taille $n \geq 5$ contient, en tant que motif, une permutation simple de taille $n-1$ ou $n-2$.*

En d'autres termes, dans toute permutation simple de taille au moins 5, il existe un ou deux éléments que l'on peut supprimer de sorte à retrouver une permutation simple.

En outre, les permutations simples pour lesquelles il est nécessaire de supprimer deux éléments sont caractérisées dans [AA05], où elles prennent le nom de permutations *exceptionnelles*.

2.2.2 Dilatation et théorèmes de substitution

Afin d'énoncer, dans le cadre des permutations, les théorèmes de substitution au sens de R. H. Möhring et F. J. Radermacher, il faut définir la notion de *substitution* de permutations $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$ dans une permutation $\sigma \in \mathcal{S}_k$, aussi appelée *dilatation* de σ par les permutations $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$.

Si σ désigne une permutation de $\mathcal{S}_k$ et α une permutation de $\mathcal{S}_p$ alors la dilatation du i-ème élément de σ par α (ou la substitution de α dans σ à la position i) produit la permutation $\pi = \overline{\sigma_1}\,\overline{\sigma_2}\ldots\overline{\sigma_{i-1}}\,(\alpha_1 + \sigma_i - 1)\ldots(\alpha_p + \sigma_i - 1)\,\overline{\sigma_{i+1}}\ldots\overline{\sigma_{k+p-1}}$ où $\overline{\sigma_j} = \begin{cases} \sigma_j \text{ si } \sigma_j < \sigma_i, \\ \sigma_j + p - 1 \text{ sinon.} \end{cases}$

Exemple 2.28. La substitution de $\alpha = 3\,1\,2\,4$ dans $\sigma = 2\,5\,4\,6\,7\,1\,3$ à la position 3 (c'est-à-dire la dilatation de l'élément $\sigma_3 = 4$) donne la permutation $\pi = 2\,8\,\mathbf{6\,4\,5\,7}\,9\,10\,1\,3$.

On notera l'opération de substitution/dilatation ainsi définie par $\sigma[1, \ldots, 1, \underbrace{\alpha}_{i}, 1, \ldots, 1]$.

Sur l'exemple précédent, $\pi = 2\,5\,4\,6\,7\,1\,3[1, 1, 3\,1\,2\,4, 1, 1, 1, 1]$. Cette notation se généralise naturellement à $\sigma[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$, pour permettre de dilater plusieurs éléments de manière simultanée.

Définition 2.29. La dilatation d'une permutation σ de taille k par k permutations $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$ de tailles respectives $p_1, p_2, \ldots, p_k$ (encore appelée substitution de $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$ dans σ) est la permutation $\pi = \sigma[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$ définie par :

$$\pi = \text{shift}(\alpha^{(1)}, \sigma_1)\ldots\text{shift}(\alpha^{(k)}, \sigma_k) \text{ où}$$

$$\text{shift}(\alpha^{(i)}, \sigma_i) = \text{shift}(\alpha^{(i)}, \sigma_i)(1)\ldots\text{shift}(\alpha^{(i)}, \sigma_i)(p_i) \text{ et}$$

$$\text{shift}(\alpha^{(i)}, \sigma_i)(x) = (\alpha^{(i)}(x) + p_{\sigma^{-1}(1)} + \ldots + p_{\sigma^{-1}(\sigma_i - 1)}) \text{ pour tout } x \text{ entre 1 et } p_i.$$

Exemple 2.30. La dilatation de la permutation $\sigma = 1\,3\,2$ par les permutations $\alpha^{(1)} = 2\,1$, $\alpha^{(2)} = 1\,3\,2$ et $\alpha^{(3)} = 1$ donne comme résultat la permutation $\pi = 1\,3\,2[2\,1, 1\,3\,2, 1] = 2\,1 \vdots 4\,6\,5 \vdots 3$, les symboles $\vdots$ indiquant la limite entre les facteurs shift$(\alpha^{(i)}, \sigma_i)$ définissant π.

Remarque 2.31. Dans certaines situations particulières, on pourra envisager de dilater un élément σ_i par la permutation ϵ, de taille 0. Cela aura pour effet d'effacer l'élément σ_i. On utilisera en particulier la dilatation par un motif vide dans les algorithmes de recherche de plus grand motif commun aux chapitres 4 et 5. Dans les théorèmes de décomposition qui suivent, et de manière générale dans tout le travail combinatoire sur la décomposition par substitution, on exclut cependant la possibilité de dilater un élément par le motif vide.

L'opération de dilatation de $\sigma \in \mathcal{S}_k$ par des permutations $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$ a été définie par M. H. Albert et M. D. Atkinson dans [AA05], sous le nom anglais de *inflation*. Elle se visualise très bien sur la représentation graphique des permutations : la représentation graphique de $\sigma[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$ est obtenue à partir de celle de σ en remplaçant chaque point σ_i par un bloc contenant la représentation graphique de $\alpha^{(i)}$. Un exemple est donné en figure 2.6.

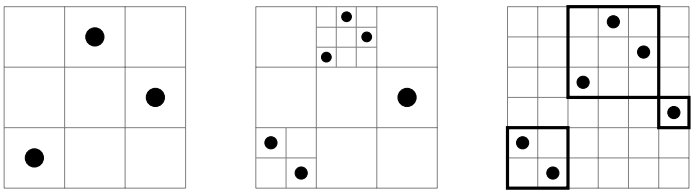


FIG. 2.6 – Dilatation de la permutation $\sigma = 1\,3\,2$ par les permutations $\alpha^{(1)} = 2\,1$, $\alpha^{(2)} = 1\,3\,2$ et $\alpha^{(3)} = 1$, à travers la représentation graphique de ces permutations. Le résultat obtenu est la permutation $\pi = 1\,3\,2[2\,1, 1\,3\,2, 1] = 2\,1\,4\,6\,5\,3$.

Remarque 2.32. On s'aperçoit facilement (par exemple sur la représentation graphique des permutations) que les permutations $\alpha^{(i)}$ dans une dilatation de la forme $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$ correspondent à des blocs (c'est-à-dire des intervalles) de σ : conformément à la remarque 2.17, les éléments de σ correspondant à un $\alpha^{(i)}$ forment un carré sans aucun point sur ses côtés.

En outre, tout bloc de σ, s'il n'est pas égal à σ, est inclus dans l'un des $\alpha^{(i)}$.

Dans [AA05], M. H. Albert et M. D. Atkinson donnent le premier théorème de décomposition spécifique aux permutations. Dans leur travail, les permutations $1\,2$ et $2\,1$ sont considérées comme des permutations simples. Avec cette convention (locale au théorème 2.33), leur résultat s'énonce de la manière suivante :

Théorème 2.33. *Pour toute permutation σ de taille au moins 2, il existe une unique permutation simple π et des permutations $\alpha^{(i)}$ uniques elles aussi, telles que $\sigma = \pi[\alpha^{(1)}, \ldots, \alpha^{(k)}]$.*
Si $\pi = 1\,2$ (resp. $2\,1$), pour avoir l'unicité des $\alpha^{(i)}$, il faut en plus imposer que $\alpha^{(1)}$ soit $\oplus$-indécomposable *(resp.* $\ominus$-indécomposable*).*

Une permutation est dite $\oplus$-*indécomposable* (resp. $\ominus$-*indécomposable*) lorsqu'elle ne peut pas être décomposée sous la forme $1\,2[\alpha^{(1)}, \alpha^{(2)}]$ (resp. $2\,1[\alpha^{(1)}, \alpha^{(2)}]$). On remarque en passant que la définition des permutations $\oplus$- et $\ominus$-indécomposables (sous le vocable *forward-* et *backward-indecomposable*) apparaît pour la première fois dans l'article [AN81] de D. Avis et N. Newborn, plus de 20 ans avant les travaux sur la décomposition par substitution des permutations.

Définition 2.34. Dans tous les cas, la permutation π du théorème 2.33 est unique. Elle est appelée *squelette* de σ.

Le théorème 2.33 introduit une dissymétrie entre les permutations $\alpha^{(1)}$ et $\alpha^{(2)}$ lorsque le squelette de la permutation décomposée est 1 2 ou 2 1. Une autre manière de formuler le théorème ci-dessus, sans introduire cette dissymétrie est proposée dans le théorème 2.35. Pour énoncer ce théorème, et dans toute la suite de la thèse, on exclut 1 2 et 2 1 de l'ensemble des permutations simples. On verra dans le paragraphe 2.3 que ce théorème a une saveur de décomposition en intervalles communs des permutations.

Théorème 2.35. *Toute permutation σ de taille au moins 2 se décompose de manière unique sous l'une des trois formes suivantes :*

$$\sigma = \begin{cases} 12\dots k[\alpha^{(1)},\dots,\alpha^{(k)}], \text{ où les } \alpha^{(i)} \text{ sont tous } \oplus\text{-indécomposables,} \\ k\dots 21[\alpha^{(1)},\dots,\alpha^{(k)}], \text{ où les } \alpha^{(i)} \text{ sont tous } \ominus\text{-indécomposables,} \\ \pi[\alpha^{(1)},\dots,\alpha^{(k)}], \text{ où } \pi \text{ est une permutation simple (donc de taille au moins 4).} \end{cases}$$

Définition 2.36. Dans le contexte du théorème 2.35, on définit le *squelette* de la permutation σ par $12\dots k$ dans le premier cas, $k\dots 21$ dans le deuxième cas, et π dans le troisième cas.

Les squelettes $12\dots k$ et $k\dots 21$ pourront être notés respectivement $\oplus$ et $\ominus$, pour tout $k \geq 2$. La valeur de l'entier k est déterminée par le nombre de permutations $\alpha^{(i)}$ qui dilatent tout symbole $\oplus$ ou $\ominus$.

Avec cette notation, on peut reformuler la définition des permutations $\oplus$-indécomposables (resp. $\ominus$-indécomposables) de la manière suivante : ce sont celles qui ne sont pas exprimables comme dilatation d'un symbole $\oplus$ (resp. $\ominus$).

Démonstration du théorème 2.35. Le théorème 2.35 est une reformulation du théorème 2.33.

Lorsque σ se décompose sous la forme $1\,2[\alpha^{(1)},\alpha^{(2)}]$, $\alpha^{(1)}$ est déjà $\oplus$-indécomposable. Et si ça n'est pas le cas pour $\alpha^{(2)}$, alors $\alpha^{(2)}$ se décompose en $1\,2[\alpha^{(3)},\alpha^{(4)}]$, avec $\alpha^{(3)}$ $\oplus$-indécomposable. Ainsi, $\sigma = 1\,2\,3[\alpha^{(1)},\alpha^{(3)},\alpha^{(4)}]$, où $\alpha^{(1)}$ et $\alpha^{(3)}$ sont $\oplus$-indécomposables.

On poursuit ainsi récursivement à l'intérieur de $\alpha^{(4)}$ jusqu'à arriver à une permutation qui n'est pas $\oplus$-indécomposable. Comme la taille des permutations diminue strictement à chaque étape, ce processus termine (au pire, sur la permutation de taille 1), et assure que σ s'écrit sous la forme $12\dots k[\alpha^{(1)},\dots,\alpha^{(k)}]$ où les $\alpha^{(i)}$ sont tous $\oplus$-indécomposables.

L'unicité de cette décomposition est directement héritée de l'unicité dans le théorème 2.33. □

Les conditions de $\oplus$- ou $\ominus$-indécomposabilité dans les deux premiers cas du théorème 2.35 sont essentielles. Par exemple, la permutation $\sigma = 1\,2\,4\,3\,5$ peut être décrite par plusieurs dilatations, dont $1\,2\,3[1,1,2\,1\,3]$ et $1\,2\,3\,4[1,1,2\,1,1]$. Mais dans la première forme, $\alpha^{(3)} = 2\,1\,3$ n'est $\oplus$-indécomposable. La décomposition unique annoncée par le théorème 2.35 est la deuxième ci-dessus, où tous les $\alpha^{(i)}$ sont bien $\oplus$-indécomposables.

Dans [AA05], M. H. Albert et M. D. Atkinson font les premiers liens entre décomposition par substitution (où les permutations simples jouent un rôle primordial) et classes de permutations. En particulier, ils utilisent la décomposition par substitution pour montrer que, si une classe contient un nombre fini de permutations simples, alors elle a une base finie et sa série génératrice est algébrique. Ils expliquent aussi comment calculer ces séries génératrices de manière effective. Leurs résultats et les méthodes développées sont étendues dans [BHV08b] pour l'énumération de permutations à motifs exclus particulières (par exemple des involutions).

La question de savoir si une classe contient un nombre fini de permutations simples apparaît alors centrale. Dans [AA05], seule une procédure de semi-décision est indiquée pour tester si une classe contient un nombre fini de permutations simples : d'après la propriété 2.27 quand on trouve deux tailles consécutives n et $n+1$ où il n'y a pas de permutations simples dans la classe, c'est que la classe ne contient aucune permutation simple de taille supérieure ou égale à n, donc on peut simplement faire des tests par taille croissante, en espérant qu'ils s'arrêtent un jour. Dans une série de trois articles [BHV08b, BHV08a, BRV08], R. Brignall, N. Ruškuc, S. Huczynska, et V. Vatter

donnent une procédure de décision résolvant ce problème, à base de techniques d'automates. On améliore leur résultat dans le chapitre 9, en décrivant un algorithme polynomial décidant si une classe contient un nombre fini de permutations simples.

2.2.3 Arbres de décomposition

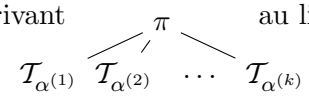
Dans les travaux de M. H. Albert et M. D. Atkinson, puis dans ceux de R. Brignall *et al.*, l'accent est mis, dans la décomposition par substitution des permutations, sur leur squelette et sur les blocs $\alpha^{(i)}$ qui dilatent ce squelette. Les raisonnements se propagent parfois à l'intérieur les blocs $\alpha^{(i)}$ pour affiner la description de la permutation, mais il n'est jamais proposé d'appliquer récursivement la décomposition par substitution des permutations, jusqu'à n'obtenir que des blocs de taille 1.

Il est pourtant naturel, en suivant les idées de R. H. Möhring et F. J. Radermacher, d'appliquer récursivement le théorème 2.35 aux blocs $\alpha^{(i)}$ que ce théorème fait apparaître, et de construire un arbre rendant compte des décompositions récursives des permutations. Cet arbre est appelé *arbre de décomposition*.

Définition 2.37. On considère une permutation σ, de taille au moins 2, et sa décomposition par substitution donnée par le théorème 2.35. On note π son squelette (ici, π peut représenter $\oplus$, $\ominus$ ou une permutation simple), et $\alpha^{(1)}, \dots, \alpha^{(k)}$ les permutations telles que $\sigma = \pi[\alpha^{(1)}, \dots, \alpha^{(k)}]$.

En notant $\mathcal{T}_{\alpha^{(i)}}$ l'arbre de décomposition associé récursivement à chaque permutation $\alpha^{(i)}$, l'*arbre de décomposition* de σ est l'unique arbre plan dont la racine est étiquetée par π, et où les sous-arbres rattachés à cette racine sont $\mathcal{T}_{\alpha^{(1)}}, \mathcal{T}_{\alpha^{(2)}}, \dots \mathcal{T}_{\alpha^{(k)}}$, dans cet ordre de gauche à droite.

Enfin, l'arbre de décomposition de la permutation 1 de taille 1 est la feuille $\bullet$.

L'arbre de décomposition est donc simplement obtenu par application récursive du théorème 2.35, et écrivant  au lieu de $\pi[\alpha^{(1)}, \dots, \alpha^{(k)}]$, et $\bullet$ au lieu de la permutation 1.

Remarque 2.38. Par canonicité de la décomposition dans le théorème 2.35, l'arbre de décomposition associé à une permutation est évidemment unique.

Exemple 2.39. On considère la permutation $\sigma = 10\,13\,12\,11\,14\,1\,18\,19\,20\,21\,17\,16\,15\,4\,8\,3\,2\,9\,5\,6\,7$. Par application récursive du théorème 2.35, elle se décompose en

$$3\,1\,4\,2[\oplus[1, \ominus[1, 1, 1], 1], 1, \ominus[\oplus[1, 1, 1, 1], 1, 1, 1], 2\,4\,1\,5\,3[1, 1, \ominus[1, 1], 1, \oplus[1, 1, 1]]].$$

L'arbre de décomposition associé à σ, qui rend compte de cette décomposition récursive, est donné par la figure 2.7.

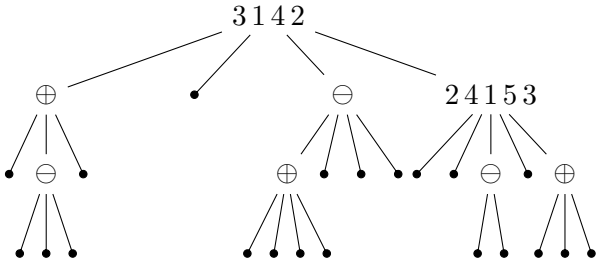


FIG. 2.7 – L'arbre de décomposition de la permutation $\sigma = 10\,13\,12\,11\,14\,1\,18\,19\,20\,21\,17\,16\,15\,4\,8\,3\,2\,9\,5\,6\,7$.

Récursivement, on se rend compte que chaque feuille de l'arbre de décomposition correspond à un élément de la permutation. La structure de l'arbre et l'étiquetage des nœuds internes indiquent la manière dont ces éléments atomiques se combinent pour donner la permutation de départ.

Selon leur étiquetage, les nœuds internes des arbres de décomposition se divisent en deux catégories :

Définition 2.40. Les nœuds internes d'un arbre de décomposition sont étiquetés par $\oplus$, $\ominus$ ou par des permutations simples. Dans les deux premiers cas, on parle de nœuds *linéaires*, sinon de nœuds *premiers*.

Les permutations $12\dots k$ et $k\dots 21$ associées aux nœuds linéaires correspondent aux structures linéaires au sens de R. H. Möhring et F. J. Radermacher [MR84]. Les permutations simples, étiquetant les nœuds premiers, correspondent aux structures premières dans ce contexte. C'est la terminologie de permutation simple (et non permutation première) qui a été adoptée dans la littérature, mais comme le font remarquer R. H. Möhring et F. J. Radermacher dès le premier paragraphe de [MR84], la primalité et la simplicité (auxquelles on peut adjoindre l'indécomposabilité) sont des synonymes désignant le même concept dans des communautés différentes.

Remarque 2.41. Une conséquence directe du théorème 2.35 (dans ses deux premiers cas) est que, dans un arbre de décomposition, il n'y a jamais d'arêtes entre deux nœuds linéaires de même signe.

Concernant les arités des nœuds dans les arbres de décomposition, l'arité de tout nœud premier est égale à la taille de la permutation simple qui étiquette ce nœud (donc est supérieure ou égale à 4). Quant aux nœuds linéaires, ils peuvent avoir n'importe quelle arité, à partir de 2.

On démontre maintenant que tous les arbres qui satisfont les conditions de la remarque 2.41 sont des arbres de décomposition.

Définition 2.42. On note $\mathbb{T}_n$ l'ensemble des arbres plans (c'est-à-dire où les fils de tout nœud interne sont totalement ordonnés) à n feuilles qui satisfont les propriétés suivantes :

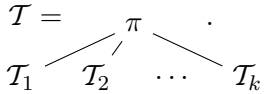
- les nœuds internes sont étiquetés par $\oplus$, $\ominus$, ou par des permutations simples,
- les nœuds d'étiquette $\oplus$ ou $\ominus$ sont d'arité supérieure ou égale à 2,
- les nœuds étiquetés par des permutations simples σ ont pour arité la taille de σ (qui est supérieure ou égale à 4 par définition),
- il n'y a pas d'arête entre deux nœuds d'étiquette $\oplus$, ni entre deux nœuds d'étiquette $\ominus$.

Par le calcul de son arbre de décomposition, on peut associer à toute permutation de taille n un arbre de la famille $\mathbb{T}_n$. Réciproquement, à tout arbre de cette famille, on peut associer une permutation :

Proposition 2.43. *Étant donné un arbre $\mathcal{T}$ de la famille $\mathbb{T}_n$, il existe une permutation σ de taille n telle que $\mathcal{T}$ soit l'arbre de décomposition de σ.*

Démonstration. La preuve est immédiate par récurrence.

On note π l'étiquette de la racine de $\mathcal{T}$ (π est $\oplus$, $\ominus$ ou une permutation simple), k son arité, et $\mathcal{T}_1, \mathcal{T}_2 \dots, \mathcal{T}_k$ les sous-arbres de $\mathcal{T}$ greffés sous sa racine, de gauche à droite.

Plus graphiquement, $\mathcal{T} =$  .

Par récurrence, les arbres $\mathcal{T}_1, \mathcal{T}_2 \dots, \mathcal{T}_k$ sont les arbres de décomposition de permutations, que l'on note $\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(k)}$. Alors on prétend que $\mathcal{T}$ est l'arbre de décomposition de $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(k)}]$. Il suffit pour cela de vérifier que la décomposition de la permutation σ en $\pi[\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(k)}]$ est sa décomposition par substitution, au sens du théorème 2.35.

Les propriétés de l'arbre $\mathcal{T}$ assurent que c'est le cas. En particulier, comme $\mathcal{T}$ ne contient aucune arête entre deux nœud $\oplus$ (resp. $\ominus$), cela assure que, dans le cas où $\pi = \oplus$ (resp. $\ominus$), les arbres $\mathcal{T}_i$ n'ont jamais une racine d'étiquette $\oplus$ (resp. $\ominus$), et donc par unicité dans le théorème 2.35, que les permutations $\alpha^{(i)}$ sont toutes $\oplus$-indécomposables (resp. $\ominus$-indécomposables). □

La propriété 2.43 autorise donc à énoncer le théorème suivant :

Théorème 2.44. *La correspondance entre permutations de taille n et arbres de la famille $\mathbb{T}_n$ (leurs arbres de décomposition) est bijective.*

Cette vision des permutations à travers leurs arbres de décomposition est cruciale dans beaucoup des résultats de cette thèse, à tel point qu'on confondra parfois les deux objets, en s'autorisant des abus de vocabulaire comme la racine d'une permutation. Dans la même idée, pour que les tailles d'une permutation et de son arbre de décomposition coïncident, la taille d'un arbre de décomposition désigne son nombre de feuilles.

Les arbres de décomposition sont un codage des permutations, de même que les arbres de décomposition modulaire des graphes, si on étiquette les nœuds premiers par leur graphe quotient, sont un codage des graphes. En général, l'étiquetage des nœuds premiers par leurs graphes quotient n'est pas explicite. Au contraire, on a l'habitude de conserver explicitement les permutations simples sur les nœuds premiers des arbres de décomposition des permutations.

Pour pouvoir travailler indifféremment avec les permutations en elles-même ou avec leurs arbres de décomposition, il faut évidemment se poser la question du calcul de l'arbre à partir de la permutation, et vice versa. Ces questions algorithmiques sont résolues, et on peut retenir le résultat suivant :

Théorème 2.45. *Le calcul de l'arbre de décomposition d'une permutation de taille n peut être réalisé en temps $\mathcal{O}(n)$. Réciproquement, étant donné un arbre de décomposition, on peut retrouver la permutation associée en temps linéaire.*

Démonstration. On commence par démontrer l'affirmation réciproque, en décrivant un algorithme linéaire de calcul de la permutation σ étant donné son arbre de décomposition.

L'algorithme fonctionne par deux parcours en profondeur de l'arbre. Dans le premier parcours, les arêtes sortant d'un nœud sont empruntées dans l'ordre indiqué par l'étiquette de ce nœud :

- de gauche à droite pour un nœud $\oplus$,
- de droite à gauche pour un nœud $\ominus$,
- pour un nœud étiqueté par une permutation simple π de taille k, d'abord l'arête qui dilate l'élément 1 de π, c'est-à-dire la $\pi^{-1}(1)$-ème arête, puis la $\pi^{-1}(2)$-ème, et ainsi de suite, pour finir par la $\pi^{-1}(k)$-ème.

Au cours de cette exploration en profondeur de l'arbre, les feuilles sont numérotées de 1 à n, dans l'ordre où elles sont rencontrées.

Dans le deuxième parcours en profondeur, les arêtes sortant de chaque nœud sont empruntées de gauche à droite, et la permutation retournée par l'algorithme est celle lue sur les feuilles de l'arbre dans ce parcours.

Pour justifier que la permutation calculée par cet algorithme est bien la permutation de départ σ, il suffit de s'apercevoir que deux éléments σ_i et σ_j vérifient $\sigma_i < \sigma_j$ si et seulement si, dans l'arbre de décomposition de σ, le plus petit ancêtre commun des feuilles correspondant à ces éléments est étiqueté par π, tel que $\pi_a < \pi_b$, a et b étant défini de sorte que σ_i se trouve dans le bloc dilatant π_a et σ_j dans le bloc dilatant π_b.

La permutation calculée par l'algorithme vérifiant de manière évidente ces conditions, cela justifie que l'algorithme décrit est correct : il retourne bien la permutation σ. De point de vue de la complexité, la procédure décrite est linéaire en la taille de l'arbre, et donc aussi en n.

Le sens direct du théorème 2.45 est en fait démontré non pas pour les arbres de décomposition des permutations, mais pour leurs *arbres des intervalles communs*. On reviendra sur ce résultat au paragraphe 2.3.3, théorème 2.56. □

Le point de vue adopté pour la définition des arbres des intervalles communs des permutations est assez différent de celui de la décomposition par substitution de M. H. Albert et M. D. Atkinson.

Cette dernière procède "par le haut", en décomposant σ en un squelette dilaté par des permutations plus petites, décomposables à leur tour en la dilatation de permutations encore plus petites. Au contraire, la décomposition d'une permutations en intervalles communs est plutôt une approche "par le bas", comme on va le voir dans le paragraphe 2.3.1.

On expliquera aussi dans le paragraphe 2.3.2 que les arbres de décomposition et les arbres des intervalles communs sont en fait deux manières différentes de voir la même structure, ce qui justifiera le théorème 2.45.

2.3 Point de vue algorithmique : l'arbre des intervalles communs

Dans un contexte plus algorithmique que combinatoire, on n'étudie pas en général les arbres de décomposition mais une autre structure arborescente venant se poser sur les permutations : l'*arbre des intervalles communs*. Cet arbre permet de stocker dans une structure de taille linéaire l'ensemble des intervalles (de cardinal *a priori* quadratique) d'une permutation.

La recherche des intervalles d'une permutation, ou plus généralement des intervalles communs à un ensemble de permutations, est un problème qui trouve ses motivations en bio-informatique, pour la détection d'association entre gènes [UY00, HS01]. Une autre motivation pour ce problème, indiquée dans [HS01], est que la recherche des intervalles d'une permutation peut être vue comme le problème réciproque de celui de l'arrangement consécutif : étant donnés un ensemble E et une collection $\mathcal{E}$ de sous-ensembles de E, on cherche une permutation σ des éléments de E telle que les éléments de chaque sous-ensemble appartenant à $\mathcal{E}$ forment un intervalle de σ.

Parmi les utilisations de l'arbre des intervalles communs pour résoudre des problèmes bio-informatiques, on peut citer [BBCP07, BCP08] qui s'intéressent au problème du tri par renversements, [BCC+08] pour l'étude du modèle *Double-Cut-and-Join*, ou encore [BCMR08] qui souligne l'importance des PQ-arbres (très proches cousins des arbres des intervalles communs) pour la génomique comparative. On reviendra en détails au chapitre 7 sur le problème du tri par renversements.

Une autre utilisation des arbres des intervalles communs est pour le calcul des arbres de décomposition modulaire des graphes. On a vu au paragraphe 2.1.2 que certains algorithmes de calcul de l'arbre de décomposition modulaire des graphes sont formés de deux étapes : le calcul d'une permutation factorisante σ pour le graphe, puis le calcul de l'arbre de décomposition modulaire à partir de σ. Le point important, souligné indépendamment par [BXHP05] et [BCMR05, BCMR08], est que le calcul de l'arbre de décomposition modulaire du graphe, étant donnée une permutation factorisante σ, suit exactement les mêmes étapes que le calcul de l'arbre des intervalles communs de σ, et que ces deux arbres ont la même structure. Les algorithmes de calcul de l'arbre des intervalles communs fournissent donc des algorithmes pour la deuxième étape du calcul de l'arbre de décomposition modulaire à partir d'une permutation factorisante. On verra au paragraphe 2.3.3 qu'il existe des algorithmes pour le calcul de l'arbre des intervalles communs fonctionnant en temps linéaire.

Avant cela, on définit cette structure d'arbre des intervalles communs au paragraphe 2.3.1, et on montre les liens très forts qui relient arbres des intervalles communs et arbres de décomposition. On verra en effet au paragraphe 2.3.2 que ces deux structures sont identiques, à étiquetage des nœuds près.

2.3.1 Définition et premières propriétés

On présente ici brièvement les définitions et quelques propriétés classiques de l'arbre des intervalles communs. Pour plus amples détails, on peut par exemple se référer à [BXHP05].

Définition 2.46. Les *intervalles communs* à un ensemble $\mathcal{P}$ de permutations de taille n sont les sous-ensembles I de $[1..n]$ tels que les éléments de I apparaissent de manière contiguë (mais dans

un ordre quelconque) dans chacune des permutations de $\mathcal{P}$. L'ordre d'apparition des éléments de I peut être différent d'une permutation à l'autre.

On suppose en général que l'identité de taille n appartient à $\mathcal{P}$, ce qui n'est pas une restriction, à renumérotation près. Cela permet que les intervalles communs I soient des intervalles d'entiers, ce qui n'est pas le cas *a priori*.

Exemple 2.47. Les intervalles communs aux trois permutations $\sigma_1 = 2\,4\,6\,8\,1\,3\,5\,7\,9$, $\sigma_2 = 6\,4\,2\,8\,5\,9\,3\,1\,7$ et $\sigma_3 = 5\,9\,7\,1\,3\,2\,6\,8\,4$ sont les ensembles d'entiers $\{1, 3\}$, $\{2, 4, 6, 8\}$ et $\{1, 3, 5, 7, 9\}$.

Ici on considèrera seulement le cas où $\mathcal{P}$ est de cardinal 2, et on supposera toujours qu'une des deux permutations est l'identité. On s'intéressera donc aux intervalles communs de Id_n et de $\sigma \in \mathcal{S}_n$, qui correspondent aux intervalles de σ au sens de la définition 2.15. Dans toute la suite, on parlera donc d'intervalles plutôt que d'intervalles communs, mais les notions présentées ici se généralisent au cas des intervalles communs à $d \geq 2$ permutations.

On rappelle que parmi les intervalles d'une permutation σ, on peut distinguer les intervalles *forts*, qui sont ceux ne chevauchant aucun intervalle de σ (voir définition 2.19). Cette définition implique que, si l'on compare deux intervalles forts I et J pour l'ordre d'inclusion, on est nécessairement dans l'une des trois situations suivantes : $I \subseteq J$, $J \subseteq I$, ou $I \cap J = \emptyset$. L'ordre d'inclusion sur l'ensemble des intervalles forts d'une permutation σ a donc une forme arborescente. Ceci amène à la définition suivante :

Définition 2.48. L'ordre d'inclusion entre les intervalles forts d'une permutation $\sigma \in \mathcal{S}_n$ induit une structure d'arbre. Cet arbre est appelé *arbre des intervalles communs* de σ : ces nœuds sont les intervalles forts de σ, sa racine est $[1..n]$, ses feuilles sont les singletons $\{1\}, \{2\}, \ldots, \{n\}$, et l'intervalle de tout nœud interne est la réunion des intervalles de ses fils.

Ainsi défini, l'arbre des intervalles communs de σ n'est pas un arbre plan. Il existe cependant une manière canonique de plonger cet arbre dans le plan. En effet, les feuilles de cet arbre correspondent aux intervalles triviaux $\{1\}, \{2\}, \ldots, \{n\}$, et on peut donc choisir le plongement canonique de sorte que la lecture de gauche à droite des feuilles de l'arbre donne la permutation σ.

Exemple 2.49. Un exemple d'arbre des intervalles communs est donné par la figure 2.8.

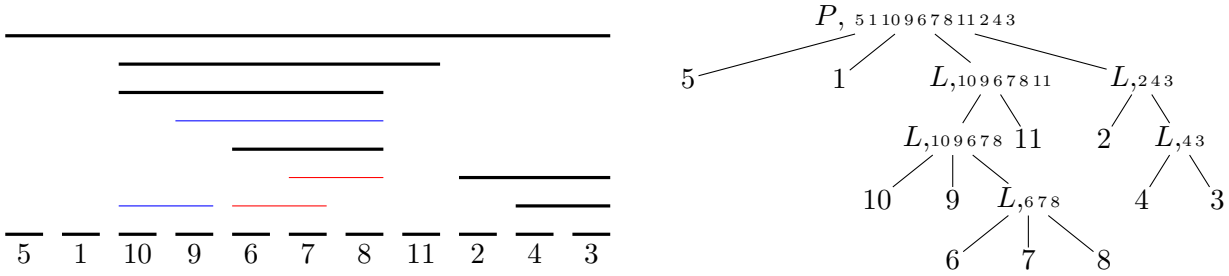


FIG. 2.8 – Les intervalles de la permutation $\sigma = 5\,11\,10\,9\,6\,7\,8\,11\,2\,4\,3$, avec les intervalles forts représentés en gras. L'ordre d'inclusion de ces intervalles forts est représenté par l'arbre des intervalles communs de σ, à droite.

Remarque 2.50. On se convainc facilement, par un raisonnement récursif, que l'intervalle associé à tout nœud interne V de l'arbre des intervalles communs est l'ensemble des valeurs étiquetant les feuilles du sous-arbre enraciné en V.

Les nœuds internes des arbres des intervalles communs se répartissent en deux catégories : les nœuds premiers (de type P) et les nœuds linéaires (de type L). Pour définir ces deux catégories, on considère l'arbre des intervalles communs plongé dans le plan de manière canonique, et on dispose ainsi pour tout nœud interne d'un ordre entre ses fils. Les nœuds premiers sont ceux tels qu'aucune union d'un ensemble consécutif de leurs fils (dans l'ordre induit par le plongement canonique) ne corresponde à un intervalle, hormis la réunion de tous les fils. Un nœud est linéaire lorsqu'une union de ses fils correspond à un intervalle si et seulement si ces fils sont consécutifs. Il est démontré (dans [BXHP05] par exemple) qu'il n'y a pas d'autre possibilité.

Il est d'usage d'indiquer le type de chaque nœud sur l'arbre des intervalles communs, comme cela est fait sur la figure 2.8.

On peut remarquer que les nœuds d'arité 2 sont, au sens de la définition ci-dessus, à la fois premiers et linéaires. Pour les besoins de cette étude, on leur attribuera le type L. Dans d'autres contextes, il est parfois plus judicieux de faire le choix de leur donner le type P.

Une légère variante des arbres des intervalles communs consiste à ne pas plonger l'arbre dans le plan, mais à ordonner les fils des nœuds linéaires selon un ordre compatible avec leur définition : une union de fils d'un nœud linéaire doit correspondre à un intervalle si et seulement si ces fils sont consécutifs dans l'ordre choisi (et un nœud est linéaire si et seulement si un tel ordre existe). Les nœuds premiers restent non ordonnés. La structure ainsi obtenue ne s'appelle plus arbre des intervalles communs mais *PQ-arbre*. Dans les PQ-arbres, les nœuds premiers sont appelés nœuds P et les nœuds linéaires nœuds Q.

Enfin, on utilise parfois un code de forme pour distinguer les nœuds premiers et les nœuds linéaires sans alourdir les notations comme dans la figure 2.8 : on utilise une forme rond pour représenter les nœuds premiers et une forme carrée pour les nœuds linéaires.

Algorithmiquement parlant, la propriété principale des arbres des intervalles communs est qu'ils permettent le stockage dans une structure linéaire de toute l'information nécessaire à décrire tous les intervalles communs d'une permutation : ils sont soit un nœud premier, soit la réunion de fils consécutifs d'un nœud linéaire. Des exemples d'utilisation en algorithmique de l'arbre des intervalles communs ont été donnés au début du paragraphe 2.3. Les aspects algorithmiques du calcul de ces arbres seront étudiés au paragraphe 2.3.3. Avant cela, on lève le voile sur les relations très étroites qui unissent les arbres des intervalles communs aux arbres de décomposition des permutations.

2.3.2 Lien étroit avec les arbres de décomposition

Sur un arbre des intervalles communs, on peut proposer un étiquetage supplémentaire des nœuds internes, par des permutations de $\mathcal{S}$, appelées permutations *quotient*.

Définition 2.51. On considère l'arbre des intervalles communs d'une permutation σ, plongé dans le plan de manière canonique. Tout nœud interne V de cet arbre correspond à un intervalle I, et les fils $V_1, V_2, \ldots, V_k$ de V (dans l'ordre de gauche à droite) correspondent à des intervalles $I_1, I_2, \ldots I_k$ dont la réunion donne I. La *permutation quotient* associée au nœud V est la permutation ρ de $\mathcal{S}_k$ telle que $\rho_i < \rho_j$ si et seulement si $I_i < I_j$, c'est-à-dire si et seulement si chaque élément de I_i est inférieur à chaque élément de I_j.

Exemple 2.52. La permutation quotient associée à la racine de l'arbre de la figure 2.8 est 3 1 4 2. Celle associée à son fils le plus à droite est 1 2.

Par définition des nœuds premiers, les permutations quotient qui les étiquettent n'ont aucun intervalle non trivial, c'est-à-dire que ce sont des permutations simples. Pour les nœuds linéaires, sachant que toute union de fils consécutifs doit former un intervalle, les seules permutations quotient possibles sont de la forme $12\ldots k$ (notées $\oplus$) ou $k\ldots 21$ (notées $\ominus$).

Proposition 2.53. *Par rapport à l'étiquetage des nœuds de l'arbre des intervalles communs par des intervalles, l'étiquetage par des permutations quotient est en fait un étiquetage alternatif plutôt qu'un étiquetage supplémentaire : on peut retrouver les intervalles étiquetant les nœuds à partir des permutations quotient.*

Démonstration. On se donne une permutation σ de taille n, et son arbre des intervalles communs $\mathcal{T}$. On étiquette les nœuds internes de $\mathcal{T}$, de type P ou L, par leurs permutations quotient, et on efface les intervalles étiquetant les nœuds de $\mathcal{T}$. Les nœuds premiers sont donc étiquetés par des permutations simples, et les nœuds linéaires par des symboles $\oplus$ ou $\ominus$, selon que la permutation quotient est l'identité ou son miroir.

Par un traitement de l'arbre de haut en bas, on attribue aux nœuds V de $\mathcal{T}$ des intervalles d'entiers, qui contiennent chacun autant de valeurs qu'il y a de feuilles dans le sous-arbre de $\mathcal{T}$ enraciné en V.

À la racine, on attribue l'intervalle $[1..n]$. On note k l'arité de la racine, et $V_1, V_2, \dots V_k$ ses fils, de gauche à droite. On note t_i le nombre de feuilles dans le sous-arbre de $\mathcal{T}$ de racine V_i.

Si la racine est étiquetée par $\oplus$, on attribue à ses fils $V_1, V_2, \dots V_k$ les intervalles $[1..t_1], [(t_1 + 1)..(t_1 + t_2)], \dots, [(t_1 + t_2 + \dots + t_{k-1} + 1)..n]$ respectivement.

Si elle est étiquetée par $\ominus$, on attribue à ses fils $V_k, V_{k-1}, \dots V_1$ (de droite à gauche donc) les intervalles $[1..t_k], [(t_k + 1)..(t_k + t_{k-1})], \dots, [(t_k + t_{k-1} + \dots + t_2 + 1)..n]$ respectivement.

Si elle est étiquetée par une permutation simple π, pour $i_1 = \pi^{-1}(1)$, le fils V_{i_1} reçoit l'intervalle $[1..t_{i_1}]$. Puis pour $i_2 = \pi^{-1}(2)$, on attribue à V_{i_2} l'intervalle $[(t_{i_1} + 1)..(t_{i_1} + t_{i_2})]$, et ainsi de suite jusqu'à $i_k = \pi^{-1}(k)$ pour lequel V_{i_k} reçoit l'intervalle $[(t_{i_1} + t_{i_2} + \dots + t_{i_{k-1}} + 1)..n]$.

On procède ainsi récursivement dans les sous-arbres, en découpant à chaque étape l'intervalle associé à un nœud en intervalles plus petits attribués à ses fils, pour distribuer jusque dans les feuilles les entiers entre 1 et n. Évidemment, l'intervalle associé à un nœud selon ce procédé est la réunion des intervalles associés à ses fils.

Enfin, la manière dont procède le découpage de l'intervalle I d'un nœud V en intervalles $I_1, I_2, \dots, I_k$ de ses fils (de gauche à droite) assure que la permutation quotient associée à V selon la définition 2.51 est bien la permutation qui étiquette le nœud V.

La structure de l'arbre étant demeurée inchangée, pour conclure que l'étiquetage par des intervalles qui a été calculé pour les nœuds de $\mathcal{T}$ correspond bien à celui qui définit l'arbre des intervalles communs, il suffit maintenant de vérifier que la permutation que l'on peut lire de gauche à droite sur les feuilles de $\mathcal{T}$ dans l'étiquetage calculé (que l'on note τ) est la permutation de départ σ.

Pour cela, on fixe $i < j$, et on montre que la relation d'ordre entre τ_i et τ_j est la même qu'entre σ_i et σ_j. On considère la i-ème et la j-ème feuille de $\mathcal{T}$, dans l'ordre de gauche à droite. Ces feuilles sont étiquetées par σ_i et σ_j dans l'arbre des intervalles communs de σ, et par τ_i et τ_j dans l'étiquetage calculé. Or la relation d'ordre entre les étiquettes de ces deux feuilles est déterminée par la permutation quotient qui étiquette leur plus petit ancêtre commun. Cette permutation quotient étant la même pour σ et pour τ, on a bien $\sigma_i < \sigma_j$ si et seulement si $\tau_i < \tau_j$. On conclut donc que $\tau = \sigma$. □

L'arbre obtenu à partir d'un arbre des intervalles communs, en étiquetant ses nœuds par leurs permutations quotient plutôt que par leurs intervalles, a la forme d'un arbre de décomposition, au sens de la définition 2.42, à ceci près qu'il faut assurer qu'il n'y ait pas d'arête entre deux nœuds linéaires de même signe. Ce dernier point est assez évident, sachant que les nœuds correspondent à des intervalles forts : si un nœud linéaire V a un fils linéaire V' de même signe, un intervalle chevauchant V' peut être construit comme union du dernier fils de V à gauche de V' et du fils le plus à gauche de V' (ou du premier fils de V à droite de V' et du fils le plus à droite de V'), contredisant ainsi que V' est un intervalle fort.

Plus précisément, et comme on pouvait s'y attendre, l'arbre des intervalles communs d'une permutation σ ainsi étiqueté est en fait égal à l'arbre de décomposition de σ.

Théorème 2.54. *L'arbre des intervalles communs entre Id_n et une permutation $\sigma \in \mathcal{S}_n$, une fois ses nœuds étiquetés par leurs permutations quotient, correspond exactement à l'arbre de décomposition de σ.*

Démonstration. On considère une permutation $\sigma \in \mathcal{S}_n$ et sa décomposition par substitution $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(k)}]$, où le squelette π de σ est $12\dots k$ (ou $\oplus$), $k\dots 21$ (ou $\ominus$) ou une permutation simple. D'après la remarque 2.32, les $\alpha^{(i)}$ sont des intervalles de σ, et la définition de la dilatation (définition 2.29) assure que la permutation quotient associée à la partition de l'intervalle $[1..n]$ en les intervalles correspondant aux $\alpha^{(i)}$ est π.

Il suffit donc de vérifier que les $\alpha^{(i)}$ sont les intervalles forts maximaux de σ pour conclure par récurrence au résultat annoncé.

On suppose d'abord que $\pi = \oplus$, le cas $\ominus$ étant traité de la même façon. Par l'absurde, supposons qu'il existe un $\alpha^{(i)}$ qui n'est pas un intervalle fort. Il existe donc un intervalle I qui chevauche $\alpha^{(i)}$. Comme $\pi = \oplus$ l'intervalle I contient au moins un point appartenant à $\alpha^{(i+1)}$ ou $\alpha^{(i-1)}$. On voit alors clairement sur la figure 2.9 que l'intersection de I et de $\alpha^{(i)}$ est un intervalle qui contredit que $\alpha^{(i)}$ est $\oplus$-indécomposable.

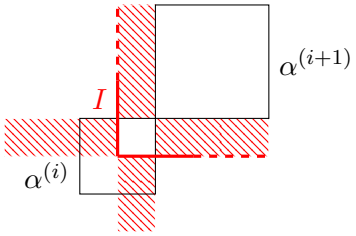


FIG. 2.9 – Preuve du théorème 2.54 : contradiction dans le cas d'un nœud linéaire.

Si au contraire π est une permutation simple, on raisonne encore par l'absurde en supposant qu'il existe un intervalle I chevauchant un intervalle $\alpha^{(i)}$, et débordant donc sur un intervalle $\alpha^{(j)}$. Alors $\alpha^{(i)} \cup \alpha^{(j)} \cup I$ est un intervalle, car la boîte englobante de $\alpha^{(i)} \cup \alpha^{(j)} \cup I$, c'est-à-dire le plus petit rectangle contenant $\alpha^{(i)} \cup \alpha^{(j)} \cup I$ (voir définition page 172), n'admet aucun point sur ses côtés (voir figure 2.10, où la boîte englobante est indiquée en pointillés). Cela implique que dans la représentation graphique de π, la boîte englobante des points $\pi(i)$, $\pi(j)$ et $\pi(\ell)$, pour les indices ℓ tels que $\alpha^{(\ell)} \cap I \neq \emptyset$, est un intervalle, et comme π est simple, cet intervalle est nécessairement $[1..k]$. Ainsi, on déduit que $\alpha^{(i)} \cup \alpha^{(j)} \cup I = [1..n]$. Supposant que $\alpha^{(i)}$ est plus à gauche (resp. plus à droite) que $\alpha^{(j)}$, on a alors $i = 1$ (resp. $i = n$). En outre, si $\alpha^{(i)}$ est situé plus bas que $\alpha^{(j)}$, on obtient que $1 \in \alpha^{(i)}$, sinon, c'est n qui appartient à $\alpha^{(i)}$. Ainsi, dans le squelette π, on a soit $\pi(1) = 1$, soit $\pi(1) = k$ (resp. $\pi(k) = 1$, soit $\pi(k) = k$). Dans les deux cas, $\pi_2\pi_3\dots\pi_k$ (resp. $\pi_1\pi_2\dots\pi_{k-1}$) est un intervalle de π, contredisant que π est simple.

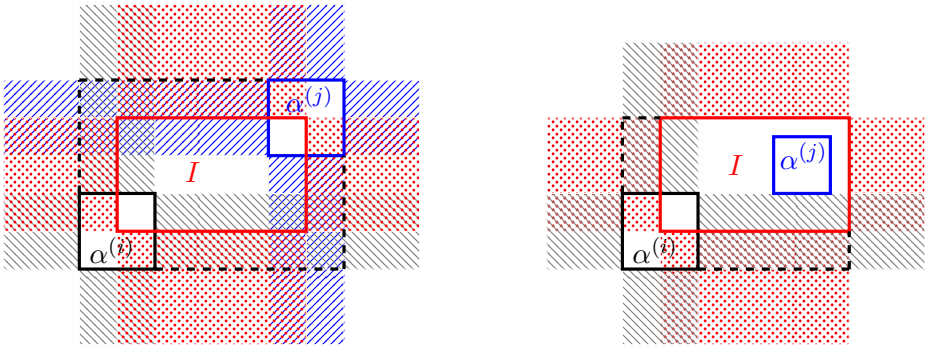


FIG. 2.10 – Preuve du théorème 2.54 : contradiction dans le cas d'un nœud premier.

Ceci assure que les $\alpha^{(i)}$ sont des intervalles forts. On montre à présent qu'ils sont des intervalles forts maximaux. Tout d'abord, comme les $\alpha^{(i)}$ sont des intervalles forts, tout intervalle fort qui contient l'un d'eux est nécessairement une union de $\alpha^{(i)}$. Une telle union de $\alpha^{(i)}$ correspond à un ensemble d'éléments de la permutation squelette π, et une union de $\alpha^{(i)}$ est un intervalle si et seulement si les éléments correspondant dans π forment aussi un intervalle.

Lorsque π est une permutation simple, le seul intervalle de cardinal supérieur à 1 de π est la réunion de tous les éléments de π, ce qui assure que $[1..n]$ est le seul intervalle fort dans lequel sont contenus les $\alpha^{(i)}$. Ce raisonnement est valable aussi lorsque $\pi = 1\,2$ ou $2\,1$.

Si π est de la forme $12\dots k$ ou $k\dots 21$ avec $k \geq 3$, les intervalles de π de cardinal supérieur à 1 sont tous les $[i..j]$ pour $1 \leq i < j \leq k$. Parmi ces intervalles, le seul qui n'en chevauche aucun autre est $[1..k]$, ce qui assure que $[1..n]$ est le seul intervalle fort de σ dans lequel sont contenus les $\alpha^{(i)}$.

Chaque $\alpha^{(i)}$ est donc un intervalle fort maximal de σ, et comme leur union est $[1..n]$, ils représentent bien tous les intervalles forts maximaux de σ, ce qui conclut la preuve. □

Remarque 2.55. Même si ce n'est pas le point de vue qui a été adopté ici, la preuve précédente peut être formulée en termes de propriétés des familles faiblement partitives [Mon03, par exemple].

Les arbres de décomposition et les arbres des intervalles communs représentent donc essentiellement la même structure, mais dans des contextes d'étude différents. La question du calcul de cet arbre a été étudiée en détails pour les arbres des intervalles communs, et les résultats se transfèrent sans difficulté aux arbres de décomposition.

2.3.3 Algorithmes de calcul en temps linéaire

On présente maintenant un bref historique des algorithmes de calcul de l'arbre des intervalles communs, et les résultats de complexité obtenu. On verra ensuite comment calculer l'étiquetage par des permutations quotient qui donne l'arbre de décomposition.

Calcul de l'arbre des intervalles communs

Comme pour la définition de l'arbre des intervalles communs, on s'intéresse ici au calcul de l'arbre des intervalles communs à deux permutations de même taille (notée n), dont une identité. Les méthodes présentées s'étendent cependant à un ensemble de $d \geq 2$ permutations (de même taille toujours) dont une identité.

Dans tout ce paragraphe, on note n la taille des deux permutations en entrée, et K le nombre de leurs intervalles communs.

L'algorithme fondateur pour le calcul de l'ensemble des intervalles communs à deux permutations est dû à T. Uno et M. Yagiura en 2000 [UY00]. Cet algorithme permet de calculer l'ensemble des intervalles communs à deux permutations en temps $\mathcal{O}(n + K)$. *A priori*, K est de l'ordre de $\mathcal{O}(n^2)$. L'algorithme de T. Uno et M. Yagiura n'est donc pas linéaire en la taille de l'entrée, mais il est linéaire en la taille du résultat calculé.

Les améliorations développées à partir de l'algorithme de T. Uno et M. Yagiura, pour obtenir des algorithmes linéaires en la taille de l'entrée, consistent à trouver une structure de taille linéaire en n, calculable en temps $\mathcal{O}(n)$, et à partir de laquelle on peut retrouver en temps $\mathcal{O}(K)$ l'ensemble des K intervalles communs. L'arbre des intervalles communs est une structure qui satisfait ces propriétés, mais il en existe d'autres, comme l'ensemble des intervalles irréductibles, ou les générateurs.

La première amélioration de ce type est due à S. Heber et J. Stoye en 2001. Dans [HS01], ils donnent un algorithme linéaire en n (adapté de celui de T. Uno et M. Yagiura) pour calculer un sous-ensemble de l'ensemble des intervalles communs, appelés les intervalles *irréductibles*. Il y a au plus $n - 1$ tels intervalles. À partir de l'ensemble des intervalles irréductibles, S. Heber et J. Stoye donnent aussi un algorithme pour calculer en temps $\mathcal{O}(K)$ la totalité des intervalles communs.

Dans [BXHP05], B.-M. Bui Xuan, M. Habib et C. Paul proposent une adaptation de l'algorithme de T. Uno et M. Yagiura qui utilise l'arbre des intervalles communs. Cet arbre, qui permet un stockage dans une structure de taille linéaire en n de l'ensemble (*a priori* quadratique) des intervalles communs, peut être calculé en temps $\mathcal{O}(n)$. Et il permet de produire en extension l'ensemble des K intervalles communs en temps $\mathcal{O}(K)$.

L'algorithme de [BXHP05] permet de calculer les intervalles forts en temps $\mathcal{O}(n)$, et d'obtenir en même temps l'ordre d'inclusion entre les intervalles forts, c'est-à-dire (à détermination près des nœuds premiers et linéaires) l'arbre des intervalles communs.

Ici, le sous-ensemble des intervalles qui contient toute l'information nécessaire est l'ensemble des intervalles forts, au lieu de l'ensemble des intervalles irréductibles de [HS01]. [BXHP05] montre aussi comment les intervalles irréductibles peuvent être lus sur l'arbre des intervalles communs, et comment on peut passer d'une structure à l'autre en temps $\mathcal{O}(n)$.

Outre en simplifier la preuve, l'adaptation de l'algorithme de T. Uno et M. Yagiura proposée par [BXHP05] permet d'élargir la méthode au champ de la décomposition modulaire des graphes. Étant donnée une permutation factorisante d'un graphe, la méthode développée par [BXHP05] pour le calcul de l'arbre des intervalles communs permet en effet de calculer l'arbre de décomposition modulaire du graphe. Cela fait des permutations un point d'entrée pour concevoir des algorithmes sur les graphes.

Dans les travaux [BCMR08, BCMR05] de A. Bergeron, C. Chauve, F. de Montgolfier et M. Raffinot, la base linéaire qui permet de construire l'ensemble des intervalles communs n'est plus un sous-ensemble de ces intervalles (comme les intervalles irréductibles de [HS01] ou les intervalles forts de [BXHP05]), mais de nouveaux objets, les *générateurs*, qui sont des paires de tableaux (R, L) satisfaisant des propriétés particulières. C'est donc une structure de données plus simple, qui donne lieu a une autre méthode (plus simple aussi que l'algorithme de Uno et Yagiura) pour le calcul de l'arbre des intervalles communs à deux permutations.

Parmi tous les générateurs possibles pour un ensemble de 2 (ou $d \geq 2$) permutations, les générateurs *commutants* possèdent des propriétés supplémentaires et permettent de calculer aisément l'ensemble des intervalles communs en temps $\mathcal{O}(K)$.

Un générateur commutant (la paire notée (Sup, Inf) dans [BCMR08, BCMR05]) est facilement construit en temps $\mathcal{O}(n)$.

Et parmi les générateurs commutants, il est est un (le générateur *canonique*), que l'on peut construire en temps linéaire à partir de tout générateur commutant, et grâce auquel on peut revenir à la notion d'intervalles forts et de PQ-arbre. Le passage de la représentation des intervalles communs par leur générateur canonique à la représentation par ensemble des intervalles forts est linéaire. Le calcul de l'arbre des intervalles communs à partir des intervalles forts doit être effectué dans un deuxième temps (avec calcul de l'ordre arborescent d'inclusion, puis étiquetage par P ou Q des nœuds), mais là encore, [BCMR08, BCMR05] décrivent une procédure pour le faire en temps linéaire. Réciproquement, on peut passer en temps linéaire du PQ-arbre au générateur canonique en temps linéaire.

Ces travaux présentent ainsi plusieurs preuves du résultat de complexité suivant :

Théorème 2.56. *L'arbre des intervalles communs d'une permutation $\sigma \in \mathcal{S}_n$ peut être calculé en temps $\mathcal{O}(n)$.*

Ajout des permutations quotient étiquetant les nœuds

Il faut remarquer que le problème du calcul des permutations quotient étiquetant les nœuds de l'arbre de décomposition, bien que facile à résoudre, n'est pas étudié à ma connaissance dans la littérature. On propose ici un algorithme linéaire en la taille de la permutation pour résoudre ce problème.

Théorème 2.57. *On considère une permutation σ de taille n, et son arbre des intervalles communs $\mathcal{T}$. Le calcul des permutations quotient associées à tous les nœuds de $\mathcal{T}$ peut être fait en temps $\mathcal{O}(n)$.*

Démonstration. L'algorithme 2.1 décrit la procédure pour calculer les permutations quotient associées à tous les nœuds de $\mathcal{T}$.

L'idée de l'algorithme est la suivante : les éléments de 1 à n remontent depuis les feuilles de $\mathcal{T}$ jusqu'à la racine, tant que les arêtes empruntées n'ont pas encore été visitées. Le long de ce chemin, les éléments de 1 à n impriment leur valeur dans tous les nœuds rencontrés : à chaque nœud V est associé à un tableau T_V, et la valeur remontant depuis la k-ème arête sortant de V (dans l'ordre de gauche à droite) s'imprime dans $T_V[k]$. Ainsi, chaque nœud interne V de $\mathcal{T}$ contient un tableau d'entiers distincts dont la taille est égale au nombre de fils de V. Il faut ensuite procéder à la normalisation de ces séquences d'entiers distincts pour obtenir les permutations quotient.

Pour éviter cette étape de normalisation, l'algorithme 2.1 utilise un compteur par nœud interne de $\mathcal{T}$, mais le principe reste le même.

Algorithme 2.1 Algorithme de calcul des permutations quotient associées à tous les nœuds de l'arbre des intervalles communs.

1: Données : L'arbre des intervalles communs $\mathcal{T}$ d'une permutation σ de taille n.

2: **pour** tout nœud interne V de $\mathcal{T}$ **faire**
3: Créer un compteur c_V initialisé à 1.
4: Créer un tableau T_V de longueur égale à l'arité de V.
5: **fin pour**
6: Créer l'ensemble $\mathcal{E}$ des arêtes visitées, initialisé à $\emptyset$.

7: **pour** i allant de 1 à n **faire**
8: On considère le chemin $\mathcal{C}$ de la feuille correspondant à $\{i\}$ à la racine.
9: **pour** toutes les arêtes (V, W) de $\mathcal{C}$, W étant le fils de V **faire**
10: **si** l'arête (V, W) n'a pas encore été visitée par l'algorithme (c'est-à-dire $(V, W) \notin \mathcal{E}$) **alors**
11: $k \leftarrow$ l'entier t.q. (V, W) est la k-ème arête sortant de V, dans l'ordre de gauche à droite
12: $T_V[k] \leftarrow c_V$
13: $c_V \leftarrow c_V + 1$
14: Insérer (V, W) dans $\mathcal{E}$.
15: **finsi**
16: **fin pour**
17: **fin pour**

18: Sortie : L'arbre $\mathcal{T}$ où les nœuds internes V sont étiquetés par les permutations contenues dans les tableaux T_V.

Pour obtenir l'étiquetage des nœuds linéaires de $\mathcal{T}$ par $\oplus$ et $\ominus$ plutôt que par des permutations identités ou miroirs d'une identité, il suffit, pour chaque nœud linéaire de $\mathcal{T}$, de comparer $T_V[1]$ et $T_V[2]$: si $T_V[1] < T_V[2]$ alors le nœud est étiqueté $\oplus$, sinon il est étiqueté $\ominus$.

Le fonctionnement de l'algorithme 2.1 est illustré sur la figure 2.11.

L'algorithme 2.1 parcourant seulement une fois chaque arête de l'arbre, il a une complexité linéaire en ce nombre d'arêtes, et donc aussi en n.

Reste à justifier que les permutations T_V calculées sont bien les permutations quotient associées aux nœuds V de $\mathcal{T}$. Pour cela, on considère un nœud V de $\mathcal{T}$ d'arité k, et deux indices i et j entre 1 et k. Le fonctionnement de l'algorithme assure que $T_V[i] < T_V[j]$ si et seulement s'il existe un élément dans le i-ème fils de V (pour l'ordre de gauche à droite) qui est inférieur à tous les éléments contenus dans le j-ème fils de V. Ces deux fils de V sont notés V_i et V_j respectivement. Comme V_i

et V_j sont des intervalles, on déduit que $T_V[i] < T_V[j]$ si et seulement si $V_i < V_j$. Par la définition 2.51, cela assure que T_V est la permutation quotient associée au nœud V. □

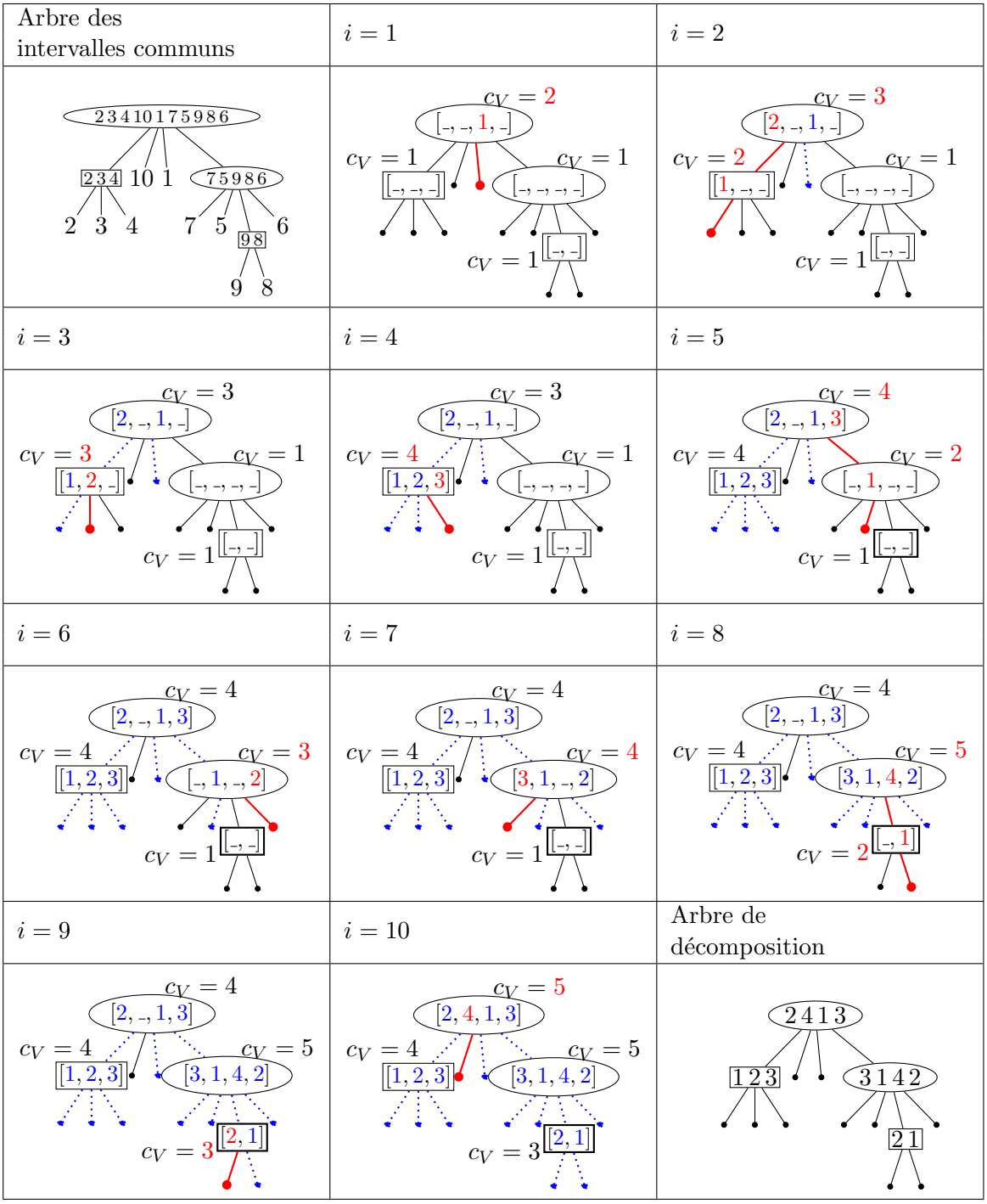


FIG. 2.11 – Déroulement de l'algorithme 2.1 pour la permutation $\sigma = 2\,3\,4\,10\,1\,7\,5\,9\,8\,6$. Les nœuds premiers de l'arbre des intervalles communs sont indiqués par une forme circulaire, et les nœuds linéaires par une forme rectangulaire. Les traits rouges et gras indiquent le calcul à l'étape courante, et les traits en pointillé bleu les calculs effectués aux étapes précédentes.

On rappelle (théorème 2.54) que l'arbre des intervalles communs d'une permutation σ, muni de l'étiquetage par des permutations quotient, est en fait l'arbre de décomposition de σ. Avec le

résultat du théorème 2.56, et le théorème 2.57 ci-dessus, on obtient donc un algorithme linéaire de calcul des arbres de décomposition. Ceci justifie le théorème 2.45 annoncé plus haut.

2.4 Produit de substitution et classes fermées par produit de substitution

On revient dans ce paragraphe aux arbres de décomposition et à la décomposition par substitution des permutations, qui est sous-jacente à la définition de ces arbres.

Parmi toutes les classes de permutations, certaines sont aisément caractérisables grâce à la décomposition par substitution. Il s'agit des classes *fermées par produit en couronne* ou *produit de substitution* (*wreath-closed* ou *substitution-closed classes* en anglais). On rencontrera ces classes à plusieurs reprises dans cette thèse, en particulier dans la partie IV. Beaucoup de propriétés générales des classes de permutations sont en effet plus faciles à démontrer sur les classes fermées par produit de substitution, et la preuve de ces propriétés dans ce cas restreint fournit une partie des outils nécessaires à sa preuve dans le cas général.

2.4.1 Définitions et propriétés

Le *produit en couronne* (*wreath product* en anglais) sur les permutations a été défini en 2002 par M. D. Atkinson et T. Stitt [AS02]. Il faut remarquer que cette définition est antérieure aux théorèmes de décomposition par substitution des permutations. Cependant, le produit en couronne est intimement lié à la substitution dans les permutations. Ce lien est explicité dès l'article [AA05] de M. H. Albert et M. D. Atkinson, où la substitution dans les permutations est définie. Suivant cette logique, le produit en couronne a reçu par la suite le nom de *produit de substitution* (*substitution product* en anglais). Dans cette thèse, on utilisera de préférence cette seconde terminologie, plus explicite.

Définition 2.58. On se donne deux ensembles $\mathcal{C}$ et $\mathcal{D}$ de permutations. Leur *produit de substitution* (aussi appelé *produit en couronne*) est l'ensemble $\mathcal{C} \wr \mathcal{D}$ des permutations σ telles qu'il existe $\pi \in \mathcal{C}$, et des permutations $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}$ appartenant toutes à $\mathcal{D}$, avec $k = |\pi|$, vérifiant que $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k)}]$.

Le produit de substitution peut ainsi être vu comme une généralisation de la dilatation à des ensembles de permutations.

Remarque 2.59. Le produit de substitution est défini sur les ensembles de permutations, en toute généralité. On s'intéressera cependant essentiellement au produit de substitution de deux classes de permutations. Évidemment, si $\mathcal{C}$ et $\mathcal{D}$ sont deux classes de permutations, alors $\mathcal{C} \wr \mathcal{D}$ est aussi une classe de permutations. La démonstration peut être trouvée dans [AS02].

Définition 2.60. Une classe de permutations $\mathcal{C}$ est *fermée par produit de substitution* si elle vérifie $\mathcal{C} \wr \mathcal{C} = \mathcal{C}$.

Lemme 2.61 ([Brib])**.** *L'intersection de classes de permutations fermées par produit de substitution est aussi une classe fermée par produit de substitution.*

Le lemme 2.61, dont la preuve est immédiate et omise ici, autorise la définition de la fermeture d'une classe par produit de substitution :

Définition 2.62. La *fermeture par produit de substitution* d'une classe $\mathcal{C}$, notée $\langle \mathcal{C} \rangle$ est la plus petite classe fermée par produit de substitution qui contient $\mathcal{C}$, c'est-à-dire l'intersection de toutes les classes fermées par produit de substitution contenant $\mathcal{C}$.

Alternativement, on peut définir $\langle \mathcal{C} \rangle$ comme $\cup_{n \geq 1} \mathcal{C}^n$, avec $\mathcal{C}^1 = \mathcal{C}$ et $\mathcal{C}^{n+1} = \mathcal{C} \wr \mathcal{C}^n$. Ces deux définitions sont équivalentes.

Exemple 2.63. La classe $\mathcal{C} = \mathcal{S}(231)$ des permutations triables par une pile n'est pas fermée par produit de substitution : par exemple $1, 12$ et 21 appartiennent à $\mathcal{C}$, alors que $231 = 21[12, 1] \notin \mathcal{C}$. La fermeture de $\mathcal{C}$ par produit de substitution est la classe $\mathcal{S}(2413, 3142)$ des permutations séparables.

La définition 2.60, le lemme 2.61, et la définition 2.62 s'énoncent de la même façon pour les ensembles $\mathcal{C}$ de permutations qui ne sont pas forcément des classes. La fermeture par produit de substitution d'un ensemble $\mathcal{C}$ de permutations, notée aussi $\langle \mathcal{C} \rangle$, est le plus petit ensemble fermé par produit de substitution qui contient $\mathcal{C}$.

Dans [AA05], M. H. Albert et M. D. Atkinson donnent une caractérisation des classes fermées par produit de substitution :

Théorème 2.64. *Une classe de permutations est fermée par produit de substitution si et seulement si tous les éléments de sa base sont des permutations simples.*

Dans l'énoncé du théorème 2.64, les permutations 1, 12, et 21 sont considérées comme des permutations simples. On note cependant que leur appartenance à la base d'une classe en rend l'étude assez immédiate.

Avec la même convention que 1, 12, et 21 sont des permutations simples, M. H. Albert et M. D. Atkinson énoncent aussi le résultat suivant, qui est une conséquence directe des théorèmes 2.33 (de décomposition par substitution) et 2.64 :

Théorème 2.65. *Une classe $\mathcal{C}$ fermée par produit de substitution est égale à la fermeture par produit de substitution de l'ensemble des permutations simples appartenant à $\mathcal{C}$.*

Ce théorème a en particulier pour conséquence [Brib] que les permutations simples appartenant à une classe sont les mêmes que celles appartenant à sa fermeture par produit de substitution.

Le théorème 2.65 peut aussi être lu avec le point de vue des arbres de décomposition. Cela revient à remplacer dans sa preuve le théorème 2.33 de décomposition de M. H. Albert et M. D. Atkinson, par le théorème 2.35, sa variante permettant de définir les arbres de décomposition.

Théorème 2.66. *Les arbres de décomposition des permutations appartenant à une classe $\mathcal{C}$ fermée par produit de substitution sont tous ceux construits sur l'ensemble de nœuds $\{\oplus, \ominus, Si(\mathcal{C})\}$, où $Si(\mathcal{C})$ désigne l'ensemble des permutations simples (de taille supérieure ou égale à 4) appartenant à la classe $\mathcal{C}$.*

Dans le cas où $12 \notin \mathcal{C}$ (resp. $21 \notin \mathcal{C}$), il faut retirer $\oplus$ (resp. $\ominus$) de l'ensemble des nœuds possibles, et dans ce cas $Si(\mathcal{C}) = \emptyset$ et $\mathcal{C}$ est l'ensemble des permutations miroirs d'une identité (resp. des permutations identités).

Hormis les deux cas particuliers mis en évidence par le théorème 2.66, une classe $\mathcal{C}$ fermée par produit de substitution peut donc être décrite par un ensemble E de permutations simples, qui est fermé pour la relation de motif (pour toute permutation $\tau \in E$, si $\sigma \preceq \tau$ et si σ est simple, alors $\sigma \in E$).

Réciproquement, étant donné un tel ensemble E, les permutations dont les arbres de décomposition sont construits sur l'ensemble de nœuds $E \cup \{\oplus, \ominus\}$ forment une classe de permutations fermée par produit de substitution.

Une classe fermée par produit de substitution est donc entièrement caractérisée par un ensemble de permutations simples fermé pour la relation de motif, qui correspond aux étiquettes possibles des nœuds premiers des permutations de la classe.

2.4.2 Exemples d'utilisation des classes fermées par produit de substitution

Grâce aux propriétés exposées au paragraphe 2.4.1, les classes de permutations fermées par produit de substitution forment une famille de classes pour laquelle certaines questions se résolvent plus facilement que dans le cadre général des classes de permutations. Cela se vérifie en particulier pour les problèmes combinatoires sur les classes de permutations qui sont abordés sous l'angle de la décomposition par substitution. Ce champ de recherche étant en cours de développement, les exemples de tels problèmes sont encore peu nombreux, mais tout à fait éloquents. On en présente quelques uns ici.

Un des résultats majeurs de l'article [AA05] fondateur de la décomposition par substitution des permutations est le théorème suivant :

Théorème 2.67. *Si une classe $\mathcal{C}$ de permutations contient un nombre fini de permutations simples, alors la base de $\mathcal{C}$ est finie.*

On reviendra dans la partie IV sur certaines conséquences de ce résultat. Ici, on s'intéresse plutôt à la structure de la démonstration du théorème 2.67.

Les classes de permutations fermées par produit de substitution, du fait que les permutations de leur base soient toutes simples (théorème 2.64), constituent un cas particulier naturel au théorème 2.67. En effet, si on considère une classe $\mathcal{C}$ de permutations fermée par produit de substitution, qui contient un nombre fini de permutations simples, alors il existe un entier N qui est la taille maximale d'une permutation simple appartenant à $\mathcal{C}$. Par le théorème 2.27, on déduit alors facilement que les permutations simples de la base de $\mathcal{C}$ sont de taille au plus $N+2$. Le théorème 2.64 affirmant que la base de $\mathcal{C}$ n'est composée que de permutations simples, ceci assure que la base de $\mathcal{C}$ est finie.

Le résultat du théorème 2.67 restreint aux classes fermées par produit de substitution est donc presque immédiat (une fois qu'on dispose des bons outils). Pour obtenir ce résultat dans le cas général, M. H. Albert et M. D. Atkinson considèrent d'abord, étant donnée une classe $\mathcal{C}$, sa fermeture $\langle\mathcal{C}\rangle$ par produit de substitution. $\langle\mathcal{C}\rangle$ étant fermée par produit de substitution, cela assure que la base de $\langle\mathcal{C}\rangle$ est finie. D'autres outils (tel le lemme d'Higman sur les beaux préordres) sont ensuite nécessaires pour en déduire que $\mathcal{C}$ a une base finie, mais le rôle fondamental que jouent les classes fermées par produit de substitution apparaît clairement dans cette preuve.

Dès l'article [AS02], le produit de substitution est utilisée à des fins énumératives : M. D. Atkinson et T. Stitt donnent une méthode pour calculer la série génératrice d'énumération de certaines classes de permutations bien décrites grâce au produit de substitution. Cette méthode est illustrée sur plusieurs exemples de classes de permutations, qui sont celles pouvant être triées par certains dispositifs composés de piles et de files d'attente.

Dans [AA05], on a aussi le résultat suivant :

Théorème 2.68. *Si une classe $\mathcal{C}$ de permutations contient un nombre fini de permutations simples, alors la série génératrice de $\mathcal{C}$ est algébrique.*

La démonstration de ce résultat commence encore par le cas, plus simple, où $\mathcal{C}$ est une classe fermée par produit de substitution.

La preuve du théorème 2.68 fait apparaître une méthodologie pour calculer les séries génératrices des classes. Dans le processus de calcul, la première étape consiste toujours à analyser la fermeture par produit de substitution de la classe. Les exemples développés dans [AA05] montrent comment calculer la série génératrice d'une classe fermée par produit de substitution, mais aussi comment trouver la série génératrice d'une classe $\mathcal{C}$ après analyse de sa fermeture par produit de substitution.

Un troisième exemple de problème qui se résout plus facilement sur les classes fermées par produit de substitution sera étudié au chapitre 9. On y décrit un algorithme polynomial qui teste, pour une classe de permutations $\mathcal{C}$ donnée par sa base finie, si $\mathcal{C}$ contient un nombre fini de permutations

simples. Le théorème 2.68 justifie d'ores et déjà l'intérêt porté à cette question, mais on y reviendra au chapitre 9.

R. Brignall, N. Ruškuc et V. Vatter ont proposé dans [BRV08] une procédure de décision pour ce problème. Dans cette procédure, il ne se dégage pas de cas particulier spécifique aux classes fermées par produit de substitution. En adaptant cette procédure pour en extraire un algorithme polynomial, on fait apparaitre que cette question est plus facile à résoudre dans le cas des classes fermées par produit de substitution, et qu'il est possible d'étendre la méthode ainsi développée pour l'adapter à l'ensemble des classes de permutations.

2.4.3 Produit de substitution et classes de base finie

On a déjà évoqué la question très importante de la base finie au paragraphe 1.4.3 page 22. Les méthodes développées pour décider si une classe de permutations a une base finie de motifs exclus dépendent évidemment de la manière dont la classe est décrite. On va voir que la donnée d'une classe comme produit de substitution de classes plus petites est un cadre adapté pour étudier la question de la base finie.

Généralisant la concaténation de [Atk99], le produit de substitution est un moyen de construire des classes de permutations à partir de classes plus petites. Il est alors naturel, étant données deux classes $\mathcal{C}$ et $\mathcal{D}$ de base finie, de chercher à faire passer la propriété de finitude de la base à la classe $\mathcal{C} \wr \mathcal{D}$.

Cette question a été envisagée dès [AS02], où M. D. Atkinson et T. Stitt démontrent en particulier que si $\mathcal{C}$ a une base finie, alors il en va de même pour $\mathcal{C} \wr \mathcal{S}(21)$. Mais ils observent aussi que la classe $\mathcal{S}(21) \wr \mathcal{S}(321654)$ n'a pas une base finie, interdisant d'espérer qu'en toute généralité, si $\mathcal{C}$ et $\mathcal{D}$ sont des classes de base finie, alors $\mathcal{C} \wr \mathcal{D}$ aussi.

C'est R. Brignall qui poursuit cette étude des relations entre classes de base finie et produit de substitution dans [Bri07], en étudiant les classes $\mathcal{D}$ telles que pour toute classe $\mathcal{C}$ ayant une base finie, $\mathcal{C} \wr \mathcal{D}$ a aussi une base finie. Les outils principaux utilisés dans ce travail sont la décomposition par substitution et les permutations en épingles.

R. Brignall met en évidence une condition suffisante sur $\mathcal{D}$ pour que la propriété ci-dessus soit satisfaite : il suffit que $\mathcal{D}$ ait une base finie et ne contienne pas de *pin sequence* arbitrairement longue. La preuve de ce résultat donne aussi une méthode pour construire la base de $\mathcal{C} \wr \mathcal{D}$ à partir de celles de $\mathcal{C}$ et $\mathcal{D}$.

En outre, [Bri07] résout complètement la question de savoir si pour toute classe $\mathcal{C}$ ayant une base finie, $\mathcal{C} \wr \mathcal{D}$ a aussi une base finie dans les cas où $\mathcal{D} = \mathcal{S}(\sigma)$ pour $|\sigma| \leq 3$ et $\mathcal{D} = \mathcal{S}(\sigma, \tau)$ pour $|\sigma| \leq 4$ et $|\tau| \leq 4$.

On mentionne enfin un dernier problème, peut-être encore plus naturel, concernant les classes de bases finies et le produit de substitution : étant donnée une classe $\mathcal{C} = \mathcal{S}(\mathcal{B})$ de base finie, la fermeture par produit en couronne $\langle \mathcal{C} \rangle$ de $\mathcal{C}$ a-t-elle une base finie ?

La réponse à cette question ne peut être unique, puisque par exemple $\langle \mathcal{S}(231) \rangle$ a une base finie, mais pas $\langle \mathcal{S}(4321) \rangle$ [Brib]. La question qui s'ouvre naturellement est alors celle de trouver des critères sur la base $\mathcal{B}$ permettant de décider si la fermeture par produit de substitution de $\mathcal{S}(\mathcal{B})$ a une base finie. R. Brignall indique dans [Brib] qu'un travail en cours de M. D. Atkinson, N. Ruškuc et R. Smith étudie ce problème [ARS].

2.5 Forme en moyenne des arbres de décomposition

On conclut ce chapitre consacré à la décomposition par substitution des permutations par une analyse de la forme "en moyenne" des arbres de décomposition des permutations.

Le théorème 2.26 assurant qu'une permutation de taille n est simple avec probabilité $1/e^2$ lorsque $n \to \infty$, on peut d'abord affirmer qu'avec cette même probabilité $1/e^2$, lorsque $n \to \infty$,

un arbre de décomposition de taille n a la forme d'une racine première n'ayant pour fils que des feuilles.

Ce paragraphe raffine cette analyse de la forme des arbres de décomposition, et présente un résultat nouveau, publié dans l'article [BCMR], dans le cadre de l'analyse en moyenne des scénarios calculés par un algorithme de tri parfait par renversement de permutations signées. On reviendra sur cette étude de complexité dans le chapitre 7. On présente cependant dès maintenant le résultat sur la forme en moyenne des arbres de décomposition, intéressant en soi.

Au préalable, on pose la définition suivante :

Définition 2.69. Une *cerise* dans un arbre de décomposition est un nœud de degré 2 dont les deux fils sont des feuilles.

D'après la remarque 2.25, une cerise est un nœud linéaire.

Théorème 2.70. *Asymptotiquement, avec probabilité* 1*, une permutation σ de taille n tirée uniformément au hasard a un arbre de décomposition dont la racine est un nœud premier dont tous les fils sont soit des feuilles soit des cerises. En outre, la probabilité que son arbre de décomposition soit de cette forme et compte exactement k cerises est asymptotiquement $\frac{2^k}{e^2 k!}$.*

On peut déduire immédiatement du théorème 2.70 que, dans le cas où l'arbre de décomposition est bien de la forme décrite par ce théorème, le nombre moyen de cerises sous la racine est 2.

La preuve du théorème 2.70 repose sur le lemme 2.72 qui généralise le résultat suivant :

Lemme 2.71. [AAK03, Lemme 7] *On note $p_{n,k}$ le nombre de permutations de taille n qui contiennent un intervalle minimal de taille k. Pour toute valeur entière c, on a*

$$\sum_{k=c+2}^{n-c} \frac{p_{n,k}}{n!} = O(n^{-c}).$$

On peut en fait s'affranchir de l'hypothèse de minimalité :

Lemme 2.72. *On note $p'_{n,k}$ le nombre de permutations de taille n qui contiennent un intervalle I de taille k. Pour toute valeur entière c, on a*

$$\sum_{k=c+2}^{n-c} \frac{p'_{n,k}}{n!} = O(n^{-c}).$$

Démonstration. La preuve du lemme 2.72 ressemble beaucoup à celle du lemme 2.71 dans l'article [AAK03]. On commence par remarquer que $p'_{n,k} \leq (n-k+1)k!(n-k+1)!$. En effet, le membre droit compte le nombre de permutations quotient qui correspondent à I (soit $k!$), les valeurs possibles de l'élément minimal de I (en nombre $n-k+1$), et la structure du reste de la permutation avec un élément en plus (celui qui est dilaté par I). Ainsi, seuls les termes extrémaux de la somme peuvent être de l'ordre de $\mathcal{O}(n^{-c})$, et les termes restant sont tous d'ordre $\mathcal{O}(n^{-c-1})$. Comme il y a moins de n termes dans la somme, on conclut avec le résultat annoncé. □

Démonstration du théorème 2.70. Le lemme 2.72 appliqué pour $c = 1$ énonce que la proportion de permutations qui contiennent des intervalles de taille au moins 3 et au plus $n-1$ est $O(n^{-1})$. Remarquons à présent que les permutations dont les intervalles sont tous de taille $1, 2$ ou n sont exactement les permutations dont l'arbre de décomposition est constitué d'une racine première dont tous les fils sont soit des feuilles soit des cerises.

D'autre part, en notant s_n le nombre de permutations simples de taille n, le nombre de permutations dont l'arbre de décomposition a une racine première dont k fils sont des cerises (les autres étant des feuilles) est $s_{n-k}\binom{n-k}{k}2^k$. D'après le théorème 2.26, une estimation asymptotique de ce nombre est $\frac{n!2^k}{e^2 k!}$, ce qui achève la démonstration du théorème 2.70. □

On peut comparer le résultat du théorème 2.70, qui concerne les arbres de décomposition des permutations, à un résultat analogue dans le cadre des graphes (et même dans un cadre algébrique plus général). Il est bien connu (voir par exemple [MR84]) qu'un graphe a une probabilité 1 d'être premier, c'est-à-dire d'avoir un arbre de décomposition modulaire dont la racine est un nœud premier dont tous les fils sont des feuilles. Notre résultat sur les arbres de décomposition s'accorde bien avec ce qui est connu des arbres de décomposition modulaire, tout en soulignant les différences qui existent entre ces deux objets.

Deuxième partie

Recherche de motifs dans les permutations

« From the moment of my birth
To the instant of my death,
There are patterns I must follow
Just as I must breathe each breath. »

Simon & Garfunkel, *Patterns*

Généralités sur la recherche de sous-structure dans les objets structurés

Dans cette partie, on s'intéresse à des problèmes algorithmiques de recherche de motif dans les permutations. Ce travail se place dans le cadre plus général de la recherche de "sous-structure" dans des objets structurés, comme les mots, ou les graphes. Les permutations se situent en quelque sorte à mi-chemin entre ces deux exemples. Comme les mots, elles ont une structure linéaire, mais qui est enrichie par le fait que les lettres en sont totalement ordonnées. Mais les permutations peuvent aussi être vues à travers leurs graphes de permutation, qui forment un sous-ensemble de l'ensemble de tous les graphes. Certains problèmes difficiles sur les graphes en général se résolvent plus facilement sur les graphes de permutation, mais d'autres gardent toute leur complexité malgré cette restriction.

On envisagera deux types de problèmes : la recherche d'occurrence d'un motif dans une permutation, et la recherche de plus grand motif commun à un ensemble de permutations. Dans le contexte des mots, ces problèmes sont à rapprocher respectivement de la recherche d'occurrence d'un sous-mot et de la recherche d'un plus grand sous-mot commun. On parle bien ici de sous-mot, et non de facteur, bien que la recherche de facteur ait été plus étudiée dans la littérature d'algorithmique du texte. Dans le cas des graphes, la notion de sous-structure qui correspond le mieux aux motifs dans les permutations est celle de sous-graphe induit, et les problèmes analogues à ceux qui nous intéressent sont donc l'isomorphisme d'un sous-graphe induit et la recherche de plus grand sous-graphe induit commun.

Du point de vue de la complexité, les problèmes de recherche d'occurrence d'un sous-mot, et de recherche de plus grand sous-mot commun à deux mots peuvent être résolus en temps polynomial. Dans l'ouvrage [CHL01], un algorithme de programmation dynamique pour le problème de la recherche de plus grand sous-mot commun est donné, ainsi que des améliorations permettant d'atteindre une complexité linéaire en espace. Au contraire, dans le cadre des graphes, la recherche de plus grand sous-graphe induit commun est un problème NP-difficile, et même difficilement approximable [Kan92], et l'isomorphisme de sous-graphe induit est NP-complet [Dam91, par exemple].

Nous verrons que pour les permutations, la recherche d'un motif dans une permutation, et la recherche de plus grand motif commun à deux permutations, sont des problèmes NP-difficiles. On s'intéressera à des restrictions de ces problèmes qui admettent des solutions polynomiales, en particulier en considérant des motifs ou des permutations dans la classe des permutations séparables. On verra par exemple que la recherche d'un motif séparable dans une permutation est polynomiale. Ceci contraste avec le résultat de [Dam91] énonçant que l'isomorphisme de sous-graphe induit est NP-difficile pour les graphes sans P_4 induit (ou cographes), qui sont les graphes de permutations associés aux permutations séparables. Ces deux résultats ne sont pourtant pas en contradiction, car les permutations ne sont pas en bijection avec les graphes de permutations, mais avec les graphes de permutations *étiquetés*.

Le chapitre 3 présente un état de l'art sur le problème de la recherche d'une occurrence d'un motif dans une permutation. On reviendra sur le caractère NP-complet de ce problème, et on examinera quelques restrictions qui le rendent polynomial. Parmi les résultats de la littérature, on présentera ceux concernant la recherche d'une occurrence du motif identité $12\dots k$ dans une permutation, et les améliorations de l'algorithme naïf de recherche de motif qui conduisent à des algorithmes efficaces pour la recherche de tout motif de taille 4. La restriction qui va nous intéresser tout particulièrement est la recherche d'un motif séparable. On définit donc la classe des permutations séparables, et la structure d'arbre de séparation qui leur est associée. Ces arbres de séparation vont jouer un rôle clé dans les algorithmes développés. Enfin, on décrit deux algorithmes polynomiaux de recherche d'occurrence d'un motif séparable dans une permutation. Ces algorithmes sont dus à P. Bose, J. F. Buss et A. Lubiw d'une part [BBL98], et à L. Ibarra [Iba97] d'autre part. Ils vont constituer une base de travail pour la conception des algorithmes dans les chapitres suivants.

Le chapitre 4 considère le problème de recherche de plus grand motif commun à deux permutations. Ce problème est NP-difficile, de par la NP-complétude de la recherche d'occurrence de motif. Un premier cas polynomial a été mis en évidence dans [MR06], pour la recherche de plus grand motif commun à deux permutations triables par pile. L'algorithme proposé repose sur une bijection entre les permutations triables par pile et les arbres, et sur un algorithme de recherche de plus grand sous-arbre commun à deux arbres. On proposera un algorithme polynomial pour la recherche de plus grand motif commun entre deux permutations, dont une est séparable. Les permutations triables par pile étant séparables, ceci généralise le résultat de [MR06]. La conception de l'algorithme repose cependant sur des outils différents, à savoir les arbres de séparation et la programmation dynamique, en s'appuyant sur les idées de P. Bose, J. F. Buss et A. Lubiw dans [BBL98]. Enfin, on verra que cet algorithme peut être adapté pour calculer le plus grand motif commun à deux permutations quelconques, en utilisant des arbres de décomposition développés à la place des arbres de séparation. Même si on perd bien sûr le caractère polynomial de l'algorithme, la complexité obtenue n'est pas exponentielle en la taille des permutations en entrée, mais seulement en un paramètre plus petit : l'arité maximale d'un nœud premier dans leur arbre de décomposition. J'ai commencé ces travaux pendant mon stage de Master [Bou06], et ils ont fait l'objet de la publication [BR06].

Le chapitre 5 envisage une généralisation du problème précédent : la recherche de plus grand motif commun à un ensemble de permutations. Là encore, cette question étant NP-difficile, on se restreint à un sous-problème, mais en imposant cette fois des conditions sur le motif cherché plutôt que sur les permutations en entrée. On s'intéresse à la recherche de plus grand motif séparable commun à un ensemble de permutations, c'est-à-dire d'un motif séparable ayant une occurrence dans chacune des permutations de l'ensemble, et qui soit le plus long pour ce critère. On décrira un algorithme polynomial pour la recherche de plus grand motif séparable commun à un ensemble de k permutations (k étant fixé), et on verra que quand le cardinal de l'ensemble de permutations en entrée n'est pas connue à l'avance, le problème est NP-difficile. Il est naturel alors de se demander si le plus grand motif séparable commun à un ensemble de permutations est une bonne approximation de leur plus grand motif commun (sans contrainte de séparabilité). On verra qu'en contraignant le motif à appartenir à une classe de permutations (comme les séparables, entre autres), on ne peut pas faire mieux qu'obtenir une approximation en $\sqrt{Opt}$, Opt désignant la taille du plus grand motif commun non contraint. Ces résultats ont été publiés dans [BRV07].

Chapitre 3

État de l'art : recherche de motifs dans les permutations

Sommaire

3.1 Le problème de recherche de motif dans les permutations 68
3.1.1 Présentation du problème 68
3.1.2 Caractère NP-complet du problème général 68
3.1.3 Exemple : recherche de sous-séquence croissante maximale 69
3.1.4 Méthodologie pour des algorithmes efficaces ; application à la recherche de motifs de taille 4 70
3.2 Permutations séparables, arbres de séparation, et généralisations . 71
3.2.1 La classe des permutations séparables et leurs arbres de séparation . . 71
3.2.2 Arbres de séparation, arbres de Schröder, et arbres de décomposition . 73
3.3 Algorithme de recherche d'un motif séparable dans une permutation 76
3.3.1 Algorithme de Bose, Buss et Lubiw 76
3.3.2 Algorithme d'Ibarra 78

3.1 Le problème de recherche de motif dans les permutations

3.1.1 Présentation du problème

Le problème considéré est celui de la recherche d'une occurrence d'un motif dans une permutation. On rappelle (voir définition 1.14 page 15) qu'un motif $\sigma \in \mathcal{S}_k$ a une occurrence dans une permutation $\tau \in \mathcal{S}_n$ s'il existe des entiers $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ tels que la sous-séquence $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$ extraite de τ est isomorphe en ordre à σ. Une occurrence de σ dans τ est alors la sous-séquence $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$.

Savoir décider si un motif apparaît dans une permutation est une question essentielle. En particulier, une procédure de décision d'occurrence d'un motif dans une permutation fournit directement une procédure de test d'appartenance d'une permutation à une classe, pour toute classe de permutations donnée par sa base de motifs exclus, et dans le cas (souvent observé dans la littérature) où cette base est finie.

On donc considère le problème de décision suivant :

Problème 3.1 (Recherche de motif)**.**
Données : Un motif σ de taille k et une permutation τ de taille n.
Sortie : Un booléen qui indique si σ a une occurrence dans τ ou non.

On sera amené à considérer des variantes de ce problème, en demandant en sortie d'exhiber une occurrence de σ dans τ s'il en existe une, de compter le nombre de telles occurrences, de les calculer toutes...

On peut aussi parfois souhaiter que le motif σ ne fasse pas partie des données du problème, et soit plutôt une donnée extérieure. La mesure de la complexité des algorithmes résolvant le problème se fait alors seulement par rapport à la taille n de τ.

Évidemment, il existe un algorithme naïf pour résoudre le problème 3.1, qui consiste à examiner toutes les sous-séquences de τ de taille k, pour tester si elles sont isomorphes en ordre à σ. Une sous-séquence $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$ de τ étant fixée, on peut tester en temps linéaire si elle est isomorphe en ordre à σ. Il suffit pour cela de calculer l'inverse σ^{-1} de σ, et de tester si la séquence réordonnée $\tau_{i_{\sigma^{-1}(1)}}\tau_{i_{\sigma^{-1}(2)}}\ldots\tau_{i_{\sigma^{-1}(k)}}$ est une séquence croissante. Cet algorithme naïf a donc une complexité de l'ordre de $\mathcal{O}(kn^k)$. Dans le cas où σ est une donnée extérieure, cet algorithme a donc une complexité polynomiale.

Pour revenir au test d'appartenance d'une permutation à une classe, cet algorithme naïf permet de tester en temps polynomial qu'une permutation appartient à une classe donnée par sa base finie, à condition de considérer que tous les motifs exclus sont des données extérieures au problème. Il est souvent possible pour une classe donnée de trouver des algorithmes de test d'appartenance *ad hoc* qui sont beaucoup plus efficaces que cette procédure naïve et peu satisfaisante.

Si l'on ne suppose plus que le motif σ est une donnée extérieure, mais qu'il fait au contraire partie des données du problème, alors l'algorithme naïf résolvant le problème 3.1 est exponentiel. Ce n'est pas surprenant, car comme on va le voir, le problème 3.1 est NP-complet.

3.1.2 Caractère NP-complet du problème général

Dans [BBL98], P. Bose, J. F. Buss et A. Lubiw étudient la complexité du problème 3.1 et démontrent que :

Théorème 3.2. *Étant donnés un motif σ et une permutation τ, le problème de savoir si σ possède une occurrence dans τ est NP-complet.*

Esquisse de démonstration. Il est facile de voir que le problème 3.1 est dans la classe NP : un certificat vérifiable polynomialement est par exemple la séquence de positions $i_1, i_2, \ldots, i_k$ telle que $\tau_{i_1}\tau_{i_2}\ldots\tau_{i_k}$ soit une occurrence de σ.

Pour démontrer que ce problème est NP-complet, [BBL98] donne une réduction polynomiale de $3SAT$ à ce problème. On explique ici seulement les lignes directrices de cette réduction.

On considère une formule φ instance du problème $3SAT$, c'est-à-dire une conjonction de disjonction de trois littéraux. Un littéral est soit une variable soit la négation d'une variable, et chaque disjonction dans la formule φ s'appelle une clause. L'idée de la réduction est de coder la formule φ dans un motif σ, et de coder dans une permutation τ les valuations des variables de φ qui rendent φ vraie, de telle façon que σ ait une occurrence dans τ si et seulement si φ est satisfaisable.

Le motif σ est composé de plusieurs sous-séquences mises bout à bout, la première codant les variables de φ et les suivantes chacune des clauses de φ. La permutation τ commence par une sous-séquence codant les deux valeurs possibles de chacune des variables de φ, et se poursuit par un fragment pour chacune des clauses de φ. Pour chaque clause, ce fragment est divisé en sept parties, qui codent chacune une valuation des trois variables de la clause qui la rendent vraie.

Les différentes parties du codage sont protégées par des gardes, de sorte que lorsque σ a une occurrence dans τ, celle-ci envoie le fragment de σ codant les variables sur le fragment de τ codant les variables, et de même pour les fragments de σ et τ codant chaque clause de φ. Une fois cette condition assurée par le codage, il n'est pas difficile de voir que toute occurrence de σ dans τ correspond à une valuation des variables qui rend φ vraie, et réciproquement. □

Sachant que le problème de recherche de motif dans une permutation est NP-complet, on peut se demander ce que serait un algorithme relativement efficace pour résoudre ce problème. L'algorithme naïf du paragraphe 3.1.1 est de complexité $\mathcal{O}(kn^k)$. Sa partie exponentielle fait donc intervenir à la fois n et k. On pourrait attendre d'un "bon" algorithme exponentiel qu'il soit exponentiel en k et polynomial en n [b]. La question de l'existence d'un tel algorithme a suscité de l'intérêt ces dernières années (voir par exemple [GV] dans le cas où les motifs évitent 321), mais le problème reste à ma connaissance ouvert à ce jour.

3.1.3 Exemple : recherche de sous-séquence croissante maximale

On étudie ici un cas particulier du problème 3.1, où le motif recherché est la permutation identité $\sigma = 12\dots k$. Un problème strictement équivalent est la recherche du miroir de l'identité $k\dots 21$. Un résultat de M.-S. Chang et F.-H. Wang dans [CW92] permet de résoudre ce problème en temps $\mathcal{O}(n \log\log n)$, indépendamment de k. La taille k du motif identité cherché peut donc indifféremment faire partie des données du problème ou non.

L'idée est de chercher la plus longue sous-séquence croissante (ou décroissante, dans le cas du motif miroir de l'identité) dans la permutation τ. Ainsi, il suffira de tester si la longueur de cette plus longue sous-séquence croissante est supérieure ou égale à k pour décider si σ à une occurrence dans τ. Si tel est le cas, une occurrence de σ dans τ est obtenue par exemple en prenant les k premiers éléments de la plus longue sous-séquence croissante dans τ.

Une modification de l'algorithme de Robinson-Schensted [Sch61] (consistant à ne conserver que la première ligne du tableau de Young construit) permet d'extraire une plus longue sous-séquence croissante d'une permutation de taille n en temps $\mathcal{O}(n \log n)$, ce qui démontre déjà que le problème de recherche d'une occurrence de $\sigma = 12\dots k$ dans une permutation est résoluble en temps polynomial. Mais on peut proposer un algorithme encore plus efficace, de complexité $\mathcal{O}(n \log\log n)$, qui résout ce problème. C'est ce qu'ont décrit M.-S. Chang et F.-H. Wang dans [CW92], en utilisant des outils d'algorithmique des graphes.

À toute permutation τ, on peut associer un graphe de permutation, où tout sommet du graphe représente un élément de la permutation, et où les arêtes du graphe codent les inversions, c'est-à-dire les paires d'éléments (i, j) telles que $i < j$ et $\tau_i^{-1} > \tau_j^{-1}$. En d'autres termes :

[b] Un tel algorithme est dit *FPT* (pour *fixed parameter tractable*) par rapport au paramètre k désignant la taille du motif.

Définition 3.3. Le *graphe de permutation* associé à une permutation τ de taille n est un graphe $G = (V, E)$ où l'ensemble des sommets V est $\{1, 2, \ldots, n\}$ et l'ensemble des arêtes est $E = \{\{i, j\} : i < j$ et i apparaît après j dans $\tau\}$.

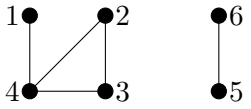


FIG. 3.1 – Le graphe de permutation associé à $\tau = 4\,1\,3\,2\,6\,5$.

Une plus longue sous-séquence croissante (resp. décroissante) dans une permutation τ correspond à un stable (resp. une clique – voir définitions page 34) de taille maximale dans le graphe de permutation associé à τ. Le problème de recherche d'un stable (ou d'une clique) de taille maximale dans un graphe sans contrainte est NP-difficile. Mais pour certaines classes de graphes, le problème du stable de taille maximale est polynomial. En particulier pour les graphes de permutations, quand ils sont décrits sous la forme d'un modèle d'intervalle, il existe un algorithme calculant en temps $\mathcal{O}(n \log\log n)$ un stable de taille maximale. C'est par cette réduction que M.-S. Chang et F.-H. Wang démontrent que :

Théorème 3.4 ([CW92])**.** *Pour le problème de recherche du motif $\sigma = 12\ldots k$ dans une permutation de taille n, il existe un algorithme en $\mathcal{O}(n \log\log n)$.*

3.1.4 Méthodologie pour des algorithmes efficaces ; application à la recherche de motifs de taille 4

Dans l'article [AAAH01], M. H. Albert, R. E. L. Aldred, M.D. Atkinson et D. A. Holton proposent une méthodologie générale qui raffine l'algorithme naïf résolvant le problème 3.1, en supposant que le motif σ recherché est une donnée extérieure. L'idée est de ne pas examiner toutes les k-uplets d'éléments de τ individuellement, mais de plutôt les regrouper pour effectuer certains tests en commun. Ceci va demander un prétraitement du motif σ, qui a le défaut d'être exponentiel, mais qui n'est effectué qu'une fois et qui permet d'accélérer les tests d'occurrence de σ dans les permutations τ.

L'algorithme est donc composé de deux étapes : le prétraitement de σ, effectué en temps $\mathcal{O}(2^k)$, et le corps de l'algorithme, qui teste si σ possède une occurrence dans τ.

Le prétraitement de σ consiste à trouver une séquence Σ de motifs $1 = \sigma^1 \preceq \sigma^2 \preceq \ldots \preceq \sigma^k = \sigma$, où chaque σ^i est de taille i, de sorte à permettre au mieux les regroupements de tests de k-uplets d'éléments de τ susceptibles d'être une occurrence de σ. Toute séquence Σ est acceptable, mais pour proposer la meilleure amélioration possible de la deuxième phase de l'algorithme, il faut trouver la séquence Σ qui minimise un paramètre $c(\sigma)$, qui conditionne la complexité de la deuxième phase.

Une fois la séquence Σ précalculée, la deuxième phase de l'algorithme cherche des occurrences de chaque σ_i dans τ, par valeurs croissantes de i. Les éléments de τ utilisés dans une occurrence de σ_i déterminent comment cette occurrence peut être étendue en une occurrence de σ_{i+1}. Pour toutes les occurrences de σ_i qui autorisent les mêmes extensions, les tests sur τ sont factorisés. Ainsi, l'algorithme étend les occurrences de chaque σ_i en occurrences de σ_{i+1}, et classe en même temps ces occurrences en groupes ayant les mêmes extensions, puis raffine ce classement à chaque étape. Cette deuxième étape de l'algorithme s'effectue en temps $\mathcal{O}(n^{1+c(\sigma)} \log n)$.

La valeur de $c(\sigma)$ n'est pas connue exactement. On peut mettre en évidence une borne supérieure sur $c(\sigma)$ qui est linéaire en k, et obtenir ainsi une complexité qui rejoint celle de l'algorithme naïf. Cette borne peut être raffinée jusqu'à $\frac{2k}{3}$, mais expérimentalement, sur quelques milliers de tests, la valeur maximale observée pour $c(\sigma)$ est $1 + \frac{k}{2}$. En prouvant cette borne supérieure sur $c(\sigma)$, on

aurait donc un algorithme pour le problème 3.1 dont la complexité serait en racine carrée de celle de l'algorithme naïf.

On peut remarquer qu'une méthodologie similaire a été adoptée de manière indépendante par S. Ahal et Y. Rabinovich dans [AR08], aboutissant à un algorithme de complexité comparable, à savoir $\mathcal{O}(n^{c(\sigma)})$ pour une autre fonction c mesurant la complexité de σ, qui vérifie ici $c(\sigma) \leq 0,47k + o(k)$.

La méthodologie développée pour concevoir l'algorithme en deux phases ci-dessus fournit donc une amélioration par rapport à l'algorithme naïf, quel que soit le motif σ. Mais pour un motif σ fixé, il est possible d'étudier plus précisément la séquence Σ optimale, ce qui conduit parfois à une borne supérieure plus précise pour $c(\sigma)$. En utilisant aussi des structures de données plus adaptées, les auteurs de [AAAH01] expliquent comment leur travail permet d'aboutir au résultat suivant lorsque le motif dont on cherche une occurrence est de taille 4 :

Théorème 3.5. *Pour tout motif σ de taille 4, il existe un algorithme qui recherche une occurrence de σ dans une permutation de taille n en temps $\mathcal{O}(n \log n)$.*

On signale au passage que pour tout motif σ de taille 3, la recherche d'une occurrence de σ dans une permutation de taille n peut être faite en temps $\mathcal{O}(n)$: par un algorithme de tri par pile si $\sigma \in \{231, 132, 213, 312\}$ ou par une simplification de l'algorithme de Robinson-Schensted si $\sigma \in \{123, 321\}$.

3.2 Permutations séparables, arbres de séparation, et généralisations

Le problème général de recherche d'un motif dans une permutation étant NP-difficile, un angle d'attaque pour chercher des algorithmes efficaces est de restreindre ce problème à des classes de permutations. Dès l'article [BBL98], où le caractère NP-complet du problème est démontré, des restrictions de ce type sont envisagées. Les auteurs considèrent en particulier la classe des permutations *séparables*, définies un an plus tôt dans [Iba97]. C'est aussi cette classe de permutations qui sera au centre des travaux effectués pendant ma thèse sur la recherche de plus grand motif commun à deux (ou à un ensemble de) permutations.

3.2.1 La classe des permutations séparables et leurs arbres de séparation

Les permutations séparables ont été largement étudiées ces dix dernières années, dans différents contextes, aussi bien algorithmiques que combinatoires [Iba97, BBL98, BXHP05, BBCP07, BCMR08]. Par conséquent, de nombreuses définitions et caractérisations de ces permutations ont été proposées, indépendamment les unes des autres. On choisit une définition qui a l'avantage de mettre en évidence le fait que les permutations séparables forment une classe de permutations, et on donne quelques caractérisations possibles des permutations séparables :

Définition 3.6. Les permutations séparables sont celles évitant les motifs 2 4 1 3 et 3 1 4 2. On notera $\mathcal{S}ep = \mathcal{S}(2\,4\,1\,3, 3\,1\,4\,2)$ la classe des permutations séparables.

Théorème 3.7. *Les propositions suivantes sont équivalentes :*

(*i*) *σ évite les motifs 2 4 1 3 et 3 1 4 2,*

(*ii*) *σ possède un arbre de séparation,*

(*iii*) *l'arbre de décomposition de σ n'a pas de nœud premier,*

(*iv*) *le graphe de permutation de σ ne contient pas de P_4 induit.*

Elles constituent quatre définitions possibles des permutations séparables.

Parmi les travaux qui utilisent ces différentes définitions des permutations séparables, on peut citer [BBL98] pour la définition à partir des graphes de permutations, [BBL98, Iba97] pour les définitions en termes de motifs exclus et d'arbres de séparation, [BBCP07, BXHP05, BCMR08] pour la définition à travers les arbres de décomposition, ...

Les arbres de décompositions ont été définis page 44, et les graphes de permutation page 70, il reste donc à définir les arbres de séparation avant de démontrer le théorème 3.7.

Définition 3.8. Un *arbre de séparation* d'une permutation σ de taille n est un arbre binaire plan dont les feuilles sont, dans l'ordre de lecture de gauche à droite, σ_1, σ_2, ..., σ_n, et tel que pour tout noeud V de l'arbre, l'ensemble des feuilles dans le sous-arbre de racine V forme un intervalle.

À chaque noeud d'un arbre de séparation, on associe donc un intervalle. La définition d'un arbre de séparation implique que chaque noeud interne possède l'un des deux types suivants :

- noeud positif $\oplus$: si l'intervalle de son fils gauche contient des valeurs inférieures à celles de l'intervalle de son fils droit,
- noeud négatif $\ominus$: si l'intervalle de son fils gauche contient des valeurs supérieures à celles de l'intervalle de son fils droit,

Remarquons que pour une permutation donnée, il peut exister plusieurs arbres de séparation. La figure 3.2 représente deux arbres de séparation possibles pour la permutation $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$.

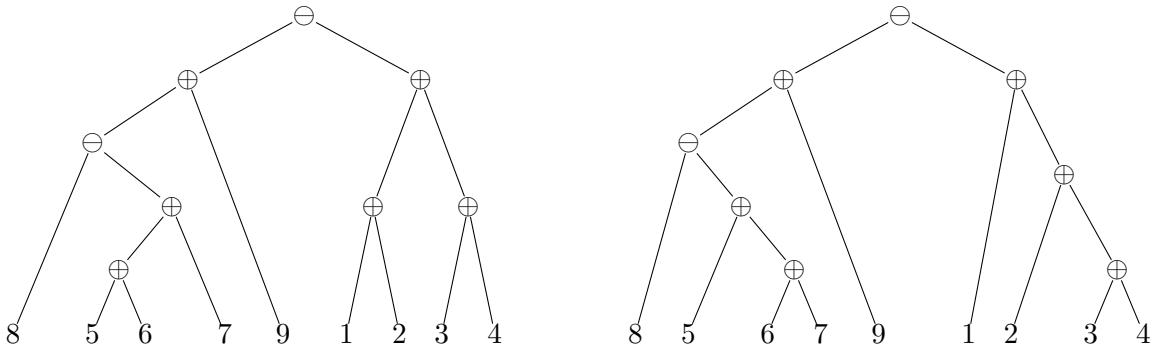


FIG. 3.2 – Deux arbres de séparation pour $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$.

On peut aussi remarquer que les étiquettes des feuilles d'un arbre de séparation peuvent être déduites des signes des nœuds internes, suivant la règle que les étiquettes des feuilles dans le sous-arbre gauche d'un nœud $\oplus$ (resp. $\ominus$) sont inférieures (resp. supérieures) à celles du sous-arbre droit.

Les arbres de séparation ont été définis indépendamment des arbres de décomposition, mais en sont de proches cousins. Nous reviendrons dans le paragraphe 3.2.2 aux points communs et aux différences entre ces deux objets.

La définition d'un graphe de permutation a été donnée plus haut (définition 3.3 page 70). On précise ce qu'est un P_4 induit dans ce graphe :

Définition 3.9. Un P_4 *induit* dans un graphe G est un chemin induit sur quatre sommets du graphe, c'est-à-dire un ensemble de quatre sommets tel que les arêtes de G induites par ces sommets forment un chemin de taille 4 sans corde •—•—•—•.

Démonstration du théorème 3.7. L'équivalence entre les propositions (i) et (ii) a été démontrée indépendamment dans [BBL98] et dans [EHtPR98]. J'ai démontré pendant mon stage de Master que les propriétés (i) et (iv) d'une part, et (ii) et (iii) d'autre part sont équivalentes. Les preuves peuvent être trouvées dans [Bou06], propriété 2.7 et théorème 5.6 respectivement. □

Parmi les résultats connus sur les permutations séparables, les suivants nous seront utiles. Les preuves en sont omises ici mais figurent dans mon rapport de stage de Master [Bou06].

D'un point de vue algorithmique, on s'intéresse au problème de décider si une permutation donnée est séparable, et dans l'affirmative, à la complexité du calcul d'un arbre de séparation associé. Sans donner les détails des algorithmes, le théorème 3.10 répond à ces questions :

Théorème 3.10. *[BBL98] Il existe un algorithme en temps linéaire qui, sur l'entrée σ, teste si σ est séparable, et le cas échéant calcule un arbre de séparation pour σ.*

Le fonctionnement de cet algorithme est décrit dans [BBL98] et formalisé dans [Bou06].

On peut aussi s'interroger sur l'énumération des permutations séparables :

Théorème 3.11. *[Wes95, EHtPR98] Le nombre de permutations séparables de taille n est Sch_{n-1}, (Sch_n) désignant la séquence des nombres de Schröder.*

Les nombres de Schröder correspondent à la séquence A006318 dans [Slo07]. La série génératrice qui leur est associée est

$$S(x) = \sum_{n\geq 0} Sch_n x^n = \frac{1 - x - \sqrt{x^2 - 6x + 1}}{2x},$$

le coefficient Sch_n est asymptotiquement de l'ordre de $(3 + 2\sqrt{2})^n$, et les premiers termes de la séquence $(Sch_n)_{n\geq 0}$ sont

$$1, 2, 6, 22, 90, 394, 1806, 8558, 41586, 206098.$$

Les nombres de Schröder (Sch_n) ainsi définis sont parfois appelés *grands* nombres de Schröder par opposition aux *petits* nombres de Schröder. Les petits nombres de Schröder (r_n) sont reliés aux nombres de Schröder standards (Sch_n) par l'identité $2r_n = Sch_n$ pour tout $n \geq 1$. Leur séquence porte la référence A001003 dans [Slo07]. Parmi les objets énumérés par les petits nombres de Schröder, certains sont directement reliés aux permutations séparables, via leurs arbres de séparation : il s'agit des *arbres de Schröder*.

Définition 3.12. Un *arbre de Schröder* est un arbre plan où tout nœud interne a une arité supérieure ou égale à 2.

Le lien entre arbres de Schröder et arbres de séparation est explicité au paragraphe 3.2.2.

3.2.2 Arbres de séparation, arbres de Schröder, et arbres de décomposition

Dans ce paragraphe, on propose des variations autour des arbres de séparation (des permutations séparables) et des arbres de décomposition (des permutations en général), qui expliquent les relations entre ces objets, et dont l'intérêt deviendra manifeste dans les paragraphes 3.3, 4.2, et 4.3.

On part des arbres de séparation des permutations séparables, qui sont bien adaptés à une utilisation algorithmique, mais qui ne sont pas définis avec unicité pour chaque permutation. On en propose une variante, les *arbres de séparation contractés* qui, s'ils sont moins manipulables algorithmiquement, sont en bijection avec les permutations séparables. On verra que ce type d'arbre correspond exactement aux arbres de décomposition restreints aux permutations séparables. Cela conduit tout naturellement à proposer une transformation des arbres de décomposition des permutations en général, vers des *arbres de décomposition développés*, qui sont une extension à toutes les permutations des arbres de séparation, plus facilement utilisables algorithmiquement que les arbres de décomposition standards.

Les arbres de séparation tels que définis par la définition 3.8 sont des arbres *binaires*, ce qui est bien adapté à des manipulations algorithmiques, comme nous le verrons dès le paragraphe 3.3.

Cependant, il n'y a pas unicité de la représentation d'une permutation séparable par un arbre de séparation. Pour pallier ce problème, on peut définir les *arbres de séparation contractés*. Un unique arbre de ce type est associé à chaque permutation séparable, mais on perd le caractère binaire. Les arbres de séparation contractés seront donc moins adaptés aux considérations algorithmiques, mais plus pratiques d'un point de vue combinatoire.

Définition 3.13. L'*arbre de séparation contracté* d'une permutation séparable σ est obtenu à partir d'un de ses arbres de séparation en contractant les arêtes qui relient deux nœuds de même signe, jusqu'à ce qu'il n'y ait plus aucune contraction possible.

Pour justifier la définition 3.13, il faut remarquer que, pour toute permutation séparable σ, quel que soit l'arbre de séparation associé à σ choisi, on obtient toujours le même arbre de séparation contracté. En outre, dans les arbres de séparation contractés, et par la définition même de ces arbres, tout nœud interne (sauf la racine) d'étiquette $\oplus$ a un père d'étiquette $\ominus$, et réciproquement. Par conséquent, tous les signes dans un arbre de séparation contracté sont déterminés par le signe de la racine. Comme dans le cas des arbres de séparation standards, les étiquettes des feuilles sont des informations redondantes qui peuvent être déduites des signes des nœuds internes de l'arbre.

La figure 3.3 donne l'arbre de séparation contracté associé à la permutation séparable $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$.

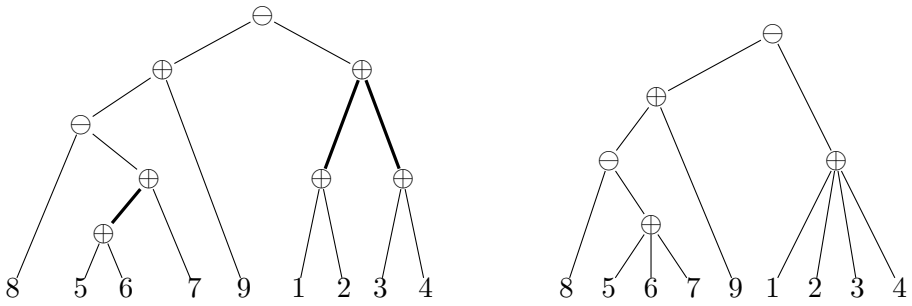


FIG. 3.3 – À droite, l'arbre de séparation contracté de la permutation $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$. Cet arbre peut être obtenu à partir de n'importe lequel des arbres de séparation de σ, en contractant les arêtes entre deux nœuds de même signe. Un arbre de séparation de σ est par exemple rappelé à gauche, les arêtes contractées étant indiquées en gras.

Dans l'opération de contraction d'arête aboutissant aux arbres de séparation contractés, on remarque que l'arité des nœuds de l'arbre ne peut qu'augmenter. Ainsi, tout nœud interne d'un arbre de séparation contracté a une arité supérieure ou égale à 2. Le squelette d'un arbre de séparation contracté est donc un arbre de Schröder. En ajoutant à un arbre de Schröder un signe pour le nœud racine, on obtient même une description complète d'un arbre de séparation contracté, puisque les étiquettes des feuilles peuvent être calculées à partir de ces seules informations. Il est facile de voir que la correspondance entre arbres de séparation contractés et permutations séparables est bijective (les détails sont donnés dans [Bou06]). Ainsi, en définissant un *arbre de Schröder signé* comme un arbre de Schröder où un signe est affecté au nœud racine, on obtient le résultat suivant :

Théorème 3.14. *Les permutations séparables sont en bijection avec les arbres de Schröder signés, via leurs arbres de séparation contractés.*

Une fois définis les arbres de séparation contractés, le lien entre arbres de séparation et arbres de décomposition des permutations séparables devrait maintenant apparaître très naturel.

Théorème 3.15. *Pour toute permutation séparable, son arbre de décomposition et son arbre de séparation contracté coïncident.*

Dans le théorème 3.15, on retrouve en filigrane la caractérisation des permutations séparables par le point (*iii*) du théorème 3.7 : les permutations séparables sont celles dont l'arbre de décomposition ne contient aucun nœud premier.

Comme on l'a dit plus haut, d'un point de vue algorithmique, on préfèrera l'usage des arbres de séparation standard à celui des arbres de séparation contractés, à cause de leur caractère binaire. Cela ne change pas les techniques utilisées, mais permet de décrire les procédures plus simplement. Pour cette raison, on verra dans le paragraphe 4.3 qu'il est utile de disposer d'une variante des arbres de décomposition qui soit "la plus binaire possible" :

Définition 3.16. Soit σ une permutation et $\mathcal{T}$ son arbre de décomposition. L'*arbre de décomposition développé* de σ est obtenu à partir de $\mathcal{T}$ en appliquant récursivement la transformation suivante sur les nœuds linéaires de $\mathcal{T}$, jusqu'à ce que tous les nœuds linéaires aient arité 2 :

> si V est un nœud linéaire de signe $\oplus$ (resp. $\ominus$) ayant $k \geq 3$ fils $V_1, V_2, \ldots, V_k$ de gauche à droite, alors on transforme V en un nœud linéaire de signe $\oplus$ (resp. $\ominus$) ayant 2 fils V_1 et V'. V' est un nouveau nœud linéaire de signe $\oplus$ (resp. $\ominus$) qui a pour fils les nœuds $V_2, \ldots, V_k$, et qui est susceptible d'être développé récursivement.

La figure 3.4 présente un exemple d'arbre de décomposition développé.

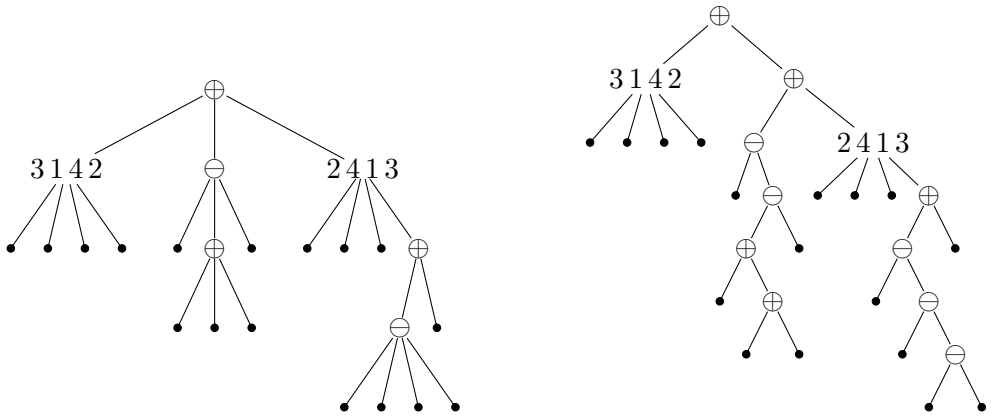


FIG. 3.4 – L'arbre de décomposition et l'arbre de décomposition développé de la permutation $\sigma = 3\,1\,4\,2\,9\,6\,7\,8\,5\,11\,17\,10\,15\,14\,13\,12\,16$.

Comme l'arbre de décomposition développé associé à une permutation séparable σ est un des arbres de séparation de σ, la définition 3.16 propose bien une extension de la notion d'arbre de séparation à toutes les permutations.

Enfin, les arbres de décomposition développés peuvent être calculés en temps linéaire à partir des arbres de décomposition standard des permutations. En effet, le nombre d'étapes de développement ne peut pas accéder le nombre de nœuds internes de l'arbre obtenu, qui est inférieur à son nombre de feuilles. Avec le théorème 2.45 (page 46) sur la complexité linéaire du calcul d'un arbre de décomposition, on a donc :

Proposition 3.17. *L'arbre de décomposition développé d'une permutation peut être calculé algorithmiquement en temps linéaire.*

Dans les algorithmes présentés dans la suite de ce chapitre, on commencera par envisager le cas des permutations séparables, en utilisant leur structure d'arbre de séparation. Puis on verra que, en utilisant les arbres de décomposition développés, on pourra adapter ces algorithmes au cas général.

3.3 Algorithme de recherche d'un motif séparable dans une permutation

On décrit ici deux algorithmes de recherche d'occurrence d'un motif *séparable* dans une permutation quelconque. Le problème général de recherche de motif dans une permutation est NP-complet, mais en imposant la restriction que le motif soit séparable, des algorithmes polynomiaux existent. C'est le cas des deux algorithmes présentés ici, qui utilisent des techniques de programmation dynamique pour aboutir à une complexité polynomiale. Les arbres de séparation jouent un rôle crucial dans la conception de ces algorithmes : ils vont guider la recherche du motif séparable considéré.

On rappelle qu'un arbre de séparation d'une permutation séparable peut être calculé en temps linéaire, par l'algorithme décrit dans [BBL98]. Modulo un pré-calcul linéaire, on peut donc s'intéresser à des algorithmes prenant en entrée une permutation quelconque τ et un arbre de séparation d'une permutation séparable σ (plutôt que la permutation σ elle-même). On notera aussi n et k les tailles respectives de τ et σ, et on conservera ces notations pour la description et l'analyse de complexité des algorithmes.

On présente deux algorithmes de recherche d'un motif séparable dans une permutation quelconque. Le premier est dû à P. Bose, J. F. Buss et A. Lubiw dans [BBL98], et son fonctionnement suit assez bien l'intuition. Le second, dû à L. Ibarra dans [Iba97], est d'approche plus difficile, bien qu'il utilise les mêmes ingrédients essentiels. Par rapport à l'algorithme de Bose, Buss et Lubiw, l'algorithme d'Ibarra améliore d'un facteur quadratique la complexité en temps, et d'un facteur linéaire la complexité en espace.

3.3.1 Algorithme de Bose, Buss et Lubiw

Étant donnés une permutation τ et un arbre de séparation $\mathcal{T}$ d'un motif séparable σ, on cherche à savoir si σ possède une occurrence dans τ. Pour répondre à cette question, l'algorithme de Bose, Buss et Lubiw utilise la technique de la programmation dynamique, et envisage des sous-problèmes, dont la résolution donne entre autre la réponse à la question initiale.

Pour tout nœud V de $\mathcal{T}$, on notera $\sigma[V]$ le motif (lui-même motif de σ) dont un arbre de séparation est le sous-arbre de $\mathcal{T}$ de racine V. On rappelle que n désigne la taille de la permutation τ.

Les sous-problèmes considérés dans l'algorithme de Bose, Buss et Lubiw sont les suivants : pour tout quadruplet d'entiers $i, j, a, b \in [1..n]$, avec $i \leq j$ et $a \leq b$, et pour tout nœud V de $\mathcal{T}$, on cherche à savoir si le motif $\sigma[V]$ a une occurrence dans τ entre les positions i et j, et n'utilisant que des éléments dont la valeur est entre a et b. Une fois tous ces sous-problèmes traités, on détermine si σ possède une occurrence dans τ en retenant seulement la réponse au sous-problème associé au nœud racine de $\mathcal{T}$ et aux valeurs $i = a = 1$ et $j = b = n$.

Pour calculer en temps polynomial les réponses aux sous-problèmes décrits ci-dessus, l'algorithme utilise la structure de l'arbre de séparation, à travers la propriété suivante :

Proposition 3.18. *Pour tout nœud interne V de l'arbre de séparation $\mathcal{T}$, de signe $\oplus$ (resp. $\ominus$), dont les fils droit et gauche sont notés V_R et V_L respectivement, $\sigma[V]$ a une occurrence dans τ, entre les positions i et j, n'utilisant que des valeurs entre a et b si et seulement s'il existe deux entiers $h \in [(i+1)..j]$ et $c \in [(a+1)..b]$ tels que*

- *il existe une occurrence de $\sigma[V_L]$ dans $\tau_i \dots \tau_{h-1}$ n'utilisant que des éléments de τ de valeur comprise entre a et $c-1$ (resp. entre c et b),*
- *et il existe une occurrence de $\sigma[V_R]$ dans $\tau_h \dots \tau_j$ n'utilisant que des éléments de τ de valeur comprise entre c et b (resp. entre a et $c-1$).*

Démonstration. S'il existe des entiers h et c vérifiant les conditions énoncées, les éléments choisis pour constituer des occurrences de $\sigma[V_L]$ et $\sigma[V_R]$ dans τ forment une occurrence de $\sigma[V]$ dans τ qui satisfait les conditions requises.

À l'inverse, s'il existe une occurrence de $\sigma[V]$ dans $\tau_i \dots \tau_j$ n'utilisant que des valeurs entre a et b, les éléments utilisés dans cette occurrence se répartissent selon les deux sous-arbres de racines V_L et V_R pour former deux occurrences de $\sigma[V_L]$ et $\sigma[V_R]$ n'ayant aucun élément en commun. En outre, l'occurrence de $\sigma[V_L]$ est à la gauche de celle de $\sigma[V_R]$ dans τ, et dans le cas où V est de signe $\oplus$ (resp. $\ominus$), les valeurs des éléments qui forment l'occurrence considérée de $\sigma[V_L]$ sont inférieures (resp. supérieures) à celles utilisées dans l'occurrence de $\sigma[V_R]$ mise en évidence. □

Pour exploiter cette propriété, l'algorithme de Bose, Buss et Lubiw maintient pour chaque nœud V de l'arbre $\mathcal{T}$ un tableau $M(V, \_, \_, \_, \_)$ à quatre dimensions tel que $M(V, i, j, a, b)$ contient 1 s'il existe une occurrence de $\sigma[V]$ dans $\tau_i\tau_{i+1} \dots \tau_j$ n'utilisant que des éléments de τ de valeur comprise entre a et b ($i \leq j$ et $a \leq b$), et 0 sinon. L'algorithme 3.1 rend compte de ce processus.

Algorithme 3.1 Algorithme de Bose, Buss et Lubiw pour la recherche d'un motif séparable dans une permutation quelconque.

1: Données : un arbre de séparation $\mathcal{T}$ d'un motif séparable σ de taille k, et une permutation quelconque τ de taille n

2: Créer un tableau M :
3: **pour** tout nœud V de $\mathcal{T}$ **faire**
4: **pour** tous entiers i, j, a et $b \in [1..n]$ **faire**
5: $M(V, i, j, a, b) \leftarrow 0$
6: **fin pour**
7: **fin pour**

8: Remplir M :
9: **pour** tout nœud V de $\mathcal{T}$, depuis les feuilles de $\mathcal{T}$ vers la racine **faire**
10: **pour** tous entiers i, j, a et $b \in [1..n]$, tels que $i \leq j$ et $a \leq b$ **faire**
11: **si** V est une feuille **alors**
12: **si** $\exists h \in [i..j]$ tel que $\tau_h \in [a..b]$ **alors**
13: $M(V, i, j, a, b) \leftarrow 1$
14: **sinon**
15: $M(V, i, j, a, b) \leftarrow 0$
16: **finsi**
17: **sinon**
18: /*V est un nœud interne.*/
19: On note respectivement V_R et V_L les fils droit et gauche de V.
20: **si** V est un nœud de signe $\oplus$ **alors**
21: $M(V, i, j, a, b) \leftarrow Max\{M(V_L, i, h-1, a, c-1) \cdot M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$
22: **sinon**
23: /*V est un nœud de signe $\ominus$.*/
24: $M(V, i, j, a, b) \leftarrow Max\{M(V_L, i, h-1, c, b) \cdot M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$
25: **finsi**
26: **finsi**
27: **fin pour**
28: **fin pour**

29: Sortie : $M(R, 1, n, 1, n)$, où R désigne la racine de $\mathcal{T}$

Théorème 3.19. *L'algorithme 3.1 est polynomial en temps et en espace. Plus précisément, il s'exécute en temps $\mathcal{O}(kn^6)$ et utilise un espace $\mathcal{O}(kn^4)$.*

Démonstration. Le calcul de $M(V, i, j, a, b)$ pour les feuilles de $\mathcal{T}$ est en temps $\mathcal{O}(n)$, et pour les nœuds internes, $M(V, i, j, a, b)$ est calculé à partir des valeurs précédentes en temps $\mathcal{O}(n^2)$. Par conséquent, pour chaque nœud V de $\mathcal{T}$, pour remplir tout le tableau $M(V, \_, \_, \_, \_)$, il faut un temps $\mathcal{O}(n^6)$, donc un temps $\mathcal{O}(kn^6)$ pour la totalité de l'algorithme. L'espace occupé par l'ensemble des tableaux est quant à lui $\mathcal{O}(kn^4)$. □

On peut remarquer qu'une légère modification de l'algorithme permet de calculer le nombre d'occurrences de σ dans τ : il suffit de remplacer les calculs de maximum aux lignes 21 et 24 de l'algorithme 3.1 par les sommes des éléments contenus dans ces ensembles.

Si on cherche à exhiber une occurrence de σ dans τ, on peut encore utiliser l'algorithme 3.1, sans en augmenter la complexité. Il suffit de retenir en plus, dans les cases du tableau contenant 1, quelles sont les valeurs de h et c qui font en sorte que cette case contienne 1.

3.3.2 Algorithme d'Ibarra

L. Ibarra a proposé dans [Iba97] un autre algorithme polynomial pour résoudre le problème de recherche de motif séparable dans une permutation quelconque. Comme l'algorithme de Bose, Buss et Lubiw, il utilise des techniques de programmation dynamique et la recherche du motif séparable est guidée par son arbre de séparation (ou plus exactement, par un de ses arbres de séparation, au choix).

L'amélioration de L. Ibarra porte sur la complexité de l'algorithme : elle passe par sa méthode à $\mathcal{O}(kn^4)$ en temps et $\mathcal{O}(kn^3)$ en espace. Pour y parvenir, L. Ibarra considère moins de sous-problèmes par nœud de l'arbre de séparation : son algorithme remplit deux tableaux $L(V, \_, \_, \_)$ et $H(V, \_, \_, \_)$, qui sont chacun de dimension 3, contre 4 dans l'algorithme de Bose, Buss et Lubiw. En utilisant les mêmes notations que dans le paragraphe précédent, ces tableaux sont définis par : pour tout nœud V de $\mathcal{T}$, pour tout $1 \leq i \leq j \leq n$ et pour tout $1 \leq x \leq n$

- $L(V, i, j, x) = Max\{\{0\} \cup \{y$: il y a une occurrence de $\sigma[V]$ dans $\tau_i\tau_{i+1}\dots\tau_j$ dont le plus petit élément est y et le plus grand élément est $\leq x\}\}$,
- $H(V, i, j, x) = Min\{\{n+1\} \cup \{y$: il y a une occurrence de $\sigma[V]$ dans $\tau_i\tau_{i+1}\dots\tau_j$ dont le plus grand élément est y et le plus petit élément est $\geq x\}\}$.

En particulier, $L(V, i, j, x) > 0$ si et seulement s'il existe une occurrence de $\sigma[V]$ dans $\tau_i\tau_{i+1}\dots\tau_j$ dont les éléments sont inférieurs ou égaux à x, et $H(V, i, j, x) < n + 1$ si et seulement s'il existe une occurrence de $\sigma[V]$ dans $\tau_i\tau_{i+1}\dots\tau_j$ dont les éléments sont supérieurs ou égaux à x. À la fin de la procédure, les valeurs $L(R, 1, n, n)$ et $H(R, 1, n, 1)$, si elles sont différentes de 0 et $n + 1$ respectivement, indiquent qu'il existe une occurrence de σ dans τ. Elles sont même les meilleurs bornes inférieures et supérieures respectivement sur les valeurs utilisées dans une occurrence de σ dans τ.

L'espace occupé par l'ensemble des tableaux $L(V, \_, \_, \_)$ et $H(V, \_, \_, \_)$ est clairement $\mathcal{O}(kn^3)$. Il est aussi facile de voir que pour une feuille V de $\mathcal{T}$, on peut remplir toutes les cases des tableaux $L(V, \_, \_, \_)$ et $H(V, \_, \_, \_)$ en temps $\mathcal{O}(n^4)$. Dans [Iba97], L. Ibarra détaille la procédure qui permet de remplir les deux tableaux $L(V, \_, \_, \_)$ et $H(V, \_, \_, \_)$ pour tout nœud interne V ayant deux fils V_L et V_R en temps $\mathcal{O}(n^4)$ à partir des tableaux $L(V_L, \_, \_, \_)$, $H(V_L, \_, \_, \_)$, $L(V_R, \_, \_, \_)$ et $H(V_R, \_, \_, \_)$. Il démontre aussi la correction de cette procédure.

L'algorithme d'Ibarra est moins intuitif, et on le mentionne, sans en expliciter les détails, surtout pour l'amélioration de complexité qu'il permet. Comme pour celui de Bose, Buss et Lubiw, on peut modifier légèrement l'algorithme d'Ibarra pour garder une trace de l'occurrence de σ dans τ, en mémorisant en plus dans chaque case $L(V, i, j, x)$ les valeurs de j et x qui ont fait passer à 1 la valeur contenue dans cette case du tableau L. En revanche, une différence entre l'algorithme de Bose, Buss et Lubiw et celui d'Ibarra est que ce dernier ne permet pas de compter le nombre d'occurrence de σ dans τ.

Dans toute cette partie de la thèse, les algorithmes développés pour la recherche de motifs communs entre des permutations sont inspirés de celui de Bose, Buss et Lubiw. La conception plus simple de cet algorithme, par rapport à celui d'Ibarra, autorise à l'étendre assez facilement à des problèmes plus larges que la recherche d'occurrence de motif séparable, même si sa complexité n'est pas très attractive. Une question qui reste ouverte est de savoir si, pour les algorithmes qui seront exposés dans cette partie, on peut en trouver des variantes en utilisant les idées de L. Ibarra, pour en améliorer la complexité.

Chapitre 4

Recherche de plus grand motif commun à deux permutations

Sommaire

4.1 Le problème de recherche de plus grand motif commun à deux permutations 82
4.1.1 Définition du problème, et premières remarques de complexité 82
4.1.2 Une généralisation : recherche de plus grand motif commun entre une permutation et une classe 83
4.1.3 Restriction à des permutations triables par pile 84
4.2 Recherche du plus grand motif commun à deux permutations, dont une séparable 86
4.2.1 Description de l'algorithme 86
4.2.2 Correction et analyse de complexité 88
4.3 Extension à deux permutations quelconques grâce aux arbres de décomposition 90
4.3.1 Description de l'algorithme 91
4.3.2 Correction et analyse de complexité 92

4.1 Le problème de recherche de plus grand motif commun à deux permutations

4.1.1 Définition du problème, et premières remarques de complexité

Dans ce chapitre 4, on étudie un problème connexe à la recherche d'occurrence de motif dans une permutation, qui a fait l'objet du chapitre 3 : étant données deux permutations τ_1 et τ_2, on recherche une permutation qui soit motif de τ_1 et de τ_2, et qui soit la plus longue possible. Ce problème de recherche de plus grand motif commun se formule ainsi :

Problème 4.1 (Recherche de plus grand motif commun)**.**
Données : Deux permutations τ_1 et τ_2 de tailles n_1 et n_2 respectivement.
Sortie : Une permutation σ qui est motif à la fois de τ_1 et de τ_2, et qui est de taille maximale parmi les permutations satisfaisant cette propriété.

On peut d'abord remarquer qu'il peut y avoir plusieurs plus grands motifs communs à deux permutations données. Par exemple, les permutations 6 3 1 5 2 4 et 5 2 1 3 6 4 n'ont aucun motif de taille 5 en commun, mais plusieurs permutations de taille 4 apparaissent comme motif dans chacune d'elles (4 2 1 3, 4 1 2 3, 3 1 4 2 et 2 1 4 3). Il faut donc parler du problème de recherche *d'un* plus grand motif commun, ou *de* plus grand motif commun, et non *du* plus grand motif commun.

Du point de vue de la complexité, le problème de recherche de motif (problème 3.1) se réduit à celui de recherche de plus grand motif commun : en effet, σ a une occurrence dans τ si et seulement si σ et τ ont un plus grand motif commun unique et est égal à σ. On déduit donc de la NP-complétude de la recherche de motif (théorème 3.2) que le problème 4.1 est NP-difficile.

Une généralisation du problème 4.1, sur laquelle nous reviendrons dans le chapitre 5, est la recherche de plus grand motif commun à un ensemble de permutations, pouvant contenir $k \geq 2$ permutations. Le nombre k de permutations dans l'ensemble en entrée peut être considéré comme faisant partie des données du problème, ou au contraire comme une constante extérieure. Nous verrons que cette distinction peut faire une vraie différence en termes de complexité.

Le paragraphe 4.1.2 présente une autre généralisation, proposée dans [AAA+03] par M. H. Albert, R. E. L. Aldred, M. D. Atkinson, H. P. van Ditmarsch, B. D. Handley, C. C. Handley et J. Opatrny. On fixe une classe X de permutations, la donnée du problème est une permutation τ, et il s'agit de calculer un plus grand motif de τ qui appartient à X. En quelque sorte, on recherche un plus grand motif commun entre une permutation et une classe. La complexité n'est dans ce cas mesurée que par rapport à la taille de τ, et il n'y a pas de contrainte sur la manière dont la classe X est décrite (par base de motifs exclus par exemple, mais pas obligatoirement).

Dans la suite, on présentera aussi deux restrictions du problème, qui le rendent polynomial : la recherche de plus grand motif commun entre deux permutations triables par pile, et la recherche de plus grand motif commun à deux permutations dont une est séparable. Le résultat sur les permutations triables par pile est dû à A. Micheli et D. Rossin dans [MR06], et fait l'objet du paragraphe 4.1.3. Celui sur les permutations séparables généralise le résultat de [MR06], mais en utilisant des outils différents (les arbres de séparation), comme cela sera décrit dans le paragraphe 4.2.

En outre, les techniques développées dans le cas où une permutation est séparable permettent de décrire un algorithme pour la recherche de plus grand motif commun entre deux permutations, sans restrictions. Cet algorithme reste bien sûr de complexité exponentielle, mais fournit une amélioration par rapport à l'algorithme naïf qui consisterait à comparer tous les motifs de τ_1 à tous ceux de τ_2. Cet élargissement fera l'objet du paragraphe 4.3.

4.1.2 Une généralisation : recherche de plus grand motif commun entre une permutation et une classe

Dans ce paragraphe, on décrit un problème généralisant la recherche de plus grand motif commun entre deux permutations. On présente les difficultés à le résoudre en général, et une famille de cas particuliers pour lesquels la résolution est polynomiale. On n'entre pas dans les détails, qui peuvent être trouvés dans [AAA+03].

Le problème *LIS* (pour *Longest Increasing Subsequence*) a déjà été présenté dans le paragraphe 3.1.3 : il s'agit de trouver la plus longue permutation de la forme $12\dots k$ qui est motif d'une permutation τ en entrée du problème. Dans l'article [AAA+03], les auteurs proposent une généralisation de ce problème. Ils définissent donc, pour tout ensemble X de permutations le problème LXS comme suit :

Problème 4.2 (LXS).
Données : Une permutation τ de taille n.
Sortie : Une permutation σ de l'ensemble X qui est motif de τ, et qui est de taille maximale parmi les permutations satisfaisant cette propriété.

Pour $X = \{12\dots k, k \geq 1\}$, on retrouve bien le problème *LIS* d'origine.

Le problème 4.2 généralise aussi le problème de recherche de plus grand motif commun. Il peut être vu comme la recherche d'un plus grand motif commun entre une permutation τ et un ensemble X de permutations. Pour $X = \{\pi : \pi \preceq \tau'\}$, la résolution du problème 4.2 revient à trouver un plus grand motif commun à τ' et à la permutation τ en entrée du problème.

Pour que cette généralisation soit pertinente, il faut que tout motif d'une permutation appartenant à X soit aussi dans X. En d'autres termes, on considère le problème 4.2 seulement pour des ensembles X qui sont des classes de permutations. Cette restriction est imposée aussi dans l'étude de [AAA+03], pour des raisons semblables, bien que formulées différemment.

Il faut remarquer que la classe X est extérieure aux données du problème, et que la complexité n'est mesurée que par rapport à la taille de la permutation τ en entrée.

Sans condition sur X, le problème 4.2 est NP-difficile, mais la réduction pour le montrer ne se fait pas avec le problème de recherche de motif comme dans tous les cas précédents. Cela serait possible si X faisait partie des données du problème, mais ça n'est pas le cas. On peut seulement réduire au problème 4.2 le problème de recherche de motif *où le motif est considéré comme extérieur aux données en entrée*, mais on a vu que cette variante du problème de recherche de motif est polynomiale.

Le caractère NP-difficile du problème 4.2 est justifié par une réduction du problème d'appartenance à X. On connait des classes X de permutations pour lesquelles le problème de décider si une permutation τ appartient à X est NP-difficile, et tester si la réponse au problème 4.2 sur l'entrée τ est égale à τ revient à tester si τ appartient à X.

La classe X étant extérieure aux données du problème 4.2, cela laisse toute latitude quant à la manière dont elle est décrite. Elle peut être donnée par sa base (finie) de motifs exclus, ou par d'autres caractérisations. Une description de X comme solution minimale (pour l'inclusion) d'une équation d'une certaine forme est particulièrement adaptée à la résolution du problème LXS. En effet, si X est décrite de cette manière (ce qui n'est pas toujours possible), on peut résoudre le problème 4.2 en temps polynomial.

Les équations dont il s'agit sont celles de la forme $X = H[X_1, \dots, X_r]$, où H est un ensemble de permutations de taille r (donc un ensemble fini), et chaque X_i est soit X lui-même, soit une classe D_i pour laquelle on sait résoudre le problème LD_iS en temps polynomial. Cela signifie que les éléments de X peuvent être vus comme une dilatation (voir définition page 41) d'un élément de H par des permutations appartenant aux X_i. Si l'on dispose d'une telle description récursive de X, un algorithme polynomial résolvant le problème LXS fonctionne par programmation dynamique,

avec des sous-problèmes pour des fenêtres de valeurs et de positions de la permutation τ en entrée, un peu comme dans l'algorithme 3.1 de Bose, Buss et Lubiw.

4.1.3 Restriction à des permutations triables par pile

De même que la recherche d'occurrence de motif dans une permutation, le problème 4.1 de recherche de plus grand motif commun à deux permutations est NP-difficile. Comme dans le chapitre 3, il est donc naturel d'envisager des restrictions de ce problème pour lesquelles on peut proposer une solution par un algorithme polynomial. Un premier exemple de telle restriction a été étudié par Micheli et Rossin, qui considèrent dans [MR06] le problème de recherche de plus grand motif commun à deux permutations *triables par pile*.

Les permutations triables par pile ont été introduites par D. E. Knuth dans [Knu73]. Comme leur nom l'indique, ce sont celles qui peuvent être triées (c'est-à-dire transformées en $12\dots n$) par un passage dans une pile. Cette première définition est illustrée par la figure 4.1.

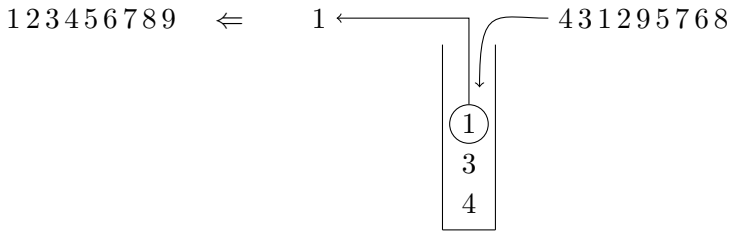


FIG. 4.1 – Tri d'une permutation par une pile.

De nombreuses caractérisations des permutations triables par pile ont ensuite été décrites, et on adopte une de ces caractérisations comme définition "formelle" :

Définition 4.3. Une permutation de $[k..n]$ est *triable par pile* si on peut la décomposer sous la forme InJ, où il existe un entier p tel que I soit une permutation de $[k..p]$ et J soit une permutation de $[p+1..n-1]$, toutes deux triables par pile. On autorise I et J à être des permutations vides.

Dans tout ce qui suit, on s'intéresse seulement aux permutations $\sigma \in \mathcal{S}_n$ (c'est-à-dire aux permutations de $[1..n]$) qui sont triables par une pile.

Les permutations triables par pile forment une classe de permutations, et on peut facilement caractériser cette classe par sa base :

Théorème 4.4 ([Knu73])**.** *Les permutations triables par pile sont celles qui évitent le motif* 2 3 1*.*

Le théorème 4.4 implique que les permutations triables par pile sont énumérées par les nombres de Catalan. On peut proposer une autre preuve de ce résultat, à travers une bijection entre permutations triables par pile de taille n et arbres plans enracinés à n arêtes. C'est aussi à travers cette bijection que [MR06] donne un algorithme polynomial pour la recherche de plus grand motif commun à deux permutations triables par pile.

Étant donné un arbre plan enraciné, on construit une permutation de la manière suivante. On effectue un premier parcours en profondeur en numérotant les arêtes empruntées dans un ordre postfixe. Puis par un second parcours en profondeur de l'arbre, on lit les étiquettes attribuées aux arêtes, cette fois dans un ordre préfixe. La permutation obtenue est triable par pile.

Réciproquement, étant donnée une permutation triable par pile, on la décompose en $\sigma = InJ$ comme dans la définition 4.3, et on construit un arbre récursivement. En notant $\mathcal{T}_I$ et $\mathcal{T}_J$ les arbres associés à I et J respectivement, et r_I et r_J leurs racines respectives, l'arbre $\mathcal{T}_\sigma$ associé à σ est obtenu en reliant $\mathcal{T}_I$ et $\mathcal{T}_J$ par une arête allant de r_I à r_J, et en enracinant l'arbre ainsi obtenu en r_I.

La figure 4.2 donne un exemple d'une permutation et d'un arbre qui sont en correspondance par cette bijection.

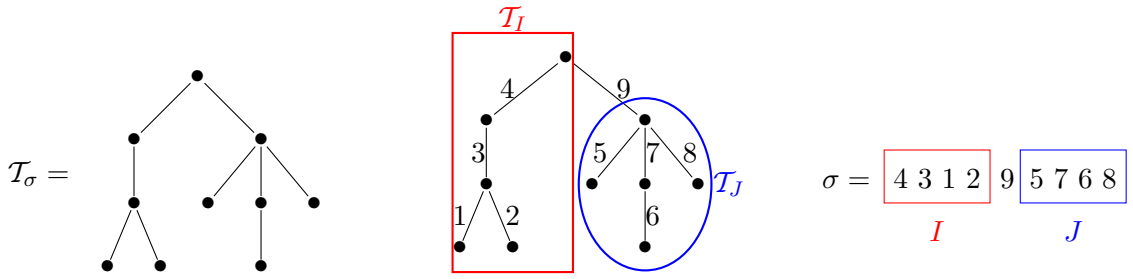


FIG. 4.2 – La bijection entre permutations triables par pile et arbres plans enracinés sur un exemple.

Cette bijection possède en outre la propriété de faire correspondre plus grand motif commun entre deux permutations triables par pile et plus grand sous-arbre commun entre deux arbres.

Définition 4.5. Un *sous-arbre* d'un arbre $\mathcal{T}$ est un arbre qui peut être obtenu à partir de $\mathcal{T}$ par contraction d'arêtes. Un *plus grand sous-arbre commun* à deux arbres $\mathcal{T}_1$ et $\mathcal{T}_2$ est un sous-arbre de $\mathcal{T}_1$ et de $\mathcal{T}_2$ qui a un nombre maximal d'arêtes parmi tous ceux possédant cette propriété.

Plus précisément, si σ_1 et σ_2 sont deux permutations triables par pile, et $\mathcal{T}_1$ et $\mathcal{T}_2$ les arbres qui leur sont associés par la bijection, alors tout plus grand motif commun à σ_1 et σ_2 (noté π) est associé par la même bijection à un arbre $\mathcal{T}_\pi$ qui est un plus grand sous-arbre commun à $\mathcal{T}_1$ et $\mathcal{T}_2$. Réciproquement, tout plus grand sous-arbre commun à $\mathcal{T}_1$ et $\mathcal{T}_2$ est en correspondance par la bijection décrite avec une permutation qui est un plus grand motif commun à σ_1 et σ_2.

Le problème de recherche de plus grand motif commun à deux permutations triables par pile est donc équivalent à celui du calcul d'un plus grand sous-arbre commun à deux arbres.

Dans [ZS89], K. Zhang et D. Shasha proposent un algorithme polynomial qui résout le problème de calcul de distance d'édition entre deux arbres, $\mathcal{T}_1$ et $\mathcal{T}_2$. Ce problème consiste à trouver la longueur minimale d'une suite d'opérations d'édition (insertion, contraction, et réétiquetage d'arêtes) permettant de transformer $\mathcal{T}_1$ et $\mathcal{T}_2$. L'algorithme de Zhang et Shasha calcule en fait une application des sommets de $\mathcal{T}_1$ dans les sommets de $\mathcal{T}_2$, en utilisant des techniques de programmation dynamique, et explique comment la distance d'édition peut être calculée à partir de cette application. Il permet aussi de produire une suite d'opérations d'édition pour passer de $\mathcal{T}_1$ à $\mathcal{T}_2$.

Le point qui nous intéresse tout particulièrement est qu'on peut facilement construire, à partir d'une suite d'opérations réalisant la distance d'édition entre deux arbres $\mathcal{T}_1$ et $\mathcal{T}_2$, un plus grand sous-arbre commun à ces deux arbres. Ainsi, on dispose d'un algorithme polynomial pour la recherche de plus grand sous-arbre commun à deux arbres. L'algorithme de Zhang et Shasha est de complexité $\mathcal{O}(n^4)$. Mais d'autres algorithmes de calcul de distance d'édition sur les arbres (et d'une suite d'opérations d'édition qui réalise cette distance) ont depuis été décrits (voir par exemple [ZWS95, Kle95, Val01, DT03, Bil05]), améliorant ainsi le temps nécessaire pour résoudre ce problème à $\mathcal{O}(n^3 \log(n))$ [DT03].

La bijection qui relie arbres et permutations triables par pile pouvant aisément être calculée en temps quadratique (complexité qui est améliorable en raffinant un peu), [MR06] a donc proposé une réinterprétation du résultat de K. Zhang et D. Shasha, pour obtenir un algorithme polynomial résolvant le problème de recherche de plus grand motif commun entre deux permutations triables par pile.

La complexité de l'algorithme obtenu est héritée de l'algorithme de Zhang et Shasha, soit $\mathcal{O}(n^4)$. Mais tous les algorithmes de calcul d'une suite d'opérations qui réalise la distance d'édition entre deux arbres fournissent immédiatement une amélioration de cette complexité.

4.2 Recherche du plus grand motif commun à deux permutations, dont une séparable

On propose ici une autre restriction du problème 4.1 qui autorise une résolution par un algorithme de complexité polynomiale. Il s'agit du problème de recherche de plus grand motif commun entre une permutation séparable et une permutation quelconque. Les permutations triables par pile étant un cas particulier de permutations séparables, ce cadre est plus général que celui du paragraphe 4.1.3.

On s'attache donc à décrire un algorithme pour démontrer le théorème suivant :

Théorème 4.6 ([BR06])**.** *Le problème de recherche de plus grand motif commun à deux permutations dont une est séparable peut être résolu en temps polynomial.*

On supposera qu'en entrée du problème 4.1, la permutation τ_1 est séparable. La séparabilité d'une permutation pouvant être testée en temps linéaire, cette hypothèse n'est pas une restriction, mais permet d'alléger les notations en évitant une distinction de cas selon que τ_1 ou τ_2 est séparable.

On peut même remarquer qu'il n'est pas nécessaire de savoir *a priori* laquelle des deux permutations en entrée est séparable : cela peut en effet être déterminé par un précalcul linéaire.

Toujours dans les remarques d'ordre général, on constate que tout plus grand motif commun à une permutation séparable et une permutation quelconque est lui-même séparable. Cette propriété est une conséquence directe du fait que les permutations séparables forment une classe de permutations, qui est par définition fermée pour la relation de motif.

4.2.1 Description de l'algorithme

L'algorithme proposé pour résoudre le problème 4.1, avec la condition supplémentaire que la permutation τ_1 en entrée est séparable, utilise les mêmes outils que celui de Bose, Buss et Lubiw [BBL98], pour la recherche d'occurrence d'un motif séparable dans une permutation quelconque. Il calcule d'abord un arbre de séparation $\mathcal{T}_1$ de τ_1, puis effectue la recherche d'un plus grand motif commun aux deux permutations en étant guidé par cet arbre. Pour éviter une explosion combinatoire, on utilise comme dans [BBL98] la technique de la programmation dynamique, le problème de départ étant décomposé en sous-problèmes en fonction de la structure de $\mathcal{T}_1$.

On rappelle que d'après le théorème 3.10, on peut non seulement tester en temps linéaire si une permutation est séparable, mais aussi calculer dans l'affirmative un arbre de séparation pour cette permutation toujours en temps linéaire. Ainsi, au lieu d'avoir deux permutations τ_1 (séparable) et τ_2 (quelconque) en entrée du problème 4.1, on supposera qu'on a plutôt un arbre de séparation $\mathcal{T}_1$ d'une permutation séparable τ_1, et une permutation τ_2. Cette reformulation du problème peut être obtenue au prix d'un précalcul linéaire. On rappelle que n_1 et n_2 désignent les tailles respectives de τ_1 et τ_2. On peut remarquer qu'alors le nombre de nœuds de $\mathcal{T}_1$ est en $\mathcal{O}(n_1)$.

Le tableau de programmation dynamique utilisé dans l'algorithme 4.1 est de la forme

$$M = \{M(V, i, j, a, b) : V \text{ un nœud de } \mathcal{T}_1, 1 \leq i \leq j \leq n_2, 1 \leq a \leq b \leq n_2\}.$$

Comme dans le paragraphe 3.3.1, pour tout nœud V de $\mathcal{T}_1$, on note $\tau_1[V]$ le motif de τ_1 dont un arbre de séparation est le sous-arbre de $\mathcal{T}_1$ de racine V. On note aussi, pour $1 \leq i \leq j \leq n_2$, $\tau_2[i..j]$ la sous-séquence de la permutation τ_2 allant de l'élément à la position i jusqu'à l'élément à la position j.

La case $M(V, i, j, a, b)$ du tableau M contient un plus grand motif commun π entre $\tau_1[V]$ et $\tau_2[i..j]$ sous la contrainte qu'il existe une occurrence de π dans $\tau_2[i..j]$ qui n'utilise que des éléments de valeur entre a et b.

La permutation vide, de taille 0, sera notée ϵ.

Exemple 4.7. Si le nœud V représente le motif 2 1 (c'est-à-dire si $\tau_1[V] = 2\,1$), et si $\tau_2 = 6\,4\,2\,5\,3\,1$, alors on a par exemple $M(V, 2, 4, 3, 5) = 1$, $M(V, 2, 5, 3, 4) = 2\,1$ et $M(V, 4, 5, 1, 2) = \epsilon$.

Algorithme 4.1 Algorithme de recherche d'un plus grand motif commun à une permutation séparable et une permutation quelconque.

Données : un arbre de séparation $\mathcal{T}_1$ d'une permutation séparable τ_1 de taille n_1, et une permutation quelconque τ_2 de taille n_2

Créer un tableau M :
pour tout nœud V de $\mathcal{T}_1$ **faire**
  pour tous entiers i, j, a et $b \in [1..n_2]$ **faire**
    $M(V, i, j, a, b) \leftarrow \epsilon$
  fin pour
fin pour

Remplir M :
pour tout nœud V de $\mathcal{T}_1$, depuis les feuilles de $\mathcal{T}_1$ vers la racine **faire**
  pour tous entiers i, j, a et $b \in [1..n_2]$, tels que $i \leq j$ et $a \leq b$ **faire**
    si V est une feuille **alors**
      si $\exists h \in [i..j]$ tel que $\tau_2(h) \in [a..b]$ **alors**
        $M(V, i, j, a, b) \leftarrow 1$
      sinon
        $M(V, i, j, a, b) \leftarrow \epsilon$
      finsi
    sinon
      /*V est un nœud interne.*/
      On note respectivement V_R et V_L les fils droit et gauche de V.
      si V est un nœud de signe $\oplus$ **alors**
        $M(V, i, j, a, b) \leftarrow Longest\{M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) : i \leq h \leq j+1, a \leq c \leq b+1\}$
      sinon
        /*V est un nœud de signe $\ominus$.*/
        $M(V, i, j, a, b) \leftarrow Longest\{M(V_L, i, h-1, c, b) \ominus M(V_R, h, j, a, c-1) : i \leq h \leq j+1, a \leq c \leq b+1\}$
      finsi
    finsi
  fin pour
fin pour

Sortie : $M(R, 1, n_2, 1, n_2)$, où R désigne la racine de $\mathcal{T}_1$

Le fonctionnement de l'algorithme est le suivant. Pour commencer, on remplit les tableaux $M(V, \_, \_, \_, \_)$ pour toutes les feuilles V de $\mathcal{T}_1$. Ensuite, on calcule $M(V, \_, \_, \_, \_)$ pour tout nœud interne V en utilisant les valeurs contenues dans les tableaux $M(V_L, \_, \_, \_, \_)$ et $M(V_R, \_, \_, \_, \_)$ associés aux fils de V (V_L et V_R désignent respectivement les fils gauche et droit de V).

Pour pouvoir combiner les motifs stockés dans les tableaux $M(V_L, \_, \_, \_, \_)$ et $M(V_R, \_, \_, \_, \_)$, afin de remplir le tableau $M(V, \_, \_, \_, \_)$, on introduit la définition suivante de concaténation de motifs :

Définition 4.8. On considère deux motifs π et π' de tailles respectives k et k'. Les *concaténations*

positive et *négative* de π et π' sont définies respectivement par :

$$\pi \oplus \pi' = \pi_1 \cdots \pi_k(\pi'_1 + k) \cdots (\pi'_{k'} + k) \text{ et}$$

$$\pi \ominus \pi' = (\pi_1 + k') \cdots (\pi_k + k')\pi'_1 \cdots \pi'_{k'}.$$

Exemple 4.9.

$4\,3\,5\,2\,1 \oplus 3\,1\,4\,2 = 4\,3\,5\,2\,1 \vdots 8\,6\,9\,7$

$4\,3\,5\,2\,1 \ominus 3\,1\,4\,2 = 8\,7\,9\,6\,5 \vdots 3\,1\,4\,2$

On peut remarquer qu'avec la notation introduite par la définition de la dilatation (définition 2.29 page 41), la concaténation positive $\pi \oplus \pi'$ s'écrit aussi $12[\pi, \pi']$. De même, $\pi \ominus \pi' = 21[\pi, \pi']$.

On dispose maintenant de tous les outils nécessaires à la description plus précise de l'algorithme 4.1 pour la recherche de plus grand motif commun à deux permutations, dont une est séparable.

Dans l'algorithme 4.1, aux lignes 21 et 24, on utilise une procédure *Longest* qui calcule, pour tout ensemble de motifs S, un motif de taille maximale dans S. Une procédure naïve qui parcourt l'ensemble S en mémorisant un motif de taille maximale parmi ceux déjà rencontrés, et qui met à jour le motif retenu dès qu'un motif plus long est rencontré, est suffisante pour nos besoins.

4.2.2 Correction et analyse de complexité

Il reste maintenant à vérifier que l'algorithme 4.1 est correct, et à calculer sa complexité.

Proposition 4.10. *L'algorithme 4.1 est correct : il calcule un plus grand motif commun aux deux permutations τ_1 et τ_2 données en entrée.*

Démonstration. On démontre la proposition 4.10 par induction.

On montre que pour tout nœud V de $\mathcal{T}_1$ et tous $i, j, a, b \in [1..n_2]$, l'algorithme stocke dans la case $M(V, i, j, a, b)$ un plus grand motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ dont l'occurrence dans $\tau_2[i..j]$ utilise seulement des valeurs entre a et b. On dira que le nœud V vérifie la propriété $\mathcal{P}$ lorsque cette assertion est vraie pour toutes les valeurs de $i, j, a, b \in [1..n_2]$.

Au vu des lignes 12 à 16 de l'algorithme 4.1, il est clair que la propriété $\mathcal{P}$ est vérifiée pour toutes les feuilles V de $\mathcal{T}_1$.

On considère donc un nœud interne V, qui a donc deux fils V_L (le fils gauche) et V_R (le fils droit). On fixe alors des entiers i, j, a et b tels que $1 \leq i, j \leq n_2$ et $1 \leq a, b \leq n_2$. On peut remarquer que dans le cas où $i > j$ ou $a > b$, le motif dans la case $M(V, i, j, a, b)$ est le motif vide (la valeur affectée à la ligne 5 de l'algorithme n'a dans ce cas jamais été modifiée), ce qui correspond bien au plus grand motif commun attendu.

On suppose donc dans la suite de la démonstration que $i \leq j$ et que $a \leq b$. On suppose en outre que V est un nœud positif, le cas d'un nœud négatif pouvant être traité de manière très similaire. Tout d'abord, on voit assez facilement que $M(V, i, j, a, b)$ contient un motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ qui n'utilise que des valeurs entre a et b. En effet, par hypothèse d'induction, on déduit que tout motif de l'ensemble $S = \{M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) : i \leq h \leq j+1, a \leq c \leq b+1\}$, et donc *a fortiori* $Longest(S)$, est un motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ n'utilisant que des valeurs entre a et b.

Pour conclure l'étape d'hérédité de la preuve par induction de la propriété 4.10, il reste à démontrer que :

Lemme 4.11. *$Longest(S)$ est un motif de taille maximale parmi tous les motifs communs entre $\tau_1[V]$ et $\tau_2[i..j]$ n'utilisant que des valeurs entre a et b dans $\tau_2[i..j]$.*

Démonstration. On rappelle qu'on cherche ici à démontrer la propriété $\mathcal{P}$ pour un nœud interne V de $\mathcal{T}_1$, sachant que les nœuds V_L et V_R satisfont la propriété $\mathcal{P}$, par hypothèse d'induction.

On note π un plus grand motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$, parmi ceux n'utilisant que des valeurs entre a et b dans $\tau_2[i..j]$. Comme illustré sur la figure 4.3, il existe des entiers $h \in \{i, \dots, j+1\}$ et $c \in \{a, \dots, b+1\}$ tels que π se décompose en $\pi = \pi_1 \oplus \pi_2$, où π_1 est un motif commun entre $\tau_1[V_L]$ and $\tau_2[i..h-1]$, utilisant seulement des valeurs entre a et $c-1$ dans $\tau_2[i..h-1]$, et π_2 est un motif commun entre $\tau_1[V_R]$ et $\tau_2[h..j]$, utilisant seulement des valeurs entre c et b dans $\tau_2[h..j]$. Il faut remarquer que dans cette décomposition, il est possible que π_1 ou π_2 soit le motif vide.

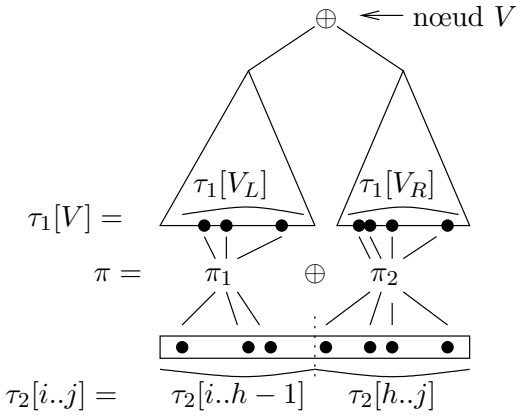


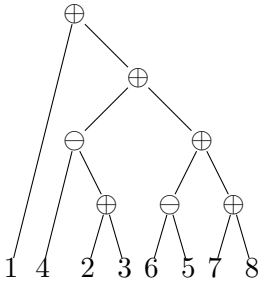
FIG. 4.3 – Décomposition d'un plus grand motif commun à deux permutations, dont une séparable.

On montre facilement que π_1 (resp. π_2) est en fait un *plus grand* motif commun entre $\tau_1[V_L]$ (resp. $\tau_1[V_R]$) et τ_2 dans les intervalles de positions et de valeurs précisés plus haut. En effet, si π_1 (resp. π_2) n'était pas un plus grand motif commun dans ces intervalles de positions et de valeurs, alors π ne serait pas non plus de taille maximale, ce qui contredit la définition même de π. Ainsi, par l'hypothèse d'induction, on déduit que $|M(V_L, i, h-1, a, c-1)| = |\pi_1|$ et $|M(V_R, h, j, c, b)| = |\pi_2|$. Le motif calculé par l'algorithme 4.1 dans la case $M(V, i, j, a, b)$ est donc de taille au moins $|M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b)| = |\pi_1 \oplus \pi_2| = |\pi|$. Et π étant de taille maximale par définition, on conclut que le motif dans la case $M(V, i, j, a, b)$ est aussi de taille maximale. □

En conclusion, on obtient donc que $M(V, i, j, a, b)$ contient un plus grand motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ parmi ceux dont l'occurrence dans $\tau_2[i..j]$ n'utilise que des valeurs entre a et b.

Lorsque V est un nœud négatif, on décompose π en $\pi_1 \ominus \pi_2$, où π_1 est un plus grand motif commun entre $\tau_1[V_L]$ et $\tau_2[i..h-1]$, utilisant seulement des valeurs entre c et b dans $\tau_2[i..h-1]$, et π_2 est un plus grand motif commun entre $\tau_1[V_R]$ et $\tau_2[h..j]$, utilisant seulement des valeurs entre a et $c-1$ dans $\tau_2[h..j]$, et on poursuit ensuite la démonstration en suivant les mêmes étapes. □

Exemple 4.12. On considère les permutations $\tau_1 = 1\,4\,2\,3\,6\,5\,7\,8$ et $\tau_2 = 4\,1\,3\,2\,5\,6\,8\,9\,7$. La permutation τ_1 est séparable, et un de ses arbres de séparation $\mathcal{T}_1$ est donné ci-contre. On détaille le fonctionnement de l'algorithme 4.1 sur cet exemple, pour le nœud V qui est le fils droit de la racine dans $\mathcal{T}_1$. Après renormalisation, on a $\tau_1[V] = 3\,1\,2\,5\,4\,6\,7$, $\tau_1[V_L] = 3\,1\,2$ et $\tau_1[V_R] = 2\,1\,3\,4$. On choisit les valeurs $i = 2, j = 7, a = 2$ et $b = 8$. On veut montrer que pour tout $h \in \{i, \dots j+1\}$ et $c \in \{a, \dots b+1\}$, $M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b)$ est un motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ qui utilise seulement des valeurs entre a et b dans $\tau_2[i..j]$. On fixe par exemple $h = 5$ et $c = 4$.



Par hypothèse d'induction, $M(V_L, i, h-1, a, c-1)$ contient un plus grand motif commun entre

$\tau_1[V_L]$ and $\tau_2[i..h-1]$ utilisant seulement des valeurs entre a et $c-1$ dans $\tau_2[i..h-1]$. Ici, $M(V_L, i, h-1, a, c-1) = 2\,1$. En effet, une occurrence de $2\,1$ dans $\tau_2[i..h-1] = 1\,3\,2$ utilisant des valeurs entre 2 et 3 est $1\,\mathbf{3\,2}$, et une occurrence de $2\,1$ dans $\tau_1[V_L] = 3\,1\,2$ est $\mathbf{3\,1}\,2$. De manière similaire, on a $M(V_R, h, j, c, b) = M(V_R, 5, 7, 4, 8) = 1\,2\,3$, comme le montrent les occurrences $\mathbf{5\,6\,8}$ dans $\tau_2[h..j] = 5\,6\,8$ et $\mathbf{2\,1\,3\,4}$ dans $\tau_1[V_R] = 2\,1\,3\,4$. L'occurrence de $M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) = 2\,1\,3\,4\,5$ dans $\tau_2[i..j]$ utilisant des valeurs entre a et b est alors obtenue en considérant simultanément les deux occurrences dans τ_2 qui ont été sélectionnées ci-dessus. Dans notre cas, une occurrence de $2\,1\,3\,4\,5$ dans $\tau_2[i..j] = 1\,3\,2\,5\,6\,8$ utilisant des valeurs entre 2 et 8 est $1\,\mathbf{3\,2\,5\,6\,8}$. On remarque aussi que l'occurrence $\mathbf{3\,1\,2\,5\,4\,6\,7}$ de $2\,1\,3\,4\,5$ dans $\tau_1[V] = 3\,1\,2\,5\,4\,6\,7$ est elle aussi obtenue en considérant en même temps les occurrences de $2\,1$ et $1\,2\,3$ dans $\tau_1[V_L]$ et $\tau_1[V_R]$ respectivement.

Proposition 4.13. *La complexité en temps de l'algorithme 4.1 est $\mathcal{O}(min(n_1, n_2)n_1n_2^6)$.*

Démonstration. L'algorithme 4.1 manipule un tableau de programmation dynamique M de taille $\mathcal{O}(n_1n_2^4)$, où chaque cellule contient un motif de taille au plus $min(n_1, n_2)$. La complexité en espace est donc au total $\mathcal{O}(min(n_1, n_2)n_1n_2^4)$.

Pour remplir toutes les cases des sous-tableaux $M(V, \_, \_, \_, \_)$ pour toutes les feuilles V de $\mathcal{T}_1$ (lignes 11 à 16 de l'algorithme 4.1), la complexité totale en temps est $\mathcal{O}(n_1n_2^5)$. Et pour tout nœud interne V, le coût du calcul d'une case du tableau $M(V, \_, \_, \_, \_)$ est $\mathcal{O}(min(n_1, n_2)n_2^2)$. Ce coût reflète la complexité de la recherche d'un élément de taille maximale (aux lignes 21 ou 24 de l'algorithme 4.1) dans un ensemble de $\mathcal{O}(n_2^2)$ éléments, la taille de chaque élément étant au plus $\mathcal{O}(min(n_1, n_2))$. Par conséquent, le remplissage de tout le sous-tableau $M(V, \_, \_, \_, \_)$ demande une complexité en temps de l'ordre de $\mathcal{O}(min(n_1, n_2)n_2^6)$. Et comme il y a $\mathcal{O}(n_1)$ nœuds internes dans $\mathcal{T}_1$, on obtient le résultat annoncé. □

Le théorème 4.6 est une conséquence directe des propositions 4.10 et 4.13.

On peut remarquer qu'une amélioration immédiate de la complexité de l'algorithme 4.1 à $\mathcal{O}(n_1n_2^6)$ peut être obtenue en stockant, dans chaque case $M(V, i, j, a, b)$ (si V est un nœud interne), un entier, un signe ($\oplus$ ou $\ominus$) et deux pointeurs, au lieu d'un motif. Plus précisément, si l'algorithme 4.1 remplit la case $M(V, i, j, a, b)$ avec le motif $\rho = M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b)$, alors il suffit de stocker dans $M(V, i, j, a, b)$ la longueur de ρ, le signe $\oplus$, et deux pointeurs vers les cases $M(V_L, i, h-1, a, c-1)$ et $M(V_R, h, j, c, b)$ du tableau M. À la fin de l'algorithme, ce système de pointeurs donne un arbre de séparation d'un plus grand motif commun à τ_1 et τ_2. Le motif associé à cet arbre peut alors être calculé en temps linéaire.

Une amélioration plus substantielle pourrait certainement être obtenue en réduisant le nombre de sous-problèmes à examiner, comme dans l'algorithme d'Ibarra (voir page 78). On pourrait aussi envisager d'utiliser des structures de données plus élaborées permettant de retrouver rapidement un élément de taille maximale dans un tableau, pour diminuer le temps de calcul à chaque passage à la ligne 21 ou 24 de l'algorithme 4.1. Les détails de ces pistes de modification n'ont pas été explorés.

Enfin, on peut conclure en comparant l'algorithme 4.1 à celui de Bose, Buss et Lubiw (algorithme 3.1 page 77). L'algorithme 4.1 résout le problème de la recherche d'un plus grand motif commun entre une permutation quelconque et une permutation séparable, a priori plus difficile que le problème de recherche d'une occurrence d'un motif séparable dans une permutation quelconque, résolu par l'algorithme de Bose, Buss et Lubiw. Pourtant, la méthode utilisée par l'algorithme 4.1 est inspirée de [BBL98], et la complexité de l'algorithme 4.1 est à peine supérieure à celle de l'algorithme 3.1.

4.3 Extension à deux permutations quelconques grâce aux arbres de décomposition

La technique développée par l'algorithme 4.1 peut être étendue à un cadre plus large que celui des permutations séparables. La définition 3.16 (page 75) propose une extension de la notion d'arbre de

séparation à toutes les permutations, à travers les *arbres de décomposition développés*. En utilisant ces arbres, on va décrire, sur le modèle de l'algorithme 4.1, un algorithme pour la recherche de plus grand motif commun à deux permutations, sans contrainte *a priori*.

Cet algorithme n'est pas polynomial en général, mais a une complexité paramétrée par l'arité maximale d'un nœud premier dans l'arbre de décomposition d'une des permutations en entrée. Ainsi, il fonctionne en temps polynomial dès que l'arité des nœuds premiers dans l'arbre de décomposition d'une des permutations en entrée peut être bornée par une constante d indépendante.

Comme dans l'algorithme 4.1, plutôt que deux permutations τ_1 et τ_2, on préfèrera prendre en entrée l'arbre de décomposition développé $\mathcal{T}_1$ de la permutation τ_1, et la permutation τ_2. D'après la proposition 3.17, cela ne modifie la complexité de l'algorithme que par une étape de précalcul linéaire.

De même, on supposera toujours qu'il est plus avantageux de choisir l'arbre de décomposition développé de τ_1 plutôt que celui de τ_2. Cette hypothèse facilite l'écriture de l'algorithme mais ne restreint pas sa généralité. En effet, on peut toujours, dans l'étape de précalcul linéaire, calculer les arbres de décomposition développés de τ_1 et de τ_2, et choisir ensuite, parmi ceux deux arbres, celui où l'arité maximale d'un nœud premier est la plus faible.

4.3.1 Description de l'algorithme

Comme annoncé plus haut, l'algorithme proposé pour la recherche de plus grand motif commun à deux permutations prend en entrée un arbre de décomposition développé $\mathcal{T}_1$ d'une permutation τ_1, et une permutation τ_2. Il résout le problème 4.1 sur l'entrée (τ_1, τ_2), c'est-à-dire qu'il calcule un plus grand motif commun à τ_1 et τ_2. Il fonctionne comme l'algorithme 4.1, avec un cas supplémentaire pour traiter les nœuds premiers de $\mathcal{T}_1$. Le traitement de ce cas supplémentaire est décrit dans l'algorithme 4.2 ci-après.

Algorithme 4.2 Algorithme de recherche de plus grand motif commun à deux permutations.

1: Données : L'arbre de décomposition développé $\mathcal{T}_1$ d'une permutation τ_1 de taille n_1 et une permutation τ_2 de taille n_2

2: Créer un tableau M : suivre la même procédure que dans l'algorithme 4.1

3: Remplir M :
4: **pour** tout nœud V de $\mathcal{T}_1$, depuis les feuilles de $\mathcal{T}_1$ vers la racine **faire**
5: **pour** tous entiers i, j, a et $b \in [1..n_2]$, tels que $i \leq j$ et $a \leq b$ **faire**
6: **si** V est une feuille ou un nœud linéaire **alors**
7: suivre la même procédure que dans l'algorithme 4.1
8: **sinon**
9: /* V est un nœud premier. /*
10: On note ρ la permutation simple qui étiquette le nœud V.
11: On note aussi d l'arité de V, et $V_1, \ldots, V_d$ les fils de V, de gauche à droite.
12: $M(V, i, j, a, b) \leftarrow Longest(S)$ où
$S = \big\{\rho[\pi_1, \pi_2, \ldots, \pi_d] : i = h_0 \leq h_1 \leq \ldots \leq h_d = j+1, a = c_0 \leq c_1 \leq \ldots \leq c_d = b+1$
et $\pi_k = M(V_k, h_{k-1}, h_k - 1, c_{\rho_k - 1}, c_{\rho_k} - 1)$ pour chaque $k \in [1..d]\big\}$
13: **finsi**
14: **fin pour**
15: **fin pour**

16: Sortie : $M(R, 1, n_2, 1, n_2)$, où R désigne la racine de $\mathcal{T}_1$

Dans cet algorithme, il est fait usage à la ligne 12 de la notation $\rho[\pi_1, \pi_2, \ldots, \pi_d]$ qui représente

la *dilatation* de $\rho \in \mathcal{S}_k$ par les permutations $\pi_1, \pi_2, \ldots, \pi_d$, comme définie page 41.

L'idée qui est exploitée par l'algorithme 4.2 est assez simple. Pour un nœud premier V, étiqueté par la permutation simple ρ et ayant d fils $V_1 \ldots V_d$, on remplit la case $M(V, i, j, a, b)$ en partitionnant les intervalles $[i..j]$ et $[a..b]$ en intervalles $I_1, \ldots, I_d$ et $A_1, \ldots, A_d$ respectivement, de telle sorte que $I_p < I_k$ et $A_p < A_k$ si $p < k$ [c]. Ensuite, on concatène les plus grands motifs communs π_k entre les $\tau_1[V_k]$ et τ_2 dans les intervalles d'indices I_k et les intervalles de valeurs A_{ρ_k}. Cette concaténation correspond en fait à une dilatation de ρ par les motifs $\pi_1, \ldots, \pi_d$. Avec les notations de l'algorithme 4.2, $I_k = [h_{k-1}..h_k - 1]$ et $A_k = [c_{k-1}..c_k - 1]$.

4.3.2 Correction et analyse de complexité

Comme dans le paragraphe 4.2, on démontre la correction de l'algorithme 4.2 et on analyse sa complexité.

Proposition 4.14. *L'algorithme 4.2 est correct : il calcule un plus grand motif commun aux deux permutations τ_1 et τ_2 données en entrée.*

Démonstration. La démonstration est semblable à celle de la proposition 4.10

Avec les notations de la preuve de la proposition 4.10, dans le cas d'un nœud premier V, étiqueté par une permutation simple ρ et ayant d fils V_1 à V_d, il existe des entiers $i = h_0 \leq h_1 \leq \ldots \leq h_d = j + 1$ et $a = c_0 \leq c_1 \leq \ldots \leq c_d = b + 1$, tels que π se décompose en $\pi = \rho[\pi^1, \ldots, \pi^d]$, où π^k est un motif commun entre $\tau_1[V_k]$ et $\tau_2[h_{k-1}..h_k - 1]$, qui utilise seulement des valeurs entre $c_{\rho_k - 1}$ et $c_{\rho_k} - 1$ dans $\tau_2[h_{k-1}..h_k - 1]$. Avec cette décomposition de π, on peut utiliser l'hypothèse d'induction sur les nœuds $(V_k)_{1 \leq k \leq d}$ et conclure la preuve comme dans le cas précédent. □

Dans cette preuve, tout repose sur le fait qu'un motif commun entre $\tau_1[V]$ et $\tau_2[i..j]$ est toujours une concaténation de motifs communs entre les fils de V et des sous-séquences de $\tau_2[i..j]$. Cette façon dont les fils d'un nœud dans l'arbre de décomposition héritent de propriétés de leur père (et réciproquement) se retrouve dans d'autres parties de mon travail de thèse, mais aussi (par exemple) dans les travaux de M. H. Albert et M. D. Atkinson [AA05, lemme 15].

La différence principale entre les algorithmes 4.1 et 4.2 réside dans l'analyse de leur complexité. Ces deux algorithmes travaillent sur les tableaux de programmation dynamique de même taille, mais le coût de calcul pour une case de ce tableau peut être bien plus élevé dans l'algorithme 4.2 que dans l'algorithme 4.1. En effet, pour tout nœud interne V, lorsque l'on remplit une case du tableau $M(V, \_, \_, \_, \_)$ avec l'algorithme 4.1, on recherche un motif de taille maximale dans un ensemble contenant $\mathcal{O}(n_2^2)$ éléments. Dans le cas de l'algorithme 4.2, cet ensemble dont il faut extraire un élément de taille maximale contient $\mathcal{O}(n_2^{2d-2})$, si V est un nœud premier d'arité d (ligne 12 de l'algorithme 4.2). La complexité de l'algorithme 4.2 dépend donc d'un paramètre : l'arité maximale d'un nœud premier dans l'arbre de décomposition de la permutation τ_1 en entrée.

Sans hypothèse particulière, la seule borne supérieure que l'on puisse donner sur l'arité maximale d'un nœud premier dans l'arbre de décomposition de la permutation τ_1 de taille n_1 est $d \leq n_1$. Cette borne est atteinte pour les permutations simples. La complexité totale en temps de l'algorithme 4.2 est donc $\mathcal{O}(min(n_1, n_2) n_1 n_2^{2n_1+2})$. Cependant, si on s'intéresse à des classes (ou plus généralement des ensembles) de permutations telles que l'arité de tout nœud premier dans leur arbre de décomposition est bornée par une constante d, alors l'algorithme est polynomial, de complexité $\mathcal{O}(min(n_1, n_2) n_1 n_2^{2d+2})$.

Par exemple, toute classe $\mathcal{C}$ contenant un nombre fini de permutations simples vérifie trivialement cette condition sur l'arité d'un nœud premier, et on peut donc calculer en temps polynomial un plus grand motif commun à deux permutations dont une appartient à $\mathcal{C}$.

Ces considérations de complexité peuvent être résumées de la manière suivante :

[c] La notation $A < B$, signifie que $\forall a \in A, \forall b \in B, a < b$.

Théorème 4.15. *Soit d un entier. On considère la classe $\mathcal{R}$ des permutations pour lesquelles tout nœud premier dans leur arbre de décomposition a arité au plus d* [d]*. Alors la recherche de plus grand motif commun à une permutation de $\mathcal{R}$ et une permutation quelconque est dans P.*

[d] On remarque que $\mathcal{R}$ est une classe close par produit de substitution, où les nœuds premiers autorisés sont tous ceux étiquetés par une permutation simple de taille au plus d.

Chapitre 5

Recherche de plus grand motif commun à un ensemble de permutations

Sommaire

5.1 Algorithme pour la recherche d'un plus grand motif séparable commun à K permutations 96
5.1.1 Description de l'algorithme 96
5.1.2 Correction et analyse de complexité 99
5.2 Complexité de la recherche de plus grand motif séparable commun à un ensemble de permutations 101
5.3 Approximation par motif commun restreint à une classe 104

Le chapitre 4 étudie le problème de la recherche de plus grand motif commun à deux permutations. Un élargissement naturel de ce problème, que l'on envisage ici, est la recherche de plus grand motif commun à un ensemble de permutations, de cardinal arbitraire.

La recherche de plus grand motif commun à un ensemble de permutations est un problème plus difficile que la recherche d'occurrence d'un motif dans une permutation (problème 3.1 page 68) : la NP-complétude du problème 3.1 implique donc que la recherche de plus grand motif commun à un ensemble de permutations est NP-difficile.

Comme dans les chapitres précédents, on étudie des restrictions de ce problème que l'on peut résoudre en temps polynomial. Plutôt que de restreindre l'ensemble de permutations en entrée du problème, on va ici contraindre plutôt le résultat calculé à appartenir à une classe de permutations donnée. On s'intéresse donc au problème suivant, étant donnée une classe $\mathcal{C}$ de permutations :

Problème 5.1 (Recherche de plus grand motif commun appartenant à une classe $\mathcal{C}$)**.**
Données : Un ensemble fini X de permutations.
Sortie : Une permutation σ qui appartient à $\mathcal{C}$, qui est motif de chaque permutation de X, et qui est de taille maximale parmi les permutations satisfaisant ces deux propriétés.

En particulier, on va s'intéresser dans ce chapitre au cas où $\mathcal{C}$ est la classe $\mathcal{S}ep$ des permutations séparables. Comme on l'a remarqué page 86, un plus grand motif commun à deux permutations τ_1 et τ_2, telles que l'une d'entre elles soit séparable, est toujours un motif séparable. Ceci est une conséquence directe du fait que $\mathcal{S}ep$ est une classe de permutations, donc est stable pour la relation de motif. Ainsi, dans la recherche de plus grand motif commun à deux permutations dont une est séparable, on peut sans perte de généralité contraindre le motif calculé à appartenir à la classe $\mathcal{S}ep$. Ce problème, étudié au paragraphe 4.2, est donc un cas particulier du problème 5.1.

Dans le paragraphe 5.1, on donne un algorithme polynomial pour la résolution du problème 5.1 pour $\mathcal{C} = \mathcal{S}ep$, dans le cas où le cardinal de l'ensemble X est borné par une constante K extérieure aux données du problème. Toujours dans le cas où $\mathcal{C} = \mathcal{S}ep$, on verra dans le paragraphe 5.2 que sans borne *a priori* sur la taille de l'ensemble X en entrée, le problème 5.1 est NP-difficile. Enfin, on discutera dans le paragraphe 5.3 de l'approximation d'un plus grand motif commun par un plus grand motif commun contraint à appartenir à une classe $\mathcal{C}$: on verra que quelle que soit la classe $\mathcal{C}$, en calculant un plus grand motif commun appartenant à $\mathcal{C}$, on ne peut pas garantir un résultat de taille plus grande que $\sqrt{Opt}$, si Opt désigne la taille d'un plus grand motif commun, sans contrainte d'appartenance à $\mathcal{C}$.

5.1 Algorithme pour la recherche d'un plus grand motif séparable commun à K permutations

Dans ce paragraphe, on propose un algorithme en temps polynomial pour résoudre le problème 5.1 pour $\mathcal{C} = \mathcal{S}ep$ et dans le cas où l'ensemble X en entrée est de cardinal K. La généralisation au cas où X est de cardinal au plus K est immédiate, mais on ne la détaille pas ici, pour s'affranchir des lourdeurs de notation qu'elle induit.

5.1.1 Description de l'algorithme

Comme dans les algorithmes 3.1 et 4.1, on résout le problème de recherche d'un plus grand motif séparable commun à K permutations avec des techniques de programmation dynamique. Cependant, comme dans ce cas aucune hypothèse de séparabilité n'est faite sur les permutations en entrée, il n'est pas possible de calculer un arbre de séparation pour servir de guide à la division du problème d'origine en sous-problèmes. Pour calculer un plus grand motif séparable commun entre les permutations de l'ensemble X en entrée, on va considérer des sous-problèmes correspondant

à des K-uplets d'intervalles de positions et de valeurs, formés d'une telle paire d'intervalles pour chaque permutations de X.

Plus précisément, on considère K permutations $\tau_1, \ldots, \tau_K$, de tailles respectives $n_1, \ldots, n_K$. On notera n le maximum des $(n_q)_{1 \leq q \leq K}$. Pour calculer un plus grand motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, on utilise un tableau de programmation dynamique M de dimension $4K$, et quand la procédure pour remplir ce tableau s'achève, on s'attend à ce que la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ contienne un motif séparable commun aux permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui, pour chaque $q \in [1..K]$, possèdent une occurrence dans $\tau_q[i_q..j_q]$ qui n'utilise que des valeurs entre a_q et b_q. Si l'on peut calculer les valeurs de toutes les cases du tableau M en temps polynomial, tout en satisfaisant cette propriété, la case $M(1, n_1, 1, n_1, \ldots, 1, n_K, 1, n_K)$ contiendra, à la fin de la procédure, un plus grand motif séparable commun aux permutations $\tau_1, \ldots, \tau_K$.

Algorithme 5.1 Algorithme de recherche d'un plus grand motif séparable commun à un ensemble de K permutations.

Données : K permutations $\tau_1, \ldots, \tau_K$ de tailles respectives $n_1, \ldots, n_K$

Créer un tableau M :
pour tous entiers i_q, j_q, a_q et $b_q \in [1..n_q]$, pour tout $q \in [1..K]$ **faire**
  $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K) \leftarrow \epsilon$
fin pour

Remplir M :
pour tous entiers i_q, j_q, a_q et $b_q \in [1..n_q]$, tels que $i_q \leq j_q$ et $a_q \leq b_q$, pour tout $q \in [1..K]$, par valeurs croissantes de $\sum_q (j_q - i_q) + (b_q - a_q)$ **faire**
  si $\exists q \in [1..K]$ tel que $i_q = j_q$ ou $a_q = b_q$ **alors**
    si $\forall q \in [1..K], \exists h_q \in [i_q..j_q]$ tel que $\tau_q(h_q) \in [a_q..b_q]$ **alors**
      $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K) \leftarrow 1$
    sinon
      $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K) \leftarrow \epsilon$
    finsi
  sinon
    /*$\forall q \in [1..K], i_q < j_q$ et $a_q < b_q$ /*
    $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K) \leftarrow Longest(S_\oplus \cup S_\ominus \cup S)$ avec

$$
\begin{aligned}
S_\oplus &= \{M(i_1, h_1 - 1, a_1, c_1 - 1, \ldots, i_K, h_K - 1, a_K, c_K - 1) \oplus M(h_1, j_1, c_1, b_1, \\
&\quad \ldots, h_K, j_K, c_K, b_K) : i_q < h_q \leq j_q, a_q < c_q \leq b_q, \forall q \in [1..K]\} \\
S_\ominus &= \{M(i_1, h_1 - 1, c_1, b_1, \ldots, i_K, h_K - 1, c_K, b_K) \ominus M(h_1, j_1, a_1, c_1 - 1, \\
&\quad \ldots, h_K, j_K, a_K, c_K - 1) : i_q < h_q \leq j_q, a_q < c_q \leq b_q, \forall q \in [1..K]\} \\
S &= \{1\} \text{ si } \forall q \in [1..K], \exists h_q \in [i_q..j_q] \text{ tel que } \tau_q(h_q) \in [a_q..b_q], \\
&= \{\epsilon\} \text{ sinon.}
\end{aligned}
$$

  finsi
fin pour

Sortie : $M(1, n_1, 1, n_1, \ldots, 1, n_K, 1, n_K)$

L'algorithme 5.1 précise comment remplir le tableau M en temps polynomial. Dans cet algorithme, *Longest* est comme avant la procédure linéaire de calcul d'un élément de taille maximale dans un ensemble. On reprend aussi du paragraphe 4.2 la notation ϵ pour la permutation de taille

0. Enfin, les notations $\oplus$ et $\ominus$ désignent respectivement les concaténations positive et négative de motifs, définies page 88.

Avant de démontrer la correction de l'algorithme 5.1, on donne deux lemmes qui expliquent comment les motifs séparables communs peuvent être fusionnés, ou au contraire séparés, pour faire apparaître d'autres motifs séparables communs. On sera aussi attentif à conserver la propriété de taille maximale d'un motif séparable commun, dans le cas de la séparation d'un motif en deux.

Lemme 5.2. *Soit π_1 un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$ qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$ (resp. $[c_q..b_q]$).*

Soit π_2 un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$ qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[h_q..j_q]$ et dans l'intervalle de valeurs $[c_q..b_q]$ (resp. $[a_q..c_q - 1]$).

Alors $\pi = \pi_1 \oplus \pi_2$ (resp. $\pi = \pi_1 \ominus \pi_2$) est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$ qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$.

Démonstration. On démontre le lemme 5.2 pour $\pi = \pi_1 \oplus \pi_2$ (le cas $\pi = \pi_1 \ominus \pi_2$ étant similaire).

On fixe un entier $q \in [1..K]$. Par hypothèse, il existe des occurrences de π_1 et de π_2 dans τ_q, l'occurrence de π_1 utilisant l'intervalle de positions $[i_q..h_q - 1]$ et l'intervalle de valeurs $[a_q..c_q - 1]$, et l'occurrence de π_2 utilisant l'intervalle de positions $[h_q..j_q]$ et l'intervalle de valeurs $[c_q..b_q]$. On s'aperçoit alors sans difficulté (voir figure 5.1) que tous les éléments utilisés dans une de ces occurrences de π_1 et π_2 forment une occurrence du motif $\pi = \pi_1 \oplus \pi_2$ dans τ_q qui utilise l'intervalle de positions $[i_q..j_q]$ et l'intervalle de valeurs $[a_q..b_q]$. Cet argument étant valable pour chaque valeur de $q \in [1..K]$, on en déduit que π est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$ qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. □

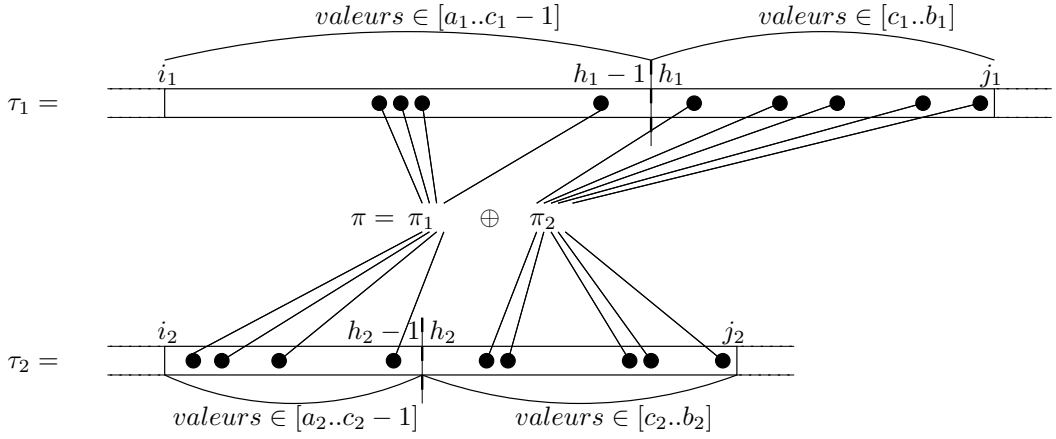


FIG. 5.1 – Fusion et séparation de motifs séparables qui possèdent des occurrences dans $K = 2$ permutations, dans des intervalles de positions et de valeurs prescrits.

Lemme 5.3. *Soit π un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$.*

Si $\pi = \pi_1 \oplus \pi_2$ (resp. $\pi = \pi_1 \ominus \pi_2$), où π_1 et π_2 sont des motifs séparables non vides*, alors il existe des positions $(h_q)_{q \in [1..K]}$ et des valeurs $(c_q)_{q \in [1..K]}$, avec $i_q < h_q \leq j_q$, $a_q < c_q \leq b_q, \forall q \in [1..K]$, telles que :*

1. *π_1 est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$ (resp. $[c_q..b_q]$), et*
2. *π_2 est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[h_q..j_q]$ et dans l'intervalle de valeurs $[c_q..b_q]$ (resp. $[a_q..c_q - 1]$).*

Démonstration. Comme pour le lemme 5.2, on considère seulement le cas $\pi = \pi_1 \oplus \pi_2$, mais la preuve est similaire si $\pi = \pi_1 \ominus \pi_2$.

On fixe un entier $q \in [1..K]$. Par hypothèse, $\pi = \pi_1 \oplus \pi_2$ a un occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et l'intervalle de valeurs $[a_q..b_q]$. Par définition de la concaténation positive, cette occurrence se partage en deux occurrences de π_1 et π_2 respectivement (voir figure 5.1). Plus précisément, il existe des entiers $h_q \in [i_q + 1..j_q]$ et $c_q \in [a_q + 1..b_q]$ tels que π_1 (resp. π_2) a une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ (resp. $[h_q..j_q]$) et l'intervalle de valeurs $[a_q..c_q - 1]$ (resp. $[c_q..b_q]$). Cet argument s'appliquant pour chaque entier $q \in [1..K]$, il apparaît clairement que π_1 (resp. π_2) est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ (resp. $[h_q..j_q]$) et dans l'intervalle de valeurs $[a_q..c_q - 1]$ (resp. $[c_q..b_q]$). Il reste à démontrer que π_1 et π_2 sont de taille maximale parmi les motifs satisfaisant cette propriété.

Pour ce faire, on raisonne par l'absurde. On suppose donc que π_1 n'est *pas* de taille maximale parmi les motifs séparables communs à $\tau_1, \ldots, \tau_K$, qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$. Dans ce cas, il existe un motif π_1^{long}, qui est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$, et tel que $|\pi_1^{long}| > |\pi_1|$. Le lemme 5.2 assure qu'alors $\pi_1^{long} \oplus \pi_2$ est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. Et comme $|\pi_1^{long} \oplus \pi_2| > |\pi_1 \oplus \pi_2| = |\pi|$, ceci contredit que π est de taille maximale. Ainsi, on conclut que π_1 est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$. De la même façon, on démontre que π_2 est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[h_q..j_q]$ et dans l'intervalle de valeurs $[c_q..b_q]$, ce qui achève la preuve du lemme 5.3. □

5.1.2 Correction et analyse de complexité

On dispose à présent des outils nécessaires pour démontrer la correction de l'algorithme 5.1 et analyser sa complexité.

Proposition 5.4. *L'algorithme 5.1 est correct : il calcule un plus grand motif séparable commun aux K permutations en entrée.*

Démonstration. On s'intéresse au tableau M calculé par l'algorithme 5.1. On montre par récurrence sur $\sum_q (j_q - i_q) + (b_q - a_q)$ que la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ contient un motif séparable commun π entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$.

Avant cela, on remarque que dans le cas où il existe $q \in [1..K]$ tel que $i_q > j_q$ ou $a_q > b_q$, alors la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ contient ϵ et on a $[i_q..j_q] = \emptyset$ ou $[a_q..b_q] = \emptyset$ pour cette valeur de $q \in [1..K]$. La propriété annoncée est donc clairement vraie dans ce cas, et on supposera dans la suite que $i_q \leq j_q$ et $a_q \leq b_q$ pour tout $q \in [1..K]$.

Si $\sum_q (j_q - i_q) + (b_q - a_q) = 0$, alors $i_q = j_q$ et $a_q = b_q$ pour tout $q \in [1..K]$. Par conséquent, le motif que l'on voudrait voir figurer dans la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ est un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui utilisent, pour tout $q \in [1..K]$, seulement la position $i_q = j_q$ et la valeur $a_q = b_q$ dans τ_q. Un tel motif vaut soit ϵ soit 1, et il vaut 1 si et seulement si $\forall q \in [1..K], \tau_q(i_q) = a_q$ c'est-à-dire si et seulement si $\forall q \in [1..K], \exists h_q \in [i_q..j_q]$ tel que $\tau_q(h_q) \in [a_q..b_q]$. On vérifie aux lignes 8 à 12 de l'algorithme 5.1 que dans ce cas, le motif calculé dans la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ est conforme à ce qui a été annoncé.

Si $\sum_q (j_q - i_q) + (b_q - a_q) > 0$, on considère deux sous-cas :

Si $\exists q \in [1..K]$ tel que $i_q = j_q$ ou $a_q = b_q$, on note π un motif séparable commun entre les permutations $\tau_1, \ldots, \tau_K$, de taille maximale parmi ceux qui possèdent, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. Alors, comme précédemment, π est soit 1 soit ϵ, puisque dans au moins une des permutations, disons τ_q, l'occurrence de π peut utiliser au plus une position (si $i_q = j_q$) ou au plus une valeur (si $a_q = b_q$). Là encore, les lignes 8 à 12 de l'algorithme 5.1 assurent que $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$ contient soit 1 soit ϵ. Plus précisément, par la condition de la ligne 9, cette case contient 1 si et seulement si 1 a, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$.

Il reste à traiter le cas récursif de la récurrence, lorsque $\forall q \in [1..K], i_q < j_q$ et $a_q < b_q$. Dans ce cas, on considère un motif séparable π de taille maximale parmi ceux qui apparaissent, pour tout $q \in [1..K]$, dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. Comme π est séparable, on a soit $\pi = \epsilon$, soit $\pi = 1$, soit $\pi = \pi_1 \oplus \pi_2$, soit $\pi = \pi_1 \ominus \pi_2$, π_1 et π_2 étant des motifs séparables plus petits que π, mais non vides. On considère alors un motif, noté π_{algo}, de taille maximale dans l'ensemble $S_\oplus \cup S_\ominus \cup S$ défini par :

$$\begin{aligned}
S_\oplus &= \{M(i_1, h_1 - 1, a_1, c_1 - 1, \ldots, i_K, h_K - 1, a_K, c_K - 1) \oplus M(h_1, j_1, c_1, b_1, \\
&\quad \ldots, h_K, j_K, c_K, b_K) : i_q < h_q \leq j_q, a_q < c_q \leq b_q, \forall q \in [1..K]\} \\
S_\ominus &= \{M(i_1, h_1 - 1, c_1, b_1, \ldots, i_K, h_K - 1, c_K, b_K) \ominus M(h_1, j_1, a_1, c_1 - 1, \\
&\quad \ldots, h_K, j_K, a_K, c_K - 1) : i_q < h_q \leq j_q, a_q < c_q \leq b_q, \forall q \in [1..K]\} \\
S &= \{1\} \text{ si } \forall q \in [1..K], \exists h_q \in [i_q..j_q] \text{ tel que } \tau_q(h_q) \in [a_q..b_q], \\
&= \{\epsilon\} \text{ sinon.}
\end{aligned}$$

Par hypothèse de récurrence, chaque case de M qui apparaît dans la définition d'un des deux ensembles $S_\oplus$ et $S_\ominus$ est un motif séparable de taille maximale parmi ceux qui possèdent, pour chaque $q \in [1..K]$, une occurrence dans τ_q qui n'utilise que des positions et des valeurs dans les intervalles imposés. Par sa définition, on voit aussi que l'ensemble S contient 1 si et seulement si 1 a, pour chaque $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. Avec ces deux remarques et le lemme 5.2, on déduit immédiatement que π_{algo} est un motif séparable qui possède, pour tout $q \in [1..K]$, une occurrence dans τ_q dans l'intervalle de positions $[i_q..j_q]$ et dans l'intervalle de valeurs $[a_q..b_q]$. Il suffit donc de démontrer que $|\pi_{algo}| = |\pi|$. Ceci étant clair lorsque $\pi = \epsilon$ ou 1, on suppose que $\pi = \pi_1 \oplus \pi_2$, le cas $\pi = \pi_1 \ominus \pi_2$ pouvant évidemment être traité de la même façon.

Comme $\pi = \pi_1 \oplus \pi_2$ a une occurrence dans chaque τ_q dans l'intervalle de positions $[i_q..j_q]$ et l'intervalle de valeurs $[a_q..b_q]$, le lemme 5.3 assure qu'il existe des positions $(h_q)_{q \in [1..K]}$ et des valeurs $(c_q)_{q \in [1..K]}$, vérifiant $i_q < h_q \leq j_q, a_q < c_q \leq b_q, \forall q \in [1..K]$, et telles que π_1 a une occurrence dans chaque τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et dans l'intervalle de valeurs $[a_q..c_q - 1]$, et π_2 a une occurrence dans chaque τ_q dans l'intervalle de positions $[h_q..j_q]$ et dans l'intervalle de valeurs $[c_q..b_q]$. Par le même lemme 5.3, on sait aussi que π_1 et π_2 sont des motifs de taille maximale parmi les motifs séparables qui apparaissent, pour tout $q \in [1..K]$, dans τ_q dans les intervalles requis de positions et de valeurs. Par hypothèse de récurrence, on a donc $|M(i_1, h_1 - 1, a_1, c_1 - 1, \ldots, i_K, h_K - 1, a_K, c_K - 1)| = |\pi_1|$ et $|M(h_1, j_1, c_1, b_1, \ldots, h_K, j_K, c_K, b_K)| = |\pi_2|$. On en déduit que $|\pi| = |\pi_1 \oplus \pi_2| =$

$|M(i_1, h_1-1, a_1, c_1-1, \ldots, i_K, h_K-1, a_K, c_K-1)| + |M(h_1, j_1, c_1, b_1, \ldots, h_K, j_K, c_K, b_K)| \leq |\pi_{algo}|$. L'inégalité $|\pi| \geq |\pi_{algo}|$ étant évidente puisque π est de taille maximale, on conclut que $|\pi| = |\pi_{algo}|$, achevant la preuve dans le cas où $\pi = \pi_1 \oplus \pi_2$.

Dans le cas $\pi = \pi_1 \ominus \pi_2$, la démonstration se déroule de la même manière, à la différence près que π_1 a une occurrence dans chaque τ_q dans l'intervalle de positions $[i_q..h_q - 1]$ et l'intervalle de valeurs $[c_q..b_q]$ et π_2 a une occurrence dans chaque τ_q dans l'intervalle de positions $[h_q..j_q]$ et l'intervalle de valeurs $[a_q..c_q - 1]$. □

Proposition 5.5. *L'algorithme 5.1 fonctionne en temps $\mathcal{O}(n^{6K+1})$ et en espace $\mathcal{O}(n^{4K+1})$.*

Démonstration. L'algorithme 5.1 utilise un tableau de programmation dynamique M de taille $\mathcal{O}(n^{4K})$, où chaque case contient un motif de taille au plus n, de telle sorte que la complexité en espace est $\mathcal{O}(n^{4K+1})$. Pour remplir une case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$, si $\exists q \in [1..K]$ tel que $i_q = j_q$ ou $a_q = b_q$ (lignes 9 à 13 de l'algorithme 5.1), la complexité temporelle est $\mathcal{O}(n^K)$. S'il n'existe pas de tel entier q (ligne 15 de l'algorithme 5.1), le temps nécessaire à remplir la case $M(i_1, j_1, a_1, b_1, \ldots, i_K, j_K, a_K, b_K)$, en utilisant les cases de M calculées avant, est $\mathcal{O}(n^{2K+1})$. En effet, on recherche un élément de taille maximale parmi $\mathcal{O}(n^{2K})$ éléments, chacun d'eux étant calculé en temps $\mathcal{O}(n)$ par concaténation de deux valeurs déjà stockées dans le tableau M. Ainsi, la complexité en temps obtenue pour remplir M complètement est $\mathcal{O}(n^{6K+1})$. □

On déduit des propositions 5.4 et 5.5 que :

Théorème 5.6. *Pour toute valeur fixée d'un paramètre K, le problème de recherche d'un plus grand motif séparable commun à K permutations est dans P.*

On peut remarquer que dans le cas où le nombre de permutations en entrée est fini, mais sans être connu à l'avance, on peut encore appliquer l'algorithme 5.1, mais sa complexité n'est plus polynomiale. On montre en fait dans le paragraphe 5.2 que le problème de recherche d'un plus grand motif séparable commun à un ensemble de permutations de cardinal arbitraire est NP-difficile.

On peut aussi se demander si un plus grand motif séparable commun à deux permutations τ_1 et τ_2 (calculé en temps polynomial par l'algorithme 5.1) est une bonne approximation d'un plus grand motif commun entre τ_1 et τ_2, sans contrainte de séparabilité (qui est NP-difficile à calculer). Le paragraphe 5.3 donne une réponse négative à cette question, comme cas particulier du corollaire 5.13.

5.2 Complexité de la recherche de plus grand motif séparable commun à un ensemble de permutations

On revient à l'étude du problème 5.1 pour la classe $\mathcal{C} = \mathcal{S}ep$, mais sans borne *a priori* sur le cardinal de l'ensemble de permutations X en entrée du problème. On va voir que ce problème est NP-difficile. On montre en fait que le problème de décision associé est NP-difficile.

Problème 5.7 (Décision de l'existence d'un motif commun séparable de taille donnée)**.**
Données : Un ensemble fini X de permutations et un entier k.
Sortie : Un booléen qui indique s'il existe une permutation séparable σ de taille k qui est motif de chaque permutation de X.

En fait, on démontre que le problème 5.7 est NP-complet dans le cas restreint où X est un ensemble de permutations *séparables*. Cela impliquera que les problèmes 5.7, et donc 5.1, sont NP-difficiles.

Théorème 5.8. *Le problème 5.7, dont les entrées sont restreintes aux ensembles X ne contenant que des permutations séparables, est NP-complet.*

Démonstration. La preuve est par réduction du problème du stable (voir définition page 34) dans un graphe.

Problème 5.9 (Décision de l'existence d'un stable de taille donnée)**.**
DONNÉES : Un graphe $G = (V, E)$ et un entier k.
SORTIE : Un booléen qui indique s'il existe un stable de taille k dans le graphe G.

Le problème 5.9 est bien connu pour être NP-complet [GJ79].

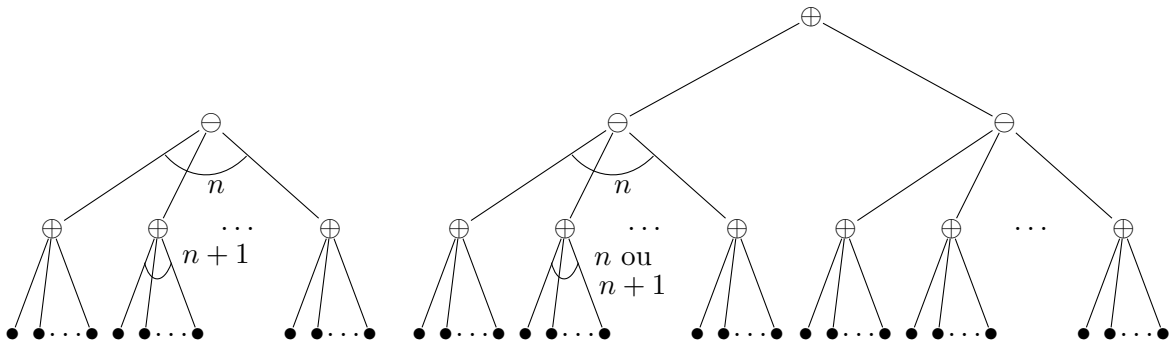
On considère un graphe $G = (V, E)$ instance du problème 5.9. On note $V = \{1, 2, \ldots, n\}$. On construit alors un ensemble X de $n + 1$ permutations séparables $\tau_0, \tau_1, \ldots, \tau_n$ qui sera instance du problème 5.7.

Dans toute la preuve, par souci de concision, on notera pour tout p, $\mathcal{I}_p = 12 \ldots p$ la permutation identité de taille p, et $\mathcal{D}_p = p \ldots 21$ son miroir.

On définit $\tau_0 = \mathcal{D}_n[\alpha_{0,1}, \ldots, \alpha_{0,n}]$ où $\alpha_{0,i} = \mathcal{I}_{n+1}$ pour tout $i \in [1..n]$. On pose aussi , pour tout $i \in [1..n]$, $\tau_i = 12[\gamma_i, \delta_i]$ avec $\gamma_i = \mathcal{D}_n[\alpha_{i,1}, \ldots, \alpha_{i,n}]$ et $\delta_i = \mathcal{D}_n[\beta_{i,1}, \ldots, \beta_{i,n}]$ où les $\alpha_{i,j}$ et $\beta_{i,j}$ sont définis par :

$$\alpha_{i,j} = \begin{cases} \mathcal{I}_{n+1} & \text{si } i \neq j \\ \mathcal{I}_n & \text{si } i = j \end{cases} \qquad \text{et} \qquad \beta_{i,j} = \begin{cases} \mathcal{I}_{n+1} & \text{si } (i,j) \notin E \\ \mathcal{I}_n & \text{si } (i,j) \in E. \end{cases}$$

Les permutations τ_i ainsi construites sont séparables, et la figure 5.2 représente la forme de leurs arbres de décomposition.



Arbre de décomposition de τ_0 Arbre de décomposition de τ_i pour $i \in [1..n]$

FIG. 5.2 – Les arbres de décomposition des permutations τ_i dans la réduction du problème du stable au problème de recherche de motif séparable commun.

On vérifie facilement que cette construction demande un temps polynomial, puisque chaque permutation τ_i est de taille au plus $2n(n + 1)$ et demande un temps de construction linéaire en sa taille. On démontre ensuite que G contient un stable S de taille k si et seulement s'il existe un motif σ de taille $n^2 + k$ qui a une occurrence dans chacune des permutations $\tau_0, \tau_1, \ldots, \tau_n$. Comme les permutations τ_i sont séparables, ce motif σ est nécessairement lui-aussi séparable et ainsi donne bien une réponse au problème 5.7 sur l'entrée $X = \{\tau_0, \tau_1, \ldots, \tau_n\}$.

Exemple 5.10. On illustre la réduction proposée sur l'exemple du graphe G à $n = 4$ sommets représenté en figure 5.3. Le graphe G contient deux stables de taille 2 : les ensembles de sommets $\{1, 4\}$ (entourés en rouge pointillé) et $\{2, 4\}$ (entourés en trait plein bleu). L'ensemble $X = \{\tau_0, \tau_1, \ldots, \tau_4\}$ construit dans la réduction est défini par

$$\begin{aligned} \tau_0 &= 4321[\mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_5] \\ &= 16\,17\,18\,19\,20\,11\,12\,13\,14\,15\,6\,7\,8\,9\,10\,1\,2\,3\,4\,5, \text{ et} \end{aligned}$$

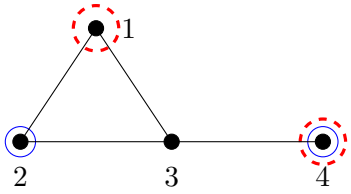


FIG. 5.3 – Le graphe G utilisé comme exemple à la réduction du problème du stable à celui de recherche de motif séparable commun.

$$\begin{aligned}
\tau_1 = 12[\gamma_1, \delta_1] \quad &\text{avec} \quad \gamma_1 = 4321[\mathcal{I}_4, \mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_5] \\
&\qquad\quad \delta_1 = 4321[\mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_4, \mathcal{I}_5], \\
\tau_2 = 12[\gamma_2, \delta_2] \quad &\text{avec} \quad \gamma_2 = 4321[\mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_5, \mathcal{I}_5] \\
&\qquad\quad \delta_2 = 4321[\mathcal{I}_4, \mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_5], \\
\tau_3 = 12[\gamma_3, \delta_3] \quad &\text{avec} \quad \gamma_3 = 4321[\mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_5] \\
&\qquad\quad \delta_3 = 4321[\mathcal{I}_4, \mathcal{I}_4, \mathcal{I}_5, \mathcal{I}_4], \\
\tau_4 = 12[\gamma_4, \delta_4] \quad &\text{avec} \quad \gamma_4 = 4321[\mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_4] \\
&\qquad\quad \delta_4 = 4321[\mathcal{I}_5, \mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_5].
\end{aligned}$$

Chacune des permutations τ_i, pour $i \in [1..4]$, est de taille inférieure ou égale à $2n(n+1) = 40$. À un stable de G de taille 2 correspond un motif de taille $n^2 + 2 = 18$, qui possède une occurrence dans chacune des permutations τ_i pour $i \in [0..4]$. Par exemple, pour le stable $\{1, 4\}$, le motif dont il est question est $\sigma = 4321[\mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_4, \mathcal{I}_5]$, qui possède une occurrence dans τ_0, et dans $\delta_1, \gamma_2, \gamma_3$ et δ_4. Pour le stable $\{2, 4\}$, on aurait $\sigma = 4321[\mathcal{I}_4, \mathcal{I}_5, \mathcal{I}_4, \mathcal{I}_5]$, qui est motif de $\tau_0, \gamma_1, \delta_2, \gamma_3$ et δ_4.

Sur l'exemple proposé, on constate la correspondance entre stable de G et motif commun aux permutations $\tau_0, \tau_1, \ldots, \tau_n$. On démontre maintenant dans le cas général qu'il y a équivalence entre l'existence d'un stable de taille k dans G et celle d'un motif commun aux permutations $\tau_0, \tau_1, \ldots, \tau_n$ de taille $n^2 + k$.

On suppose donc que G contient un stable S de taille k. On définit alors le motif $\sigma = \mathcal{D}_n[\rho_1, \ldots, \rho_n]$ où les ρ_i sont définis par :

$$\rho_i = \begin{cases} \mathcal{I}_{n+1} & \text{si } i \in S \\ \mathcal{I}_n & \text{si } i \notin S. \end{cases}$$

Le motif σ est évidemment de taille $n^2 + k$, et on démontre qu'il est motif de chacune des permutations $(\tau_i)_{0 \le i \le n}$. On voit immédiatement que $\sigma \preceq \tau_0$. Pour chaque $i \in S$, les voisins j de i n'appartiennent pas à S, et donc σ a une occurrence dans δ_i. Pour les $i \in [1..n]$ qui n'appartiennent pas à S, $\sigma \preceq \gamma_i$ (puisque tous les $\alpha_{i,j}$ valent $\mathcal{I}_{n+1}$, sauf $\alpha_{i,i}$ qui vaut $\mathcal{I}_n$). Dans les deux cas, $\sigma \preceq \tau_i$, ce qui démontre la première partie de l'équivalence.

On suppose maintenant qu'il existe un motif σ de taille n^2+k qui a une occurrence dans chacune des permutations $\tau_0, \tau_1, \ldots, \tau_n$. On démontre alors les propriétés suivantes :

(i) σ est de la forme $\mathcal{D}_n[\rho_1, \ldots, \rho_n]$ pour des permutations $\rho_i \in \{\mathcal{I}_d : 1 \le d \le n+1\}$.
(ii) En outre, pour chaque $i \in [1..n]$, σ possède une occurrence dans γ_i ou dans δ_i.

Démonstration. Comme σ a une occurrence dans τ_0, σ est forcément soit une permutation identité, soit un miroir d'une identité, soit la dilatation par des $\mathcal{I}_d$ d'un miroir d'une identité. Comme

$|\sigma| = n^2 + k$, les premiers cas sont exclus, et dans le troisième cas, il y a forcément n permutations identités, ou σ serait de taille au plus $(n+1)(n-1) = n^2 - 1 < n^2 + k$. Ceci prouve le point (i).

On prouve le point (ii) par l'absurde. On suppose donc disposer d'une occurrence de σ dans τ_i qui utilise un élément a dans γ_i et un élément b dans δ_i. Alors par définition de τ_i, $a < b$ et les deux éléments a_σ et b_σ correspondant à a et b dans σ sont en ordre croissant. Ils appartiennent donc à un même bloc croissant dans σ, disons ρ_j. Et pour tout élément x de γ_i (resp. δ_i), x et b (resp. a et x) sont aussi en ordre croissant, et donc l'occurrence de σ dans τ_i est une permutation identité. Ceci contredit le point (i), et termine la preuve. □

On définit l'ensemble $S \subseteq V$ par $S = \{i : \rho_i = \mathcal{I}_{n+1}\}$. Les ρ_i pour $i \notin S$ sont donc de taille au plus n, et comme σ est de taille $n^2 + k$, il est nécessaire que S contienne au moins k éléments. On démontre que S un stable dans G (dont on peut donc extraire un stable de taille k). Pour ce faire, on considère $i \in S$, et l'occurrence de σ dans τ_i. Par le point (ii), σ a en fait une occurrence soit dans γ_i soit dans δ_i. Comme $i \in S$, $\rho_i = \mathcal{I}_{n+1}$, et donc l'occurrence de σ dans τ_i est dans δ_i (elle ne peut pas être dans γ_i puisque $\alpha_{i,i} = \mathcal{I}_n$). Par définition, pour les sommets j de S, $\rho_j = \mathcal{I}_{n+1}$, et il est donc nécessaire que $\beta_{i,j} = \mathcal{I}_{n+1}$. Cela se produit exactement pour les sommets j qui ne sont pas voisins de i, et ainsi on déduit que i ne possède aucun voisin dans S. Cet argument pouvant être appliqué à tous les sommets de S, on conclut que S est un stable, achevant ainsi la preuve du théorème 5.8. □

5.3 Approximation par motif commun restreint à une classe

On a vu que le problème 4.1 de recherche de plus grand motif commun à deux permutations est NP-difficile. Mais les résultats du paragraphe 5.1 assurent que ce problème restreint aux motifs communs qui sont séparables est polynomial. Une question naturelle est alors de savoir si un plus grand motif séparable commun à deux permutations (calculable en temps polynomial) est une bonne approximation de leur plus grand motif commun. Plus généralement, on peut se poser cette question lorsque le motif commun cherché est contraint d'appartenir à une classe de permutations $\mathcal{C}$ quelconque (voir le problème 5.1).

On démontre ici que cette approximation d'un plus grand motif commun par un motif appartenant à une classe est assez mauvaise, pour toute classe $\mathcal{C}$ (corollaire 5.13). Ce résultat est obtenu en étudiant la taille du plus grand motif appartenant à $\mathcal{C}$ qui possède une occurrence dans une permutation arbitraire (théorème 5.12). Pour cela, il nous faut d'abord analyser le nombre de permutations d'une taille fixée qui contiennent une occurrence d'un motif donné :

Lemme 5.11. *Pour toute permutation $\sigma \in \mathcal{S}_k$, le nombre de permutations $\tau \in \mathcal{S}_n$ telles que σ est un motif de τ est au plus $(n-k)!\binom{n}{k}^2$.*

Démonstration. Soient $\sigma = \sigma_1\sigma_2\dots\sigma_k$ et $\tau = \tau_1\tau_2\dots\tau_n$ deux permutations telles que σ est un motif de τ. Il existe des positions $i_1 < i_2 < \dots < i_k$ telles que $\tau_{i_1}\tau_{i_2}\dots\tau_{i_k}$ est une sous-séquence de τ isomorphe en ordre à σ. Alors τ se décompose en $\tau = u_1\tau_{i_1}u_2\tau_{i_2}\dots u_k\tau_{i_k}u_{k+1}$ où u_i est une fenêtre d'éléments consécutifs dans τ.

À l'inverse, supposons que soient données une permutation $\sigma \in S_k$, et des séquences $u_1, u_2, \dots, u_{k+1}$ de $k+1$ fenêtres, formées d'entiers tous distincts de $\{1\dots n\}$, et dont les tailles somment à $n-k$. On définit $E = \{n_1, n_2, \dots n_k\}$ l'ensemble des k entiers de $\{1\dots n\}$ qui n'apparaissent dans aucun des u_i. On note enfin $\bar{j}$ le j-ème plus petit élément de E. Alors la permutation $\tau = u_1\overline{\sigma_1}u_2\overline{\sigma_2}\dots\overline{\sigma_k}u_{k+1}$ est une permutation de $\mathcal{S}_n$ dont σ est motif. Par exemple, si $\sigma = 2143$, $n = 9$, $u_1 = 31$, $u_2 = \varnothing$, $u_3 = 8$, $u_4 = 65$ et $u_5 = \varnothing$, on a $E = \{2, 4, 7, 9\}$ et $\tau = 31\mathbf{4}2\mathbf{8}9\mathbf{6}5\mathbf{7}$.

On peut remarquer que plusieurs suites de u_i peuvent aboutir à la même permutation τ si σ possède plusieurs occurrences dans τ, comme illustré par la figure 5.4.

Cependant, cela permet de majorer le nombre de permutations τ qui contiennent le motif σ par le nombre de suites de fenêtres u_i possibles. Il y a $\binom{n}{n-k}$ choix pour les valeurs des éléments

$$\sigma = 2431 \text{ et } (u_1, u_2, u_3, u_4) = \begin{cases} (1, \emptyset, \emptyset, 56, \emptyset) \\ (1, \emptyset, 4, 6, \emptyset) \\ (1, \emptyset, 45, \emptyset, \emptyset) \end{cases} \implies \tau = 1374562$$

FIG. 5.4 – Plusieurs décompositions de τ en $u_1\tau_{i_1}u_2\tau_{i_2}\dots u_k\tau_{i_k}u_{k+1}$, lorsque σ a plusieurs occurrences $\tau_{i_1}\dots\tau_{i_k}$ dans τ.

apparaissant dans l'un des u_i, et $(n-k)!$ façon d'ordonner ces valeurs, par un mot w de taille $n-k$ utilisant exactement une fois chaque valeur choisie. La dernière étape est de couper le mot w en $k+1$ fenêtres $u_1, u_2, \dots u_{k+1}$, éventuellement vides. Il y a $\binom{n}{k} = \binom{n}{n-k}$ tels découpages de w, ce qui prouve le lemme 5.11. □

Avec ce lemme, on démontre facilement qu'il peut exister un écart quadratique entre la taille d'une permutation et la taille du plus grand motif appartenant à une classe $\mathcal{C}$ qu'elle contient :

Théorème 5.12. *Pour tout $\epsilon > 0$ et toute classe de permutations $\mathcal{C}$, il existe une suite $(\tau_n)_{n\in\mathbb{N}}$ de permutations $\tau_n \in \mathcal{S}_n$ telles que*

$$|\sigma_n| = o\left(n^{0.5+\epsilon}\right)$$

où σ_n est un plus grand motif appartenant à la classe $\mathcal{C}$ qui possède une occurrence dans τ_n.

Démonstration. On démontre ce théorème par l'absurde. On démontre d'abord que si le résultat annoncé était faux, toute permutation de taille n contiendrait un motif de $\mathcal{C}$ de taille *exactement* $k = \lceil n^{0.5+\epsilon} \rceil$. Ensuite, on montre que le nombre de permutations de taille n contenant un motif de $\mathcal{C} \bigcap \mathcal{S}_k$ est strictement inférieur à $n!$.

On suppose donc qu'il existe $\epsilon > 0$ et une classe de permutations $\mathcal{C}$ tels que pour toute permutation $\tau \in \mathcal{S}_n$, le plus grand motif $\sigma \in \mathcal{C}$ dans τ est de taille $|\sigma| \geq \lceil |\tau|^{0.5+\epsilon} \rceil = k$. Comme $\mathcal{C}$ est par définition fermée pour la relation $\preceq$ de motif, pour toute permutation $\tau \in \mathcal{S}_n$, il existe un motif $\sigma \in \mathcal{C}$ dans τ dont la taille est *exactement* $|\sigma| = k$.

Mais, d'après le théorème de Marcus et Tardos (théorème 1.33 page 21), il existe une constante c telle que le nombre de permutations dans $\mathcal{C} \bigcap \mathcal{S}_k$ est au plus c^k. Par le lemme 5.11, pour toute permutation de $\mathcal{C} \bigcap \mathcal{S}_k$, le nombre de permutations de $\mathcal{S}_n$ qui contiennent cette permutation en tant que motif est au plus $(n-k)!\binom{n}{k}^2$. Ainsi, le nombre de permutations de $\mathcal{S}_n$ ayant un motif dans $\mathcal{C} \bigcap \mathcal{S}_k$ est au plus $c^k(n-k)!\binom{n}{k}^2$. Mais pour $k = \lceil n^{0.5+\epsilon} \rceil$, $c^k(n-k)!\binom{n}{k}^2 = o\left(n^{n^{1-2\epsilon}}\right) = o\left(n!\right)$. On peut remarquer qu'une preuve similaire est utilisée dans [EELW07] pour trouver la plus petite permutation contenant tous les motifs d'une taille donnée.

On peut aussi noter qu'ici, on a choisi $k = \lceil n^{0.5+\epsilon} \rceil$ par commodité, mais que les mêmes arguments et résultats sont valables pour $k = C \cdot \lceil n^{0.5+\epsilon} \rceil$, quelle que soit la constante C. □

On peut maintenant conclure quant à la qualité de l'approximation d'un plus grand motif commun par un motif appartenant à une classe $\mathcal{C}$:

Corollaire 5.13. *Le problème 4.1 de recherche de plus grand motif commun ne peut pas être résolu de manière approchée avec une meilleure approximation que $\sqrt{Opt}$ par le problème 5.1 de recherche de plus grand motif commun appartenant à une classe $\mathcal{C}$, en notant Opt la taille d'une solution optimale au problème 4.1.*

Démonstration. On considère le problème de recherche de plus grand motif commun aux deux permutations σ et σ, identiques. Alors le plus grand motif commun entre elles est σ. Mais leur plus grand motif commun appartenant à $\mathcal{C}$ est un plus grand motif de $\mathcal{C}$ apparaissant comme motif dans σ. D'après le théorème 5.12, il est possible qu'un tel motif soit de taille $\sqrt{|\sigma|}$ asymptotiquement. □

Troisième partie

Analyse combinatoire de modèles pour l'évolution des génomes

« Ma racine est au fond des bois. »

Émile Gallé

Introduction aux problèmes de distance dans les modèles pour l'évolution des génomes

Cette partie de ma thèse est consacrée à l'étude de deux modèles pour l'évolution des génomes, d'un point de vue essentiellement combinatoire, mais aussi algorithmique. Les deux modèles envisagés seront celui de la *duplication en tandem avec perte aléatoire*, et celui du *tri parfait par renversements*. Un chapitre est consacré à chacun d'eux, et on présente ici les aspects qui leur sont communs dans la modélisation des phénomènes biologiques.

L'objet de cette introduction n'est pas de décrire précisément les aspects biologiques de l'évolution des génomes, mais d'en donner les principales caractéristiques prises en compte dans les modèles étudiés.

Les phénomènes biologiques mis en jeu dans le processus d'évolution des génomes sont nombreux et complexes, et peuvent être envisagés à différentes échelles. Il peut se produire des mutations à l'échelle des nucléotides, ou des mutations plus globales, qui affectent l'ordre des gènes dans le génome [BP02].

Dans l'étude des processus d'évolution au niveau des nucléotides, les objets biologiques considérés sont les séquences d'ADN ou d'ARN, et les mutations sont typiquement l'insertion, la suppression et la substitution d'un nucléotide. Ces objets ont d'abord été modélisés par des mots sur un alphabet à quatre lettres, puis les modèles ont été enrichis pour prendre en compte d'autres paramètres biologiques importants (structures secondaires d'ARN par exemple). On pourra consulter [Den] pour une présentation synthétique de ces modèles.

Le niveau de détail que l'on envisage ici est plus large : on s'intéresse aux génomes en tant que successions de gènes, dans un ordre donné. À cette "large" échelle, les mutations affectent l'ordre des gènes dans le génome. Même en se restreignant à l'analyse de ces mutations plus globales, de nombreux processus sont mis en jeu : on peut citer les inversions [BMS05, Lab04], les transpositions [CL04, Lab05, Lab06], les duplications [CCMR06, BCM$^+$, BV], les renversements [DST07, FV04], ... Devant la complexité de la réalité, le plus souvent, les modèles pour l'évolution des génomes se concentrent sur un seul type de mutation possible, même si quelques travaux font exception à cette règle [EMS03, par exemple]. Les objets utilisés pour la modélisation et pour l'analyse formelle des problèmes posés dans ce contexte sont les permutations (signées ou non).

Une permutation de $[1..n]$ représente un génome de n gènes. Chaque gène reçoit un numéro unique, entre 1 et n. Un entier dans la permutation représente donc un gène, et la permutation représente l'ordre d'apparition des gènes dans le génome. On pourra toujours supposer qu'un génome donné correspond à la permutation identité $12 \dots n$, modulo ré-étiquetage. Les mutations possibles prises en compte dans le modèle d'évolution correspondent alors à des règles de transformation qui modifient l'ordre des éléments les uns par rapport aux autres dans la permutation. Les permutations qu'on peut obtenir par application de ces règles sont le résultat de "mutations" de la permutation de départ.

On peut raffiner un peu cette modélisation en utilisant plutôt des permutations *signées*. En effet, dans les génomes, les gènes ont un sens : des deux extrémités, l'une correspond au début du gène et l'autre à la fin. Pour avoir une modélisation qui prend en compte ce paramètre, on peut affecter à chaque élément de la permutation un signe, indiquant le sens dans lequel le gène apparaît dans le génome.

Dans le cadre du modèle de duplication en tandem avec perte aléatoire, la modélisation des génomes passe par des permutations non signées. En revanche, on utilisera la modélisation par des permutations signées pour l'étude du problème du tri parfait par renversements.

La problématique principale dans l'étude des réarrangements de génomes est d'estimer la distance évolutive qui sépare deux espèces. On se concentre pour cela sur les gènes qui apparaissent dans les génomes de ces deux espèces, sous la forme de gènes *homologues* c'est-à-dire ayant un unique ancêtre commun. Un gène de la première espèce et son homologue dans la deuxième espèce sont représentés par le même entier, et ces deux génomes correspondent donc à des permutations sur le même ensemble d'entiers (sur $[1..n]$ par exemple). L'ordre entre les gènes dans les deux génomes considérés n'a aucune raison *a priori* d'être le même.

Dans un modèle donné, un *scénario* est une suite de mutations autorisées par le modèle, qui modifient l'ordre des gènes à partir du premier génome pour aboutir au second. Chaque scénario a un coût qui dépend des mutations (ou *étapes*) qui le composent. La *distance* entre deux permutations est alors le coût minimal d'un scénario conduisant de l'une à l'autre. Un tel scénario réalisant la distance entre deux permutations sera appelé scénario *optimal*.

La manière la plus simple de définir le coût d'un scénario est de considérer que chaque étape est de coût unitaire. La distance entre deux génomes est alors le nombre de mutations nécessaires dans le modèle étudié pour passer de la permutation σ_1 représentant le génome de la première espèce à la permutation σ_2 qui est associée au génome de la deuxième espèce. On peut aussi envisager des modèles où les mutations peuvent avoir des coûts différents, qui s'additionnent pour définir le coût d'un scénario composé d'une succession d'étapes.

Comme expliqué plus haut, on pourra supposer sans perte de généralité qu'on a toujours $\sigma_1 = 12\dots n$, et on procédera ainsi pour l'étude du modèle de duplication en tandem avec perte aléatoire. Pour le problème du tri parfait par renversements, il est plus adapté de supposer qu'on a toujours $\sigma_2 = 12\dots n$, ce qui ne restreint pas non plus la généralité du propos, modulo un ré-étiquetage adapté. Dans le premier cas, on dit qu'on calcule des scénarios d'évolution à partir de l'identité ; dans le second cas, on parle de scénario vers l'identité.

Le terme de "distance", qui désigne le nombre de mutations nécessaires pour transformer σ_1 en σ_2, est en fait un abus de langage. En effet, cette distance est par définition *orientée*, et selon le modèle d'évolution de génome, la distance évolutive associée n'est pas toujours symétrique. Il est vrai que dans la plupart des modèles étudiés dans la littérature, les opérations de transformation des permutations sont réversibles (pour le même coût). Mais certains modèles sortent de ce cadre, et c'est en particulier le cas du modèle de duplication en tandem avec perte aléatoire.

Dans le chapitre 6, on s'intéresse au modèle de duplication en tandem avec perte aléatoire [CCMR06]. On décrit une caractérisation des permutations obtenues en au plus p étapes à partir d'une permutation identité dans ce modèle, en termes de motifs exclus. Dans le cas particulier du modèle de *duplication complète avec perte aléatoire*, on décrit complètement la base de motifs exclus qui caractérise la classe des permutations obtenues en au plus p étapes. On étudie aussi les motifs exclus qui apparaissent dans cette base, pour en donner une caractérisation locale, et obtenir des résultats d'énumération (même s'ils restent partiels). Dans le modèle général, on montre que la base de la classe des permutations obtenues en au plus p étapes est finie, et on donne une borne sur la taille des motifs qu'elle contient. On propose aussi un algorithme de calcul de scénario d'évolution dans ce modèle, et on compare le coût des scénarios ainsi calculés au coût d'un scénario optimal. Les travaux présentés au chapitre 6 ont fait l'objet des deux articles [BR09] et [BP08].

Le chapitre 7 s'intéresse à un autre modèle pour l'évolution des génomes : celui du tri par-

fait par renversements [FV04]. On y analyse la complexité d'un algorithme de calcul de scénario optimal décrit par S. Bérard, A. Bergeron, C. Chauve et C. Paul dans [BBCP07]. Il s'agit d'un algorithme de complexité paramétrée, qui est exponentiel dans le pire des cas. On démontre qu'il est en fait polynomial en moyenne. On analyse ensuite certains paramètres des scénarios optimaux calculés, pour le cadre restreint des permutations séparables (ou commutantes), particulièrement pertinent d'un point de vue biologique. On donne des estimations asymptotiques du nombre moyen de renversements dans un scénario optimal, et de la taille moyenne d'un renversement dans un tel scénario : les résultats de cette analyse sont cohérents avec les observations biologiques, ce qui vient conforter la pertinence de ce modèle. Une grande partie des résultats du chapitre 7 sont publiés dans [BCMR], dont une version étendue est en préparation.

Chapitre 6

Duplication en tandem avec perte aléatoire

Sommaire

6.1 Présentation du modèle "duplication en tandem avec perte aléatoire" 114
6.1.1 Définition du modèle et quelques résultats antérieurs 114
6.1.2 Analyse combinatoire du modèle restreint de "duplication complète avec perte aléatoire" 116
6.2 Permutations minimales avec d descentes 118
6.2.1 Caractérisation 119
6.2.2 Énumeration pour les tailles extrémales 121
6.3 Analyse combinatoire du modèle général de "duplication en tandem avec perte aléatoire" 136
6.3.1 Permutations obtenues en une étape de largeur au plus K 136
6.3.2 Permutations obtenues en p étapes de largeur au plus K 137
6.4 Algorithmes de calcul de scénarios, et étude de complexité 142
6.4.1 Étude algorithmique du modèle de duplication complète avec perte aléatoire 142
6.4.2 Borne supérieure dans le modèle général 143
6.4.3 Borne inférieure dans le modèle général 148

6.1 Présentation du modèle "duplication en tandem avec perte aléatoire" et première analyse dans un cadre restreint

6.1.1 Définition du modèle et quelques résultats antérieurs

Le modèle de *duplication en tandem avec perte aléatoire* a été introduit par K. Chaudhuri, K. Chen, R. Mihaescu et S. Rao dans [CCMR06] en 2006. Dans ce modèle, les seules opérations à travers lesquelles les génomes peuvent évoluer sont les duplications et pertes de gènes. Dans la plupart des modèles pour l'évolution des génomes, les duplications et pertes de gènes ne sont pas prises en considération : d'autres opérations sur les génomes sont envisagées, et l'ajout des duplications et pertes à ces modèles induit une complexité combinatoire souvent dissuasive. Il faut remarquer quelques exceptions dans des travaux de N. El-Mabrouk et D. Sankoff [EMS03].

Ce modèle (qu'on appellera modèle de duplication-perte par souci de concision) a suscité beaucoup d'intérêt dans la communauté biologique, où le phénomène de duplication-perte est considéré comme le type de réarrangement prépondérant dans l'évolution des génomes mitochondriaux des animaux. C'est en outre un modèle riche, qui généralise le modèle de duplication complète avec perte aléatoire, étudié en soi par le passé et dont l'intérêt est manifeste. Plus de détails sont donnés dans [CCMR06] et dans ses références.

Dans le modèle de duplication-perte, les génomes sont représentés par des permutations (non signées). Ces permutations évoluent à travers des étapes de duplication en tandem avec perte aléatoire (ou étape de duplication-perte, ou même étape lorsque le contexte n'est pas ambigu). Chaque étape de duplication-perte est composée de deux opérations élémentaires indissociables : (*i*) la duplication en tandem d'un fragment continu du génome, c'est-à-dire que la copie du fragment dupliqué est insérée dans le génome juste après le fragment original, et (*ii*) la perte d'une des deux copies de chaque gène dupliqué. Comme dans [CCMR06], on suppose que cette perte se produit immédiatement après la duplication des gènes, ce qui est une assez bonne approximation de la réalité, à l'échelle de l'évolution. On appelle *largeur* d'une étape le nombre de gènes dupliqués. Un exemple est donné en figure 6.1.

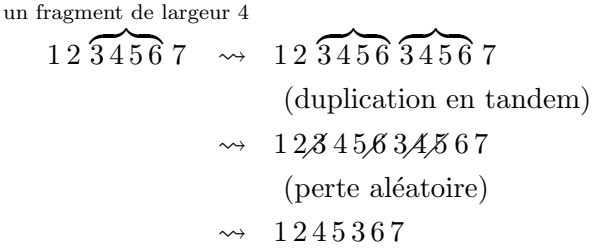


FIG. 6.1 – Exemple d'une étape de duplication en tandem avec perte aléatoire, de largeur 4.

Pour mesurer la "distance" entre deux permutations dans ce modèle, on définit une notion de *coût* associé à une étape. Dans [CCMR06], les auteurs définissent le coût d'une étape de largeur k par α^k, où $\alpha \geq 1$ est un paramètre. L'intuition qui conduit à cette fonction de coût géométrique est un modèle où tous les points de la permutation se trouvant après le début de la zone de duplication ont la même probabilité $(1 - \frac{1}{\alpha})$ d'être la fin du fragment dupliqué. Le coût d'une étape est alors inversement proportionnel à sa probabilité. Les auteurs de [CCMR06] suggèrent que d'autres fonctions de coût naturelles pourraient être étudiées, en particulier des fonctions affines de la largeur. Ici, on envisage un modèle avec une fonction de coût constante par morceaux, très simple : le coût d'une étape de largeur k est 1 si $k \leq K$ et est infini si $k > K$. K est un paramètre dont la valeur est dans $\mathbb{N} \cup \{\infty\}$. On supposera bien sûr que $K \geq 2$, ou le modèle serait vide de sens.

On envisagera aussi (dans la section 6.4 seulement) la possibilité que $K = K(n)$ soit une fonction de la taille n de la permutation sur laquelle sont appliquées les opérations de duplication-perte.

Un cas particulier du modèle de duplication en tandem avec perte aléatoire va nous intéresser : il s'agit du modèle de duplication complète avec perte aléatoire. Dans ce modèle, la duplication ne concerne pas un fragment arbitraire de la permutation, mais la permutation toute entière à chaque étape. De manière équivalente, on peut le décrire comme étant le modèle de duplication-perte avec une fonction de coût constante égale à 1. En effet, une étape de duplication-perte portant sur un fragment de la permutation peut toujours être simulée par une étape portant sur la permutation toute entière, et dans ce cas, le coût est 1 dans les deux cas. On remarque que ce modèle peut être vu comme un cas particulier du modèle décrit dans [CCMR06] (pour la valeur du paramètre $\alpha = 1$) ou de notre modèle (avec $K = \infty$ ou $K = K(n) = n$). En un certain sens, ces deux modèles sont des généralisations du modèle de duplication complète avec perte aléatoire.

Parallèlement aux motivations biologiques qui ont conduit à étudier le modèle de duplication-perte, on peut remarquer que ce modèle d'évolution de permutations entre aussi dans le cadre des *permuting machines* défini dans [AAA+07]. Une telle machine prend en entrée une permutation, et la transforme en une permutation de sortie, la transformation devant satisfaire deux propriétés : d'abord l'indépendance par rapport aux valeurs, et ensuite le respect de la relation de motif. Les détails sont donnés dans [AAA+07]. On vérifie facilement que la transformation de duplication-perte satisfait ces deux propriétés. On peut donc voir une étape de duplication-perte (dans l'un des modèles définis plus haut) comme une exécution d'une *permuting machine* adaptée. Quand on va considérer des permutations obtenues après une séquence d'étapes de duplication-perte, cela correspondra à des permutations obtenues en sortie d'une combinaison en série de plusieurs *permuting machines* identiques.

La problématique générale qui nous intéresse dans l'étude du modèle de duplication-perte est celle de l'estimation de la "distance" entre deux permutations. Ce modèle a la particularité que les opérations de duplication-perte ne sont pas réversibles à même coût : on passe par exemple de 1 2 3 4 à 2 4 1 3 en une étape, mais il en faudra au moins 2 pour revenir en arrière. Les "distances" que l'on va considérer sont donc orientées : la distance d'une permutation σ_1 à une permutation σ_2 est le coût minimal d'une séquence d'étapes (ou scénario) de duplication-perte permettant de transformer σ_1 en σ_2. Le coût d'un scénario est défini par la somme des coûts des étapes de duplication-perte qui le compose. On supposera toujours que σ_1 est l'identité $12\ldots n$, ce qui ne restreint pas la généralité du propos, modulo réétiquetage.

$$1\,2\,3\,4\,5\,6\,7 \rightsquigarrow 3\,4\,7\,1\,2\,5\,6 \rightsquigarrow 3\,4\,1\,5\,7\,2\,6$$

FIG. 6.2 – Un exemple de scénario de duplication-perte pour $\sigma = 3\,4\,1\,5\,7\,2\,6$, en partant de l'identité $1\,2\,3\,4\,5\,6\,7$. Ce scénario est composé de deux étapes. Dans le modèle de duplication complète avec perte aléatoire, il est de coût 2, ce qui est le coût minimal ici.

Dans le contexte de l'étude des classes de permutations, une première remarque à faire est que la propriété d'une permutation de pouvoir être obtenue en au plus p étapes de duplication-perte est une propriété stable pour la relation de motif $\preceq$: si on peut obtenir σ en au plus p étapes, et si $\pi \preceq \sigma$, alors on peut aussi obtenir π en au plus p étapes. Pour s'en convaincre, il suffit de voir qu'un scénario de duplication-perte pour σ, s'il est restreint aux éléments de σ formant une occurrence de π, donne un scénario de duplication-perte pour π. Par définition (voir page 16), cela signifie que l'ensemble des permutations que l'on peut obtenir en au plus p étapes forme une classe de permutations, dont la base de motifs exclus est constituée des permutations minimales (au sens de la relation $\preceq$) qui ne peuvent pas être obtenues en p étapes de duplication-perte.

La remarque précédente peut aussi être vue comme instance d'un résultat plus général démontré dans [AAA+07] : les permutations obtenues en sortie d'une combinaison en série de *permuting machines* forment une classe de permutations.

Le problème du calcul de distance a été complètement résolu dans [CCMR06] pour le modèle de duplication complète avec perte aléatoire. Les auteurs ont décrit un algorithme qui, étant donnée une permutation σ, calcule un scénario de duplication-perte de l'identité à σ. Ils ont aussi montré que le nombre d'étapes du scénario calculé est minimal. On poursuivra cette étude en donnant une caractérisation en termes de motifs exclus de la classe des permutations qui peuvent être obtenues avec un coût au plus p à partir de l'identité. Dans ce modèle de duplication complète avec perte aléatoire, on mène une étude combinatoire approfondie des motifs exclus qui définissent cette classe.

Dans le modèle général de duplication-perte, on analysera aussi les permutations obtenues après un nombre p d'étapes. On introduit pour cela la notation suivante :

Définition 6.1. La classe $\mathcal{C}(K, p)$ décrit la classe de toutes les permutations obtenues à partir de $12\dots n$ (pour n'importe quelle valeur de n) après p étapes de duplication-perte de largeur au plus K, pour des valeurs fixées des paramètres p et K.

On ne considère pas le cas $K = K(n)$ dans ce contexte. Enfin, on rappelle que les étapes de duplication-perte ne sont pas réversibles, et que $\mathcal{C}(K, p)$ n'est donc pas la classe des permutations que l'on peut trier (c'est-à-dire transformer en $12\dots n$) en p étapes de duplication-perte de largeur au plus K.

Un premier travail consistera à caractériser les permutations de $\mathcal{C}(K, p)$ par une base de motifs exclus. Pour le cas $p = 1$, on décrira complètement la base de la classe $\mathcal{C}(K, 1)$. En revanche, pour le cas général, on obtient seulement une borne supérieure sur la taille des motifs exclus qui caractérisent $\mathcal{C}(K, p)$. Cela permet en particulier de conclure que $\mathcal{C}(K, p)$ a une base finie.

Un autre point de vue, plus algorithmique, consiste à examiner combien d'étapes d'une largeur donnée sont nécessaires dans un scénario de duplication-perte de $12\dots n$ à une permutation σ de taille n. Plus précisément, en fixant une taille n et une valeur du paramètre K –constante ou fonction de n–, on voudrait estimer le nombre p tel que toute permutation de taille n puisse être obtenue en au plus p étapes de duplication-perte de largeur au plus K, en partant de $12\dots n$. On obtient des bornes supérieures et inférieures, dans le pire des cas et en moyenne, sur le nombre d'étapes d'un scénario de duplication-perte pour $\sigma \in \mathcal{S}_n$. Nous verrons que ces bornes supérieures et inférieures coïncident dans la plupart des cas.

6.1.2 Analyse combinatoire du modèle restreint de "duplication complète avec perte aléatoire"

Avant de s'intéresser aux classes $\mathcal{C}(K, 1)$ et $\mathcal{C}(K, p)$, on se concentre d'abord sur le modèle plus simple de duplication complète avec perte aléatoire. Il correspond à $K = \infty$ dans notre modèle, mais il a déjà été considéré dans [CCMR06]. On ne démontre pas de nouveaux résultats dans cette partie, mais on donne une interprétation combinatoire de résultats existants, avec le point de vue des classes de permutations à motifs exclus.

Dans le modèle de duplication complète avec perte aléatoire, chaque étape de duplication-perte est de coût 1, quelle que soit sa largeur. Ainsi, sans perte de généralité, on peut supposer qu'à chaque étape, c'est tout le génome qui est dupliqué. Le coût d'une permutation $\sigma \in \mathcal{S}_n$ est donc défini comme le nombre minimal d'étapes d'un scénario de duplication-perte qui transforme $12\dots n$ en σ.

Les permutations de coût au plus p dans le modèle de duplication complète avec perte aléatoire ont été caractérisées de manière implicite dans l'article [CCMR06], par le théorème 6.2 :

Théorème 6.2. *Soit σ une permutation de taille n. Dans le modèle de duplication complète avec perte aléatoire, $\lceil \log_2(\text{nombre de sous-séquences croissantes maximales de } \sigma) \rceil$ étapes de duplication-perte sont nécessaires et suffisantes pour obtenir σ à partir de $12\dots n$.*

Une *sous-séquence croissante* de σ est simplement une séquence d'éléments de σ à des positions *consécutives* (à la différence de la notion de motif) et dont les valeurs sont ordonnées en ordre croissant. Une sous-séquence croissante est maximale si elle ne peut être prolongée ni par la gauche ni par la droite.

Exemple 6.3. Par exemple, $\sigma = 5\,2\,4\,3\,1\,6$ a 4 sous-séquences croissantes maximales, qui sont 5, 2 4, 3 et 1 6.

On peut reformuler le théorème 6.2 pour obtenir plutôt le théorème 6.8, qui utilise, à la place des sous-séquences croissantes maximales, un paramètre très classique en combinatoire des permutations : le nombre de descentes. On en rappelle la définition et quelques propriétés (voir [Bón04a, Sta97] par exemple).

Définition 6.4. Soit σ une permutation de taille n. On dit qu'il y a une *descente* (resp. une *montée*) à la position i dans σ, pour $1 \leq i \leq n-1$, si $\sigma_i > \sigma_{i+1}$ (resp. $\sigma_i < \sigma_{i+1}$). On note $desc(\sigma)$ (resp. $asc(\sigma)$) le nombre de descentes (resp. de montées) d'une permutation σ.

Les notations $desc(\sigma)$ et $asc(\sigma)$ font référence aux termes anglais *descents* et *ascents*.

Exemple 6.5. Par exemple, $\sigma = 5\,2\,4\,3\,1\,6$ a 3 descentes, qui se trouvent aux positions 1, 3 et 4.

D'après la définition 6.4, on voit immédiatement que pour tout $\sigma \in \mathcal{S}_n$, $desc(\sigma)+asc(\sigma) = n-1$. Une permutation σ de taille n a au plus $n-1$ descentes, le cas de $n-1$ descentes correspondant uniquement à la permutation miroir de l'identité $n(n-1)\ldots21$.

La remarque suivante nous sera utile dans la section 6.4.2 :

Remarque 6.6. Le nombre moyen de descentes parmi les permutations de taille n est $\frac{n-1}{2}$.

En effet, en considérant le miroir σ^r d'une permutation σ, on a $asc(\sigma) = desc(\sigma^r)$. Le résultat annoncé est alors facilement obtenu par le calcul suivant :

$$2 \times \frac{1}{n!} \sum_{\sigma \in \mathcal{S}_n} desc(\sigma) = \frac{1}{n!} \sum_{\sigma \in \mathcal{S}_n} desc(\sigma) + desc(\sigma^r) = \frac{1}{n!} \sum_{\sigma \in \mathcal{S}_n} (n-1) = n-1.$$

Il est aussi immédiat de remarquer que :

Remarque 6.7. Le nombre de sous-séquences croissantes maximales de σ est $desc(\sigma)+1$.

Plus précisément, les positions des descentes et n indiquent les positions des derniers éléments de chacune des sous-séquences croissantes maximales de σ.

Le théorème 6.2 se reformule alors de la manière suivante :

Théorème 6.8. *Soit σ une permutation de taille n. Dans le modèle duplication complète avec perte aléatoire, $\lceil \log_2(desc(\sigma)+1) \rceil$ étapes de duplication-perte sont nécessaires et suffisantes pour obtenir σ à partir de $12\ldots n$.*

De manière équivalente, on peut dire aussi que les permutations que l'on peut obtenir en au plus p étapes dans le modèle de duplication complète avec perte aléatoire sont exactement celles dont le nombre de descentes est au plus $2^p - 1$.

Comme dans le modèle général de duplication-perte, les permutations obtenues en au plus p étapes dans le modèle de duplication complète avec perte aléatoire forment une classe de permutations. Connaissant un scénario en au plus p étapes pour une permutation σ, un scénario pour chaque motif π de σ est directement hérité de celui de σ en ne tenant compte que des éléments de σ qui forment une occurrence de π. Les motifs exclus qui constituent la base de la classe des permutations obtenues en au plus p étapes dans ce modèle sont les permutations minimales (au

sens de la relation $\preceq$) qui ne peuvent pas être obtenues en p étapes de duplication complète avec perte aléatoire. D'après ce qui précède, ces motifs exclus sont donc les permutations minimales qui ont (au moins) 2^p descentes. Par minimalité, ces permutations ont en fait *exactement* 2^p descentes.

On étudie ces permutations ayant $d = 2^p$ descentes plus en détail dans la section 6.2. On peut d'ores et déjà remarquer (mais on y reviendra dans le corollaire 6.14) qu'une permutation à d descentes et minimale pour ce critère est de taille au plus $2d$: par minimalité, elle ne contient jamais deux montées consécutives. Cela implique en particulier que les motifs exclus qui caractérisent la classe des permutations que l'on peut obtenir en au plus p étapes de duplication-perte forment une base finie.

Ainsi, on peut énoncer le théorème 6.9 :

Théorème 6.9. *Les permutations que l'on peut obtenir en au plus p étapes dans le modèle de duplication complète avec perte aléatoire forment une classe de permutations. Cette classe est caractérisée par une base finie de motifs exclus : les permutations ayant exactement d descentes et qui sont minimales (au sens de la relation d'ordre $\preceq$) pour ce critère.*

Dans la section 6.2, on propose une étude combinatoire des motifs de cette base. On reviendra ensuite dans les sections 6.3 et 6.4 à l'analyse du modèle plus général de duplication en tandem avec perte aléatoire.

6.2 Permutations minimales avec d descentes

On a vu dans la Section 6.1 que, dans le modèle duplication complète avec perte aléatoire, les permutations obtenues en au plus p étapes de duplication-perte sont celles ayant au plus $2^p - 1$ descentes. Elles forment une classe de permutations, dont la base de motifs exclus est constituée des permutations minimales ayant 2^p descentes. Plus généralement, on s'intéresse dans cette section à l'étude de la classe des permutations ayant au plus $d-1$ descentes, sans que d soit nécessairement une puissance de 2. Comme dans le théorème 6.9, cette classe peut être décrite par sa base de motifs exclus $\mathcal{B}_d$ défini comme suit :

Définition 6.10. L'ensemble $\mathcal{B}_d$ est formé des permutations ayant d descentes et qui sont minimales (au sens de la relation de motif) pour ce critère.

D'après cette définition, les permutations de $\mathcal{B}_d$ ont *au moins* d descentes. Il est cependant aisé de voir que, par minimalité, leur nombre de descentes est en réalité *exactement* d.

Notre travail porte essentiellement sur l'ensemble $\mathcal{B}_d$ en lui-même, mais notre motivation est l'étude de la classe $\mathcal{S}(\mathcal{B}_d)$ des permutations ayant un nombre de descentes strictement inférieur à d. Un premier résultat concernant cette classe apparaît comme un cas particulier d'un théorème de M. D. Atkinson, M. Murphy et N. Ruškuc (théorème 2.9 dans [AMR02]) :

Corollaire 6.11. *La classe des permutations qui ont au plus $d-1$ descentes a une base finie.*

Démonstration. Les permutations ayant au plus $d-1$ descentes peuvent être partitionnées en d séquences croissantes, séparées les unes des autres soit par des montées, soit par des descentes. Cette décomposition n'est pas nécessairement unique.

Dans [AMR02], les auteurs notent $W(e_1, \ldots, e_k)$, avec pour tout i, $e_i \in \{+, -\}$, l'ensemble des permutations qui admettent une décomposition (de gauche à droite) comme concaténation de k séquences d'entiers, la i-ème séquence étant croissante ou décroissante selon que e_i vaut $+$ ou $-$ respectivement. Ces séquences peuvent être indifféremment séparées par des montées ou des descentes.

Avec ces notations, il est facile de voir que l'ensemble des permutations ayant au plus $d-1$ descentes est $W(f_1, \ldots, f_d)$ avec $f_i = +$ pour tout i. Le théorème 2.9 de [AMR02] énonçant que tout $W(e_1, \ldots, e_k)$ a une base finie donne le résultat. □

Pour le cas considéré, la base finie annoncée par le corollaire 6.11 est $\mathcal{B}_d$. Le travail présenté ici peut être vu comme un raffinement du résultat de [AMR02], dans le cas très particulier $W(+,\dots,+)$.

Les résultats présentés dans cette section sont le fruit d'une collaboration avec Elisa Pergola [BP08].

On donne d'abord une caractérisation locale des permutations de $\mathcal{B}_d$. En effet, la définition des permutations de $\mathcal{B}_d$, comme étant les minimales (pour la relation de motif) avec d descentes, est assez peu utilisable telle quelle. Dans la section 6.2.1, on prouve que les permutations de $\mathcal{B}_d$ sont celles dont les montées satisfont une propriété simple et locale : il y a une montée dans $\sigma \in \mathcal{S}_n \cap \mathcal{B}_d$ à l'indice i si et seulement si $2 \leq i \leq n-2$ et $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ est une occurrence soit du motif 2 1 4 3, soit du motif 3 1 4 2.

En utilisant cette caractérisation, on s'intéresse ensuite à l'énumération des permutations de $\mathcal{B}_d$. Cette tâche difficile n'est pas menée à son terme, mais on obtient des résultats partiels, en fixant la taille des permutations de $\mathcal{B}_d$ sur des valeurs particulières. Dans la section 6.2.2, on verra que les permutations de $\mathcal{B}_d$ sont de taille comprise entre $d+1$ et $2d$, et on donnera l'énumération des permutations de $\mathcal{B}_d$ de taille $d+1$, $d+2$ et $2d$. Pour ce faire, on introduira une représentation des permutations minimales à d descentes sous forme de *posets* (ou ensembles partiellement ordonnés).

6.2.1 Caractérisation

Le but est ici de proposer des conditions nécessaires et suffisantes d'appartenance d'une permutation à $\mathcal{B}_d$. On commence par une condition nécessaire très simple :

Proposition 6.12. *Soit $\sigma \in \mathcal{B}_d$ une permutation minimale à d descentes. Toute montée dans σ est alors précédée et suivie d'une descente.*

Démonstration. Il est facile de voir que σ ne peut ni commencer ni finir par une montée : sinon, en enlevant le premier (respectivement, le dernier) élément de σ, on obtiendrait une permutation avec le même nombre d de descentes, contredisant la minimalité de σ pour ce critère.

De la même façon, si σ contient deux montées consécutives, $\sigma_{i-1}\sigma_i$ et $\sigma_i\sigma_{i+1}$, effacer l'élément σ_i dans σ ne change pas le nombre de descentes, et apporte la même contradiction. □

On peut donc décomposer une permutation de $\mathcal{B}_d$ en séquences non vides de descentes, séparées par des montées isolées. Cette décomposition est illustrée sur la figure 6.3.

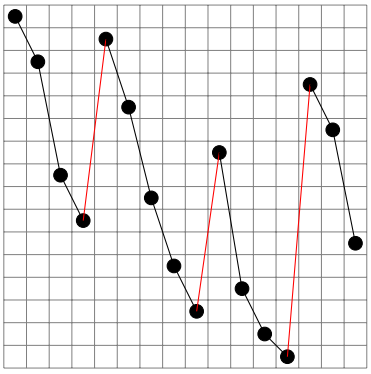


FIG. 6.3 – Décomposition d'une permutation minimale à d descentes en séquences non vides de descentes séparées par des montées isolées.

On peut en fait pousser un peu plus loin cette décomposition en séquences non vides de descentes séparées par des montées isolées, pour en extraire la caractérisation suivante des permutations de $\mathcal{B}_d$:

Théorème 6.13. *Une permutation $\sigma \in \mathcal{S}_n$ est une permutation minimale à d descentes si et seulement si elle possède exactement d descentes et chacune de ses montées $\sigma_i\sigma_{i+1}$ est telle que $2 \leq i \leq n-2$ et $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ forme une occurrence soit du motif 2 1 4 3, soit du motif 3 1 4 2.*

Démonstration. Soit σ une permutation minimale à d descentes. D'après la propriété 6.12, toute montée $\sigma_i\sigma_{i+1}$ est telle que $2 \leq i \leq n-2$, et $\sigma_{i-1}\sigma_i$ et $\sigma_{i+1}\sigma_{i+2}$ sont des descentes. Cela nous assure que $\sigma_{i-1} > \sigma_i$, $\sigma_i < \sigma_{i+1}$ et $\sigma_{i+1} > \sigma_{i+2}$.

Supposons à présent que $\sigma_{i-1} > \sigma_{i+1}$: dans ce cas, la permutation obtenue en enlevant l'élément σ_i dans σ a le même nombre de descentes que σ (et une montée de moins). Ceci contredit la minimalité de σ, et on déduit que $\sigma_{i-1} < \sigma_{i+1}$. De la même façon, si $\sigma_i > \sigma_{i+2}$, on ne change pas le nombre de descentes en enlevant l'élément σ_{i+1}, et donc $\sigma_i < \sigma_{i+2}$.

La permutation σ satisfait donc les cinq inégalités suivantes autour de sa montée $\sigma_i\sigma_{i+1}$:

$$\sigma_{i-1} > \sigma_i, \qquad \sigma_i < \sigma_{i+1}, \qquad \sigma_{i+1} > \sigma_{i+2}, \qquad \sigma_{i-1} < \sigma_{i+1}, \qquad \text{and} \qquad \sigma_i < \sigma_{i+2}.$$

Il est aisé de déduire de ces inégalités que les éléments $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ forment soit un motif 2 1 4 3, soit un motif 3 1 4 2.

Les différentes configurations (autorisées ou interdites) autour de la montée $\sigma_i\sigma_{i+1}$ dans σ sont résumées dans la figure 6.4 □

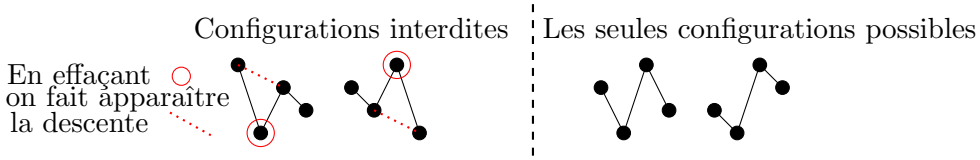


FIG. 6.4 – Les éléments $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ autour d'une montée $\sigma_i\sigma_{i+1}$ dans une permutation σ minimale avec d descentes.

Cette caractérisation promettait d'être un outil pour l'énumération des permutations minimales à d descentes, mais elle n'a tenu ses promesses que partiellement. Nous n'avons obtenu pour l'instant que des résultats partiels d'énumération des permutations de $\mathcal{B}_d$, à taille n fixée (en fonction de d). Le cas $n = d+1$ est immédiat. On examine les cas $n = d+2$ et $n = 2d$ dans la section suivante, en utilisant une représentation des permutations par des *posets* (ou ensembles partiellement ordonnés) qui provient directement de la caractérisation des permutations minimales à d descentes donnée au théorème 6.13.

Représentation des permutations minimales à d descentes par des posets

On considère un sous-ensemble des permutations de taille n, qui sont minimales à d descentes, et qui ont leurs montées et leurs descentes aux mêmes positions. En d'autres termes, à chaque position donnée i, soit toutes les permutations de l'ensemble ont une montée, soit toutes ont une descente. D'après le théorème 6.13, autour des montées, les éléments sont localement ordonnés de la même façon dans toutes ces permutations. On peut représenter ce sous-ensemble de permutations (supposé maximal pour l'inclusion) par un seul objet : un ensemble partiellement ordonné (appelé poset dans la suite) qui indique les conditions nécessaires sur l'ordre relatif des éléments de la permutations les uns par rapport aux autres.

Chaque descente est représentée par un lien du premier élément (le plus grand) vers le second (le plus petit). Pour chaque montée $\sigma_i\sigma_{i+1}$, les éléments $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ forment une structure en *diamant* avec σ_{i+1} en haut, σ_i en bas, σ_{i-1} à gauche et σ_{i+2} à droite. Un exemple est donné en figure 6.5. D'après le théorème 6.13, tout étiquetage des éléments du poset qui en respecte les contraintes d'ordre est une permutation minimale à d descentes.

On dira qu'une permutation σ satisfait la *propriété du diamant* si chacune de ses montées $\sigma_i\sigma_{i+1}$ est telle que $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ forme un diamant, c'est-à-dire est une occurrence soit du motif 2 1 4 3 soit du motif 3 1 4 2.

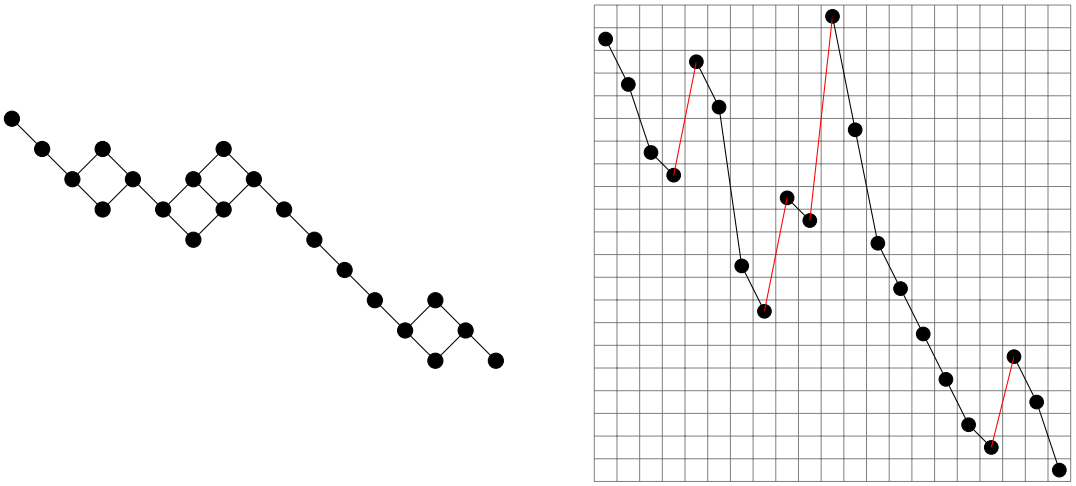


FIG. 6.5 – Un poset représentant un ensemble de permutations minimales à 16 descentes et 4 montées (par conséquent de taille 21). Par exemple, la permutation 20 18 15 14 19 17 10 8 13 12 21 16 11 9 7 5 3 2 6 4 1 appartient à cet ensemble. On donne aussi sa représentation graphique.

6.2.2 Énumeration pour les tailles extrémales

Il nous faut ici commencer par décrire les tailles possibles des permutations de $\mathcal{B}_d$, afin que la notion de "taille extrémale" ait un sens.

Une conséquence évidente de la proposition 6.12 est le corollaire suivant :

Corollaire 6.14. *Toute permutation minimale à d descentes a sa taille comprise entre $d+1$ et $2d$.*

Démonstration. Soit $\sigma \in \mathcal{B}_d$, et soit n la taille de σ. Comme on l'a vu plus haut, par minimalité, σ a exactement d descentes. On en déduit immédiatement que σ possède au moins $d+1$ éléments, *i.e.* $n \geq d+1$. En outre, la seule permutation de taille $d+1$ ayant d descentes est $(d+1)d\ldots321$, qui est minimale.

D'après la proposition 6.12, σ se décompose en séquences non vides de descentes séparées par des montées isolées. Une permutation de $\mathcal{B}_d$ est donc de taille maximale lorsqu'elle possède d descentes isolées, séparées par $d-1$ montées isolées. Une telle permutation est par conséquent de taille $2d$. On obtient ainsi que $n \leq 2d$. □

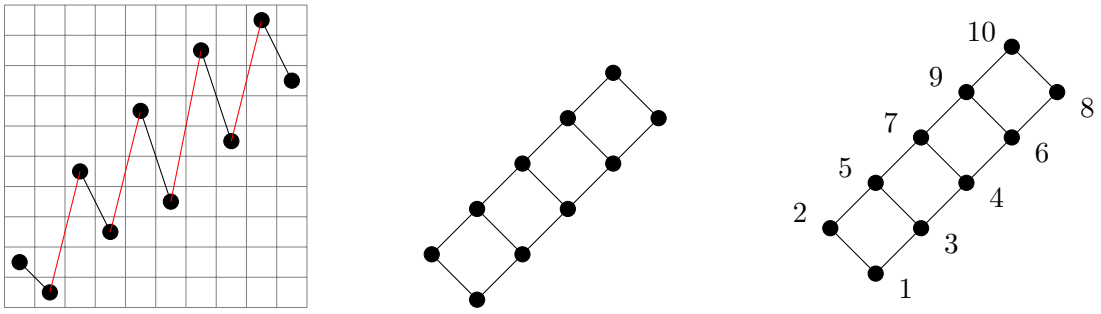
En outre, il apparaît clairement dans cette démonstration qu'il y a une unique permutation de taille $d+1$ dans $\mathcal{B}_d$: c'est le miroir de la permutation identité de taille $d+1$, *i.e.* la permutation $(d+1)d\ldots321$.

On s'intéresse maintenant à l'énumération des permutations de $\mathcal{B}_d$ qui sont de taille n extrémale (mais non triviale), c'est-à-dire pour $n = 2d$ et $n = d+2$.

6.2.2.1 Taille $2d$

Les permutations minimales à d descentes et de taille $2d$ ont, grâce à leur minimalité, une forme très spécifique. En effet, comme toutes les permutations minimales à d descentes, elles ne contiennent

jamais deux montées consécutives. Mais elles ne contiennent pas de descentes consécutives non plus, ou alors leur taille ne pourrait pas atteindre $2d$ (voir la preuve du corollaire 6.14). Ainsi, elles sont toutes composées d'une alternance entre descentes isolées et montées isolées, en commençant et en finissant par une descente. Voir la figure 6.6(a) pour un exemple.



(a) (b) (c)

FIG. 6.6 – (a) La permutation $\sigma = 2\,1\,5\,3\,7\,4\,9\,6\,10\,8$ qui est minimale avec $d = 5$ descentes et qui est de taille $2d = 10$, (b) le poset qui représente l'ensemble des permutations minimales à $d = 5$ descentes et de taille $2d = 10$ et (c) l'étiquetage de ce poset associé à σ.

Par conséquent, toutes les permutations minimales à d descentes de taille $2d$ ont leurs montées et leurs descentes aux mêmes positions, de telle sorte que l'ensemble de ces permutations est représenté par un unique poset. Ce poset a une forme d'*échelle* : c'est une séquence de $d-1$ diamants, deux diamants consécutifs possédant une arête en commun. Ces diamants correspondent aux montées dans les permutations, qui sont séparées par seulement une descente dans ce cas particulier. La figure 6.6(b) donne un exemple.

La représentation des permutations minimales à d descentes par des posets décrite page 120 justifie la proposition suivante :

Proposition 6.15. *Les permutations minimales à d descentes et de taille $2d$ sont en bijection avec les étiquetages du poset en échelle à d marches, par les entiers $\{1, 2, \ldots, 2d\}$, et qui respectent les contraintes d'ordre de ce poset.*

Cette correspondance bijective est illustrée en figure 6.6(c).

Avec cette représentation des permutations minimales à d descentes et de taille $2d$ sous forme d'étiquetages de posets, certaines propriétés de ces permutations apparaissent d'elles-même. Par exemple, dans une telle permutation, l'élément 1 est toujours en deuxième position, et l'élément $2d$ est toujours l'avant-dernier.

Le résultat d'énumération obtenu pour les permutations minimales à d descentes et de taille $2d$ est le suivant :

Théorème 6.16. *Les permutations minimales à d descentes et de taille $2d$ sont énumérées par les nombres de Catalan $C_d = \frac{1}{d+1}\binom{2d}{d}$.*

On propose deux preuves de ce résultat : une preuve analytique et une preuve bijective.

Démonstration du théorème 6.16 par une méthode analytique La première démonstration du théorème 6.16 que l'on donne ici utilise la méthode ECO, décrite en détails dans [BDLPP99]. Pour les objets qui nous intéressent, la méthode ECO va consister à décrire un processus de construction de tous les étiquetages du poset en échelle à d marches à partir de tous les étiquetages du poset en échelle à $d-1$ marches, sans construire deux fois le même étiquetage.

Dans sa présentation habituelle, la méthode ECO propose de construire des objets de taille d à partir d'objets de taille $d-1$, par un processus d'*expansion locale*, c'est-à-dire par le seul ajout d'un bloc élémentaire d'objet. Dans notre cas, pour obtenir un étiquetage de taille d à partir d'un étiquetage de taille $d-1$, on pourra être amené à modifier un grand nombre d'étiquettes, mais en conservant toujours l'ordre relatif de ces étiquettes entre elles. Dans ce sens, on peut considérer que le processus d'expansion reste bien local.

Une fois le processus d'expansion locale décrit, la méthode ECO attribue des *labels* (ou *étiquettes*) aux objets combinatoires considérés (les étiquetages du poset en échelle à d marches dans notre cas). Le label d'un objet est le nombre de ses *enfants*, c'est-à-dire le nombre d'objets qu'il produit par le processus d'expansion locale. Ces enfants peuvent à leur tour recevoir un label par le même procédé. L'arbre (ou la forêt) infini(e) dans lequel chaque objet est le père de ses enfants est appelé *arbre de génération* associé à la classe combinatoire.

Les labels des objets étant définis, on peut dériver du processus d'expansion locale une *règle de réécriture* sur ces labels, appelée aussi *règle de succession*. Elle décrit, en fonction du label d'un objet, les labels des enfants qu'il produit. Elle donne aussi un (ou des) point(s) de départ, correspondant à l'objet (ou aux objets) de taille minimale. La règle de succession suivante est connue pour décrire des classes énumérées par les nombres de Catalan (par exemple les chemins de Dyck, voir [BDLPP99] :

$$\begin{cases} (2) \\ (k) \rightsquigarrow (2)(3)\cdots(k)(k+1) \end{cases}$$

Une manière de démontrer que les étiquetages des posets en échelle à d marches sont énumérés par les nombres de Catalan est donc de décrire une construction ECO pour cette classe d'objets combinatoires dont la règle de succession est celle donnée ci-dessus.

Pour obtenir cette règle de succession, nous allons voir que les labels qui conviennent pour les étiquetages de posets en échelle à d marches sont les valeurs $(2d - \sigma_{2d} + 1)$, σ_{2d} étant l'étiquette de l'élément le plus à droite dans le poset. On peut aussi remarquer que $2d$ est l'étiquette du deuxième élément en partant de la droite (et que c'est l'étiquette maximale).

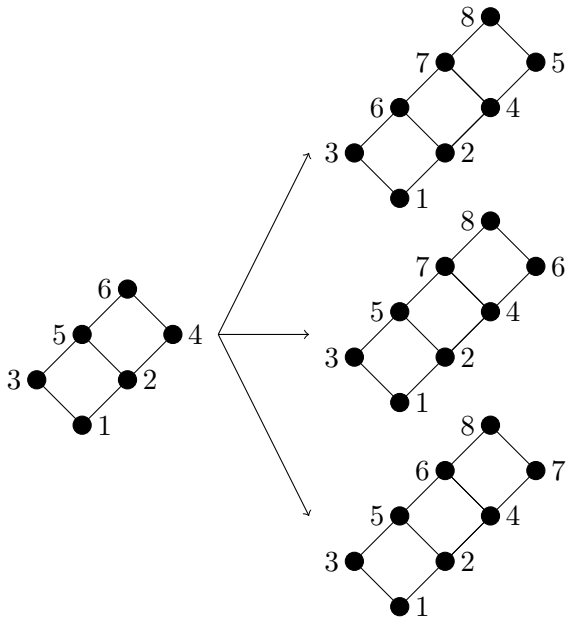


FIG. 6.7 – Un étiquetage du poset en échelle à $d = 3$ marches, et ses enfants dans la construction ECO.

On considère un étiquetage (noté σ)[e] du poset en échelle à d marches, dont le label est (k). On définit le processus d'expansion locale de la façon suivante : les enfants de σ sont les étiquetages du poset en échelle à $d+1$ marches obtenus en ajoutant une nouvelle marche sur la droite, qui est étiquetée $2d+2$ pour l'élément le plus haut, et i pour l'élément le plus à droite, pour tout i tel que $\sigma_{2d}+1 \leq i \leq 2d+1$. Les étiquettes j de σ telles que $j \geq i$ sont alors remplacées par $j+1$, de sorte à conserver d'une part l'ordre relatif entre les éléments de σ, et d'autre part la propriété que les étiquetages des posets en échelle à $d+1$ marches utilisent tous les entiers de $[1..2d+2]$ exactement une fois. Un exemple est donné en figure 6.7.

Il est facile de voir que les étiquetages obtenus respectent les conditions d'ordre du poset, et que tous les étiquetages peuvent être obtenus par ce procédé. En outre, le nombre k d'enfants de σ dans ce processus d'expansion locale est $2d+1-\sigma_{2d}$, ce qui justifie que les labels pour la méthode ECO sont les quantités $(2d-\sigma_{2d}+1)$ pour tous les étiquetages σ du poset en échelle à d marches.

On examine maintenant les labels des enfants d'un étiquetage σ du poset en échelle à d marches dont le label est (k). Ces enfants sont donc des étiquetages du poset en échelle à $d+1$ marches. Il y en a évidemment k, et d'après la formule ci-dessus, leurs labels sont $(2(d+1)-i+1)$ pour i entre $\sigma_{2d}+1 = 2d+2-k$ et $2d+1$, c'est-à-dire que les enfants d'un étiquetage de label (k) ont pour labels $(2), (3), \ldots (k), (k+1)$.

Le point de départ pour cette construction ECO est le poset en échelle à une seule marche, muni de son unique étiquetage possible (21), et dont le label est (2).

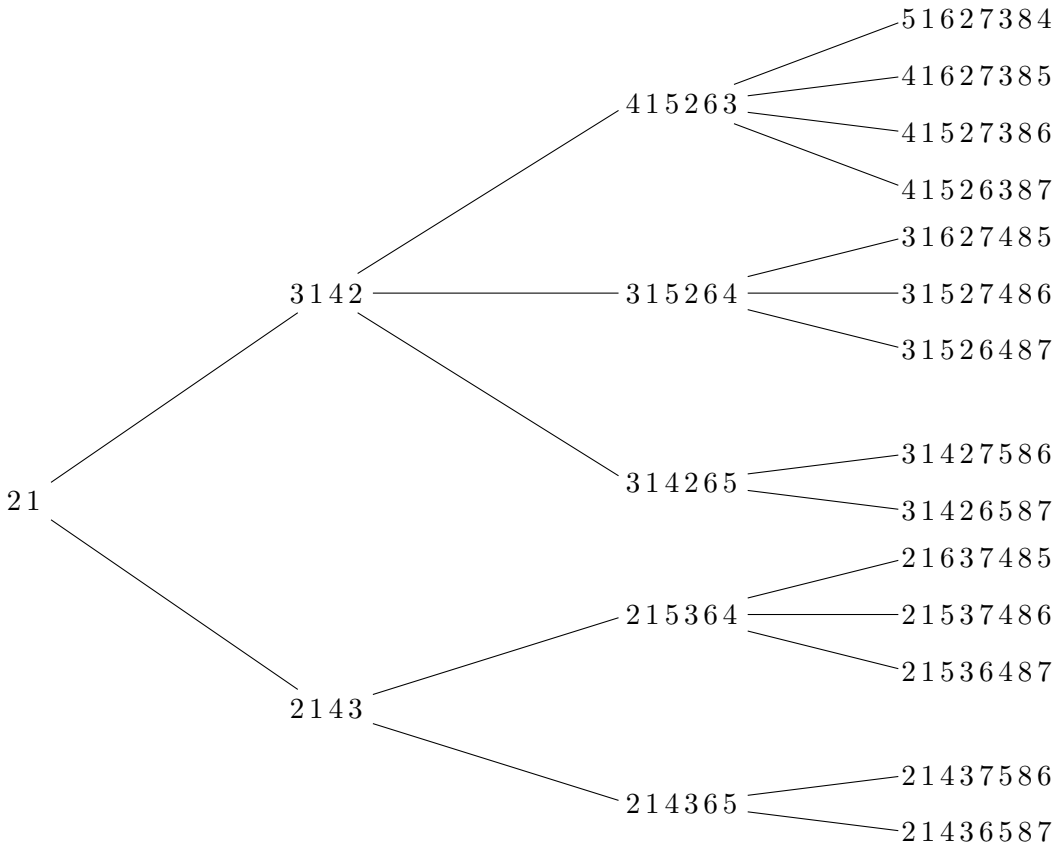


Fig. 6.8 – Les quatre premiers niveaux de l'arbre de génération associé à la construction ECO pour les étiquetages des posets en échelle décrite ici.

On obtient donc la règle de succession suivante pour cette construction par la méthode ECO

[e] On note ici σ aussi bien pour l'étiquetage du poset en échelle à d marches que pour la permutation qui lui est associée par la bijection de la proposition 6.15.

des étiquetages des posets en échelle :

$$\begin{cases} (2) \\ (k) \rightsquigarrow (2)(3)\cdots(k)(k+1) \end{cases}$$

Elle correspond bien aux classes combinatoires énumérées par les nombres de Catalan.

La figure 6.8 montre le début de l'arbre de génération associé à cette construction ECO. Pour plus de lisibilité, on ne représente pas les étiquetages des posets en échelle à d marches dans les nœuds de cet arbre, mais plutôt les permutations minimales à d descentes et de taille $2d$ qui leur sont associées.

Démonstration du théorème 6.16 par une méthode bijective Il apparaît clairement, dans la représentation des permutations minimales à d descentes et de taille $2d$ par des étiquetages du poset en échelle à d marches, que ces permutations sont en bijection avec des tableaux de Young à 2 lignes, chacune de longueur d. Une application directe de la formule des équerres donne alors le résultat du théorème 6.16. La définition des tableaux de Young et l'énoncé de la formule des équerres peuvent par exemple être trouvés dans [Nov99].

Ici, on fait le choix de donner une preuve bijective du théorème 6.16 sans faire intervenir les tableaux de Young, et en se ramenant directement aux objets combinatoires de base énumérés par les nombres de Catalan : les chemins de Dyck.

On décrit donc une bijection entre les chemins de Dyck de demi-longueur d et les étiquetages du poset en échelle à d marches. La définition des chemins de Dyck est rappelée au chapitre 10. Une propriété simple et bien connue (voir par exemple [Bai96]) des chemins de Dyck est la caractérisation suivante :

Proposition 6.17. *On considère un chemin du plan, partant de* $(0,0)$*, et composé d'autant de pas Nord-Est* $(1,1)$ *que de pas Sud-Est* $(1,-1)$*. Ce chemin est un chemin de Dyck si et seulement si le nombre de pas Nord-Est est supérieur ou égal au nombre de pas Sud-Est dans chacun de ses préfixes.*

On peut reformuler cette caractérisation des chemins de Dyck de la manière suivante :

Corollaire 6.18. *Un chemin du plan, partant de* $(0,0)$ *et composé d'autant de pas Nord-Est que de pas Sud-Est, est un chemin de Dyck si et seulement si le* i*-ème pas descendant a au moins* i *pas montants à sa gauche dans le chemin.*

La bijection annoncée se définit très simplement : partant d'un chemin de Dyck $\mathcal{D}$ de demi-longueur d, on numérote ses pas par les entiers de 1 à $2d$, de gauche à droite. On étiquette ensuite la ligne supérieure du poset en échelle par les entiers correspondant aux pas Sud-Est de $\mathcal{D}$, et la

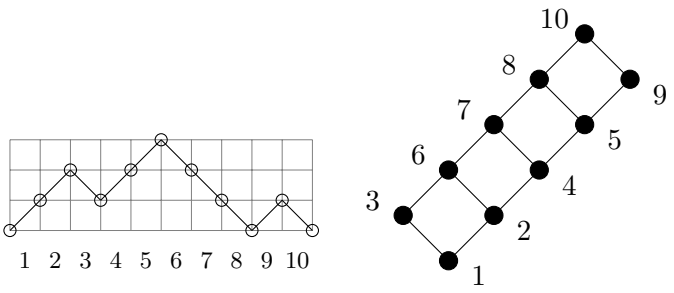


Fig. 6.9 – Illustration sur un exemple de la bijection décrite entre les permutations minimales à d descentes de taille $2d$ (vues comme des étiquetages du poset en échelle à d marches) et les chemins de Dyck de demi-longueur d.

ligne inférieure par les entiers correspondant aux pas Nord-Est de $\mathcal{D}$. Voir la figure 6.9 pour un exemple de cette construction.

Cette application est bien une bijection entre les chemins de Dyck et les étiquetages du poset en échelle, qui codent les permutations qui nous intéressent. Il suffit pour s'en convaincre de remarquer qu'un étiquetage du poset en échelle à d marches satisfait les contraintes d'ordre du poset si et seulement si pour tout i, le i-ème élément (noté x) sur la ligne supérieure du poset a au moins i éléments qui lui sont inférieurs sur la ligne inférieure du poset (l'élément y sur la ligne inférieure qui est relié à x par une marche de l'échelle, et tous les éléments qui sont en-dessous de y). La figure 6.10 donne une vision graphique de cette condition.

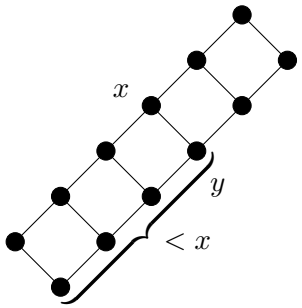


FIG. 6.10 – Une condition sur les étiquetages des posets en échelle pour qu'ils satisfassent les contraintes d'ordre imposées par le poset.

Cette condition nécessaire et suffisante qui caractérise les étiquetages des posets en échelle est équivalente à la caractérisation des chemins de Dyck donnée par le corollaire 6.18.

6.2.2.2 Taille $d+2$

Dans le paragraphe 6.2.2.1 ci-dessus, on s'est intéressé à l'énumération des permutations minimales à d descentes dont la taille est $2d$, c'est-à-dire de taille *maximale*. D'après le corollaire 6.14, la taille *minimale* d'une telle permutation est $d+1$, et nous avons vu plus haut que seule la permutation miroir de l'identité $((d+1)d\dots21)$ est minimale à d descentes et de taille $d+1$. On s'intéresse maintenant aux permutations minimales à d descentes et de taille $d+2$, c'est-à-dire de taille la plus petite possible, hormis les cas triviaux. On donne une formule close pour leur énumération dans le théorème 6.19.

Théorème 6.19. *Les permutations minimales à d descentes et de taille $d+2$ sont énumérées par la séquence (s_d) définie par $s_d = 2^{d+2} - (d+1)(d+2) - 2$.*

On propose deux preuves possibles du théorème 6.19. Toutes deux utilisent la représentation par posets des permutations minimales à d descentes et de taille $d+2$, qui ont par conséquent une unique montée. La première preuve est une application directe de cette représentation par posets, qui donne lieu à des calculs plutôt lourds. La seconde preuve proposée s'abstrait de ces lourdeurs calculatoires, au prix d'un raisonnement plus élaboré : le cœur de la preuve réside dans une correspondance entre les sous-ensembles de $\{1, 2, \dots, d+1\}$ qui ne sont pas des intervalles et les permutations minimales à d descentes et de taille $d+2$, chacun de ses ensembles étant associé à exactement deux permutations.

Démonstration du théorème 6.19 par une méthode calculatoire On rappelle que les permutations qui sont minimales à d descentes et de taille $d+2$ ont une unique montée, qui sépare deux séquences de descentes. Les quatre éléments autour de la montée sont organisés en un diamant dans la représentation par poset de la permutation.

Pour décrire une permutation σ minimale à d descentes et de taille $d+2$, il suffit donc de donner la partition des éléments de $\{1, 2, \ldots, d+2\}$ en deux ensembles A et B, A (resp. B) représentant les éléments de la première (resp. deuxième) séquence de descentes. Il y a cependant des contraintes pour le choix de cette partition : le cardinal de chacune des parts est compris entre 2 et d (de sorte à assurer que la première et la deuxième séquence de descentes de σ sont bien définies) et le choix qui est fait doit respecter la propriété du diamant. On dira qu'une telle partition (A, B) satisfait la propriété du diamant si la permutation σ associée satisfait cette propriété.

Le nombre de partitions de $\{1, 2, \ldots, d+2\}$ en deux parts (éventuellement vides) sans ces contraintes est 2^{d+2}. À cette valeur, il faut soustraire un total de $2(d+1)+2 = 2d+6$ pour les partitions où l'une des parts est de cardinal 0 ou 1 (l'autre part étant alors de cardinal $d+1$ ou $d+2$). Parmi les partitions dont les deux parts sont de cardinal compris entre 2 et d, il faut maintenant compter le nombre N de celles qui ne satisfont pas la propriété du diamant, pour obtenir le nombre de permutations minimales à d descentes et de taille $d+2$ comme étant la valeur $2^{d+2}-(2d+6)-N$.

On considère donc une partition (A, B) de $\{1, 2, \ldots, d+2\}$ en deux parts de cardinal compris entre 2 et d. On rappelle que A (resp. B) représente les éléments de la première (resp. deuxième) séquence de descentes d'une permutations σ, dont on veut assurer qu'elle vérifie la propriété du diamant. Pour ce faire, on note a_1 et a_2 les deux plus petits éléments de A, avec $a_1 < a_2$, et b_1 et b_2 les deux plus grands éléments de B, avec $b_1 < b_2$. Pour que σ satisfasse la propriété du diamant, il est nécessaire qu'au moins deux éléments de A soient inférieurs à b_2 et qu'au moins un élément de A soit inférieur à b_1.

La figure 6.11 représente les deux formes possibles d'une partition (A, B) qui satisfait la propriété du diamant.

FIG. 6.11 – Représentation schématique d'une partition (A, B) satisfaisant la propriété du diamant. Sur les $d+2$ cases représentant les entiers de 1 à $d+2$ en ordre croissant, les cases pleines correspondent aux éléments de B, les cases vides aux éléments de A, et les cases désignées par ? ? peuvent aussi bien appartenir à A qu'à B.

Les conditions nécessaires indiquées ci-dessus sont également suffisantes. Les partitions (A, B) qui ne satisfont pas la propriété du diamant sont donc celles vérifiant l'une des conditions suivantes :

(a) il n'y a aucun élément de A qui soit inférieur à b_2,
(b) il y a un unique élément de A qui est inférieur à b_2,
(c) il n'y a aucun élément de A qui soit inférieur à b_1, mais il y a au moins deux éléments de A compris entre b_1 et b_2.

Il est important de remarquer que ces trois conditions sont mutuellement exclusives : deux (ou trois) d'entre elles ne peuvent jamais être satisfaites en même temps.

Les formes possibles des partitions (A, B) qui satisfont les conditions (a), (b) ou (c) sont représentées schématiquement sur la figure 6.12, où sur les $d+2$ cases représentant les entiers de 1 à $d+2$ en ordre croissant, les cases pleines correspondent aux éléments de B et les cases vides aux éléments de A.

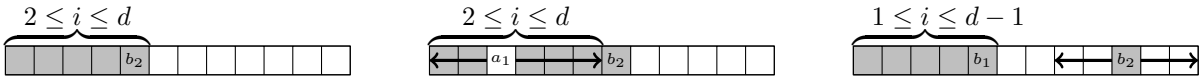
Il est ensuite immédiat de compter les partitions (A, B) satisfaisant l'une des propriétés (a), (b) ou (c). Il y en a :

(a) $d-1$ qui satisfont la propriété (a),
(b) $\sum_{i=2}^{d} i = \frac{d(d+1)}{2} - 1$ qui satisfont la propriété (b),
(c) et $\sum_{i=1}^{d-1}(d-i) = \frac{d(d-1)}{2}$ qui satisfont la propriété (c).

On obtient ainsi qu'il y a

$$2^{d+2} - (2d+6) - (d-1) - (\frac{d(d+1)}{2} - 1) - \frac{d(d-1)}{2} = 2^{d+2} - (d+1)(d+2) - 2$$

permutations minimales à d descentes et de taille $d+2$.



(a) Partitions satisfaisant la propriété (a)

(b) Partitions satisfaisant la propriété (b)

(c) Partitions satisfaisant la propriété (c)

FIG. 6.12 – Les partitions de $\{1, 2, \ldots, d+2\}$ en deux parts de cardinal compris entre 2 et d qui ne satisfont pas la propriété du diamant. La valeur i sur ces schémas indique le nombre de cases dans la zone sous l'accolade. Sur la figure (b) (resp. (c)), la flèche matérialise la zone où peut se trouver la case a_1 (resp. b_2).

Ceci termine la preuve du théorème 6.19 par une méthode calculatoire. On remarque que cette méthode calculatoire est différente de celle donnée dans l'article [BP08]. On propose maintenant une preuve bijective du même résultat.

Démonstration du théorème 6.19 par une méthode bijective Un *sous-ensemble non intervallaire* de $\{1, 2, \ldots, d+1\}$ (en anglais *non-interval subset*) est un sous-ensemble de $\{1, 2, \ldots, d+1\}$ qui n'est pas un intervalle. En particulier, un tel sous-ensemble n'est jamais vide, ni réduit à un élément.

Par exemple, les sous-ensembles non intervallaires de $\{1, \ldots, 4\}$ sont $\{1, 3\}$, $\{1, 4\}$, $\{2, 4\}$, $\{1, 2, 4\}$ et $\{1, 3, 4\}$.

On peut facilement calculer le nombre de sous-ensembles non intervallaires de $\{1, 2, \ldots, d+1\}$ en fonction de d. Ce résultat d'énumération est donné par la proposition suivante :

Proposition 6.20. *Le nombre de sous-ensembles non intervallaires de* $\{1, 2, \ldots, d+1\}$ *est* $2^{d+1} - \frac{(d+1)(d+2)}{2} - 1$.

Démonstration. Il y a 2^{d+1} sous-ensembles de $\{1, 2, \ldots, d+1\}$, l'un étant l'ensemble vide. Il suffit donc de prouver que le nombre d'intervalles non vides de $\{1, 2, \ldots, d+1\}$ est $\frac{(d+1)(d+2)}{2}$. On voit facilement qu'il y a i intervalles inclus dans $\{1, 2, \ldots, d+1\}$ dont le plus grand élément est i : il s'agit des intervalles $[j..i]$ pour tout j tel que $1 \leq j \leq i$. L'identité bien connue $\sum_{i=1}^{d+1} i = \frac{(d+1)(d+2)}{2}$ conclut alors la preuve de la proposition. □

Il faut remarquer que la séquence $(2^{d+1} - \frac{(d+1)(d+2)}{2} - 1)_d$ est répertoriée dans l'Encyclopédie en ligne des Séquences d'Entiers [Slo07] avec le numéro [A002662].

Pour prouver le théorème 6.19 par une méthode bijective, on peut se ramener aux sous-ensembles non intervallaires de $\{1, 2, \ldots, d+1\}$ et démontrer qu'il y a deux fois plus de permutations minimales à d descentes et de taille $d+2$ que de tels sous-ensembles. Dans ce but, on partitionne l'ensemble des permutations minimales à d descentes et de taille $d+2$ en deux parts S_1 et S_2, et on exhibe deux bijections Φ^1 et Φ^2 entre S^1 (resp. S^2) et l'ensemble des sous-ensembles non intervallaires de $\{1, 2, \ldots, d+1\}$. Dans la suite, on notera $\mathcal{NI}$ cet ensemble.

On définit les ensembles S^1 et S^2 comme suit. L'ensemble S^1 contient les permutations σ minimales à d descentes et de taille $d+2$ telles que (1) $d+2$ est l'élément maximal de l'unique montée de σ et (2) la première séquence de descentes de σ n'est pas formée d'éléments dont les valeurs sont toutes consécutives. L'ensemble S^2 contient toutes les autres permutations minimales à d descentes et de taille $d+2$. Sur la figure 6.13, on peut voir les formes possibles des permutations de S^1 et S^2.

On décrit d'abord la bijection Φ^1 entre $\mathcal{NI}$ et S^1. On considère un sous-ensemble non intervallaire s de $\{1, 2, \ldots, d+1\}$. On appelle t l'ensemble des "trous" associé à s : $t = \{1, 2, \ldots, d+1\} \setminus s$. On

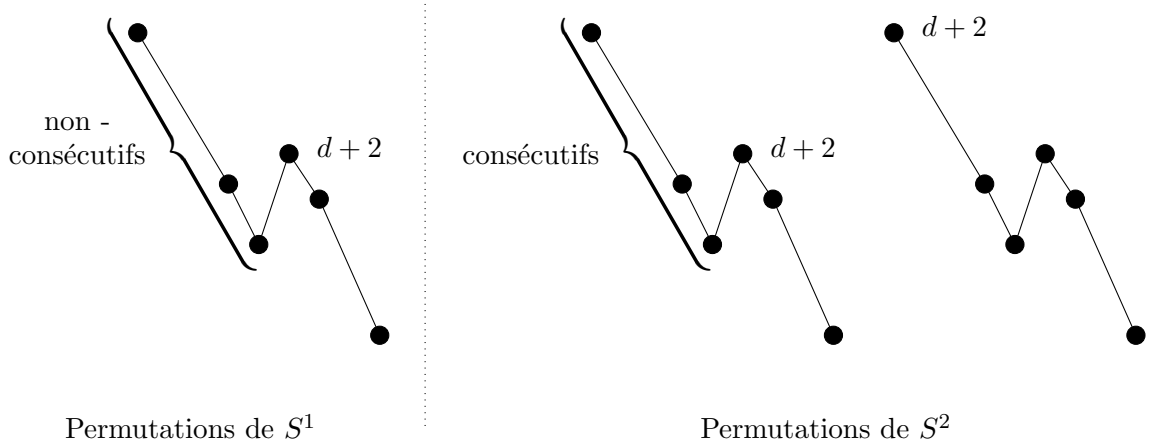


FIG. 6.13 – Les différentes formes des permutations dans les ensembles S^1 et S^2.

définit alors $\Phi^1(s)$ comme la permutation qui commence par les éléments de s, en ordre décroissant, puis se poursuit avec l'élément $d+2$, et enfin se termine par les éléments de t en ordre décroissant. Cette définition est illustrée en figure 6.14.

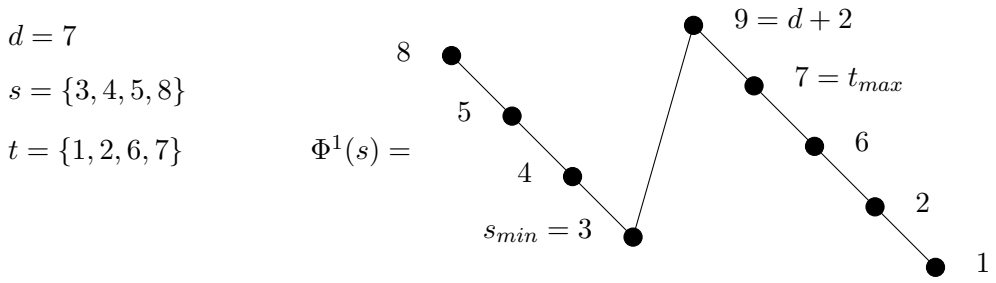


FIG. 6.14 – Illustration de la définition de la bijection Φ^1 sur un exemple.

Proposition 6.21. *L'application Φ^1 est une bijection entre $\mathcal{NI}$ et S^1.*

Démonstration. Soit s un sous-ensemble non intervallaire de $\{1, 2, \ldots, d+1\}$, et soit t l'ensemble des trous associés $t = \{1, 2, \ldots, d+1\} \setminus s$.

On démontre d'abord que $\Phi^1(s) \in S^1$. Comme $s \in \mathcal{NI}$, s contient au moins deux éléments, et t au moins un. Par conséquent, $\Phi^1(s)$ est composée de deux séquences non vides de descentes séparées par une montée, et pour conclure que $\Phi^1(s)$ est une permutation minimale à d descentes et de taille $d+2$, il reste seulement à vérifier que la propriété du diamant est satisfaite autour de cette montée. Dans notre cas, pour démontrer cette propriété du diamant, il suffit de vérifier que l'élément minimal s_{min} de s est inférieur à l'élément maximal t_{max} de t. Comme s n'est pas un intervalle, il existe au moins un élément de t qui est supérieur à s_{min}, et ainsi $s_{min} < t_{max}$.

En utilisant une fois encore le fait que s n'est pas un intervalle, on conclut immédiatement que $\Phi^1(s) \in S^1$.

Sachant que parmi les permutations minimales à d descentes et de taille $d+2$, les permutations de S^1 sont celles telles que les éléments de leur première séquence de descentes ne forment pas un intervalle, il apparaît alors clairement que Φ^1 est une bijection entre $\mathcal{NI}$ et S^1. □

La bijection entre $\mathcal{NI}$ et S^2 est plus complexe, et nécessite de classer les permutations de S^2 en cinq types, de A à E. Ces différents types sont illustrés sur la figure 6.15.

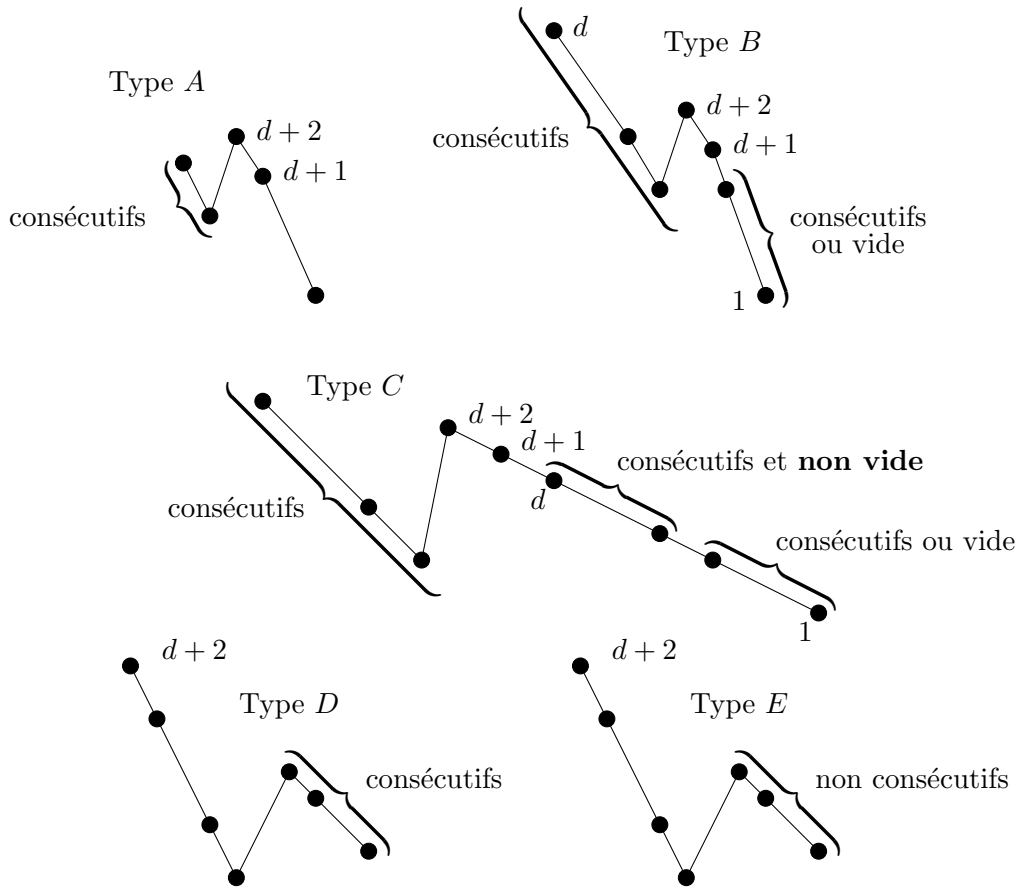


FIG. 6.15 – La classification des permutations de S^2 en cinq types, de A à E.

Les permutations σ de type A sont celles de S^2 telles que (1) $d+2$ est le second élément de la montée de σ, et (2) la première séquence de descentes de σ contient exactement deux éléments, qui sont consécutifs.

Les permutations σ de type B sont celles de S^2 telles que (1) $d+2$ est le second élément de la montée de σ, (2) la première séquence de descentes de σ est composée d'éléments consécutifs, et contient au moins trois éléments, et (3) la seconde séquence de descentes de σ est de la forme $(d+2)(d+1)r$, où r est soit vide, soit une séquence décroissante d'éléments consécutifs dont l'élément minimal est 1.

Les permutations σ de type C sont celles de S^2 telles que (1) $d+2$ est le second élément de la montée de σ, (2) la première séquence de descentes de σ est composée d'éléments consécutifs, et contient au moins trois éléments, et (3) la seconde séquence de descentes de σ est de la forme $(d+2)(d+1)r_1r_2$, où r_1 est une séquence **non vide** et décroissante d'éléments consécutifs dont l'élément maximal est d, et r_2 est soit vide, soit une séquence décroissante d'éléments consécutifs dont l'élément minimal est 1.

Les permutations σ de type D sont celles de S^2 telles que (1) $d+2$ est le premier élément de σ, et (2) les éléments de la seconde descente de σ sont consécutifs.

Les permutations σ de type E sont celles de S^2 telles que (1) $d+2$ est le premier élément de σ, et (2) les éléments de la seconde descente de σ ne sont pas consécutifs.

Proposition 6.22. *Chaque permutation de S^2 a un type unique, entre A et E.*

Démonstration. On distingue deux cas, en fonction de la position de $d+2$ dans σ : $d+2$ est soit le premier élément de σ, soit le second élément de la montée de σ. Dans le premier cas, il est clair

que le type de σ est unique et vaut D ou E. Supposons donc que $d+2$ est le second élément de la montée de σ. Dans ce cas, puisque $\sigma \in S^2$, les éléments de la première séquence de descentes de σ doivent être consécutifs.

Examinons les positions possibles de $d+1$ dans σ. Si $d+1$ est le premier élément de σ, alors, étant donné que la première séquence de descentes de σ est composée d'éléments consécutifs, la propriété du diamant ne peut pas être satisfaite autour de la montée de σ. En effet, dans une telle situation, il est impossible que l'élément le plus à droite du diamant soit supérieur à celui le plus en bas. Ainsi, la seule position possible pour $d+1$ dans σ est celle juste à droite de $d+2$.

Si la première séquence de descentes de σ est réduite à deux éléments, alors σ est de type A. Si au contraire cette première séquence de descentes de σ contient au moins trois éléments, alors σ est de type C ou B, selon que l'élément juste à droite de $d+1$ dans σ est ou n'est pas d. Dans chacun de ces deux cas, comme la première séquence de descentes de σ est composée d'éléments consécutifs, il est facile de voir que la seconde séquence de descente est bien décomposée en $(d+2)(d+1)r_1r_2$ ou $(d+2)(d+1)r$, avec r, r_1 et r_2 des séquences décroissantes d'éléments consécutifs, pour correspondre aux définitions des types C et B données plus haut. □

On dispose à présent des outils pour définir l'application Φ^2 de $\mathcal{NI}$ dans S^2, et pour prouver que c'est une bijection.

On considère un sous-ensemble non intervallaire s de $\{1, 2, \ldots, d+1\}$, et on note t l'ensemble des trous associé $t = \{1, 2, \ldots, d+1\} \setminus s$.

1. Si t contient un unique élément x, alors nécessairement $x \neq 1$ et $x \neq d+1$, sinon s serait un intervalle. Dans ce cas, $\Phi^2(s)$ est la permutation de type A dont la première descente est $x(x-1)$. Cette permutation satisfait de manière évidente la propriété du diamant (voir une illustration en figure 6.16).

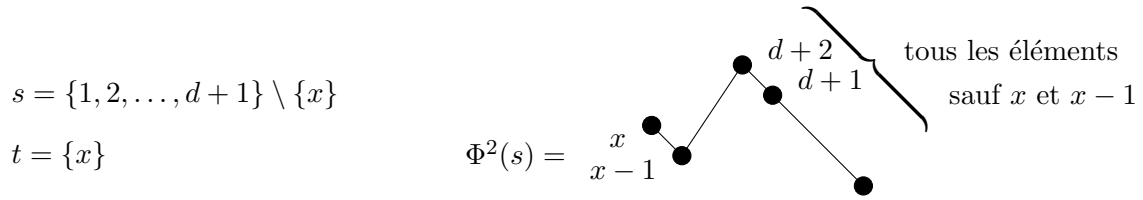


FIG. 6.16 – Définition de la bijection Φ^2 pour s tel que $|t| = 1$.

Si t contient au moins deux éléments, on note n le cardinal de s et m le cardinal de t augmenté en 1. On peut remarquer que $m \geq 3$ et $n \geq 2$. On notera aussi t_1 et t_2 les deux plus petits éléments de t (avec $t_1 < t_2$), et s_n and s_{n-1} les deux plus grands éléments de s (avec $s_{n-1} < s_n$). La permutation de S^2 associée à s par la bijection Φ^2 aura m éléments sur sa première séquence de descentes, et n sur la seconde, et va être définie en fonction de l'ordre relatif de t_1, t_2, s_n et s_{n-1}.

En fait, seuls quelques ordres relatifs de ces éléments entre eux sont acceptables, car ils doivent satisfaire les conditions $t_1 < t_2$, $s_{n-1} < s_n$, et $t_1 < s_n$ (sinon s serait un intervalle). Ces contraintes donnent lieu à cinq ordres possibles.

2. Si $t_1 < t_2 < s_{n-1} < s_n$ or $t_1 < s_{n-1} < t_2 < s_n$, alors $\Phi^2(s)$ est la permutation de type E obtenue de la façon suivante : le premier élément est $d+2$, suivi par les éléments de t en ordre décroissant, et enfin les éléments de s en ordre décroissant. Les contraintes satisfaites par t_1, t_2, s_n et s_{n-1} assurent que cette permutation satisfait la propriété du diamant (voir un exemple à la figure 6.17).

$s = \{s_1, s_2, \ldots, s_n\}$ avec $s_1 < s_2 < \ldots < s_n$

$t = \{t_1, t_2, \ldots, t_{m-1}\}$ avec $t_1 < t_2 < \ldots < t_{m-1}$

$t_1 < t_2 < s_{n-1} < s_n$ ou $t_1 < s_{n-1} < t_2 < s_n$

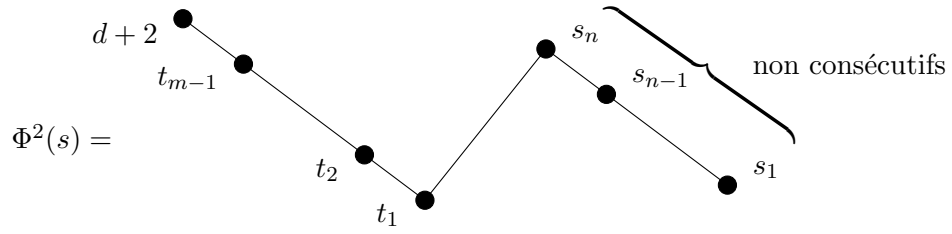


FIG. 6.17 – Définition de la bijection Φ^2 pour s tel que $t_1 < t_2 < s_{n-1} < s_n$ ou $t_1 < s_{n-1} < t_2 < s_n$.

3. Si $s_{n-1} < t_1 < t_2 < s_n$, alors l'ensemble non intervallaire s est complètement déterminé par le cardinal n de s et la valeur de l'élément maximal s_n de s. Il est nécessaire en effet que s soit égal à $\{1, 2, \ldots, n-1\} \uplus \{s_n\}$ pour que la condition $s_{n-1} < t_1 < t_2 < s_n$ soit satisfaite. Dans ce cas, on associe à s une permutation de type D en procédant comme suit. Le premier élément de $\Phi^2(s)$ est $d+2$, la seconde séquence de descentes de $\Phi^2(s)$ est constituée de n éléments consécutifs en ordre décroissant, avec s_n comme élément maximal, et les éléments restants sont placés en ordre décroissant après $d+2$ pour compléter la première séquence de descentes de $\Phi^2(s)$. Pour prouver que cette permutation est bien de type D, il faut vérifier qu'elle appartient bien à S^2, c'est-à-dire qu'elle satisfait bien la propriété du diamant. On voit facilement qu'il y a au moins $n+1$ éléments inférieurs à s_n : les $n-1$ autres éléments de s, plus t_1 et t_2. Ainsi, les éléments 1 et 2 ne peuvent pas être dans la deuxième séquence de descentes de $\Phi^2(s)$, et donc la première séquence de descentes de $\Phi^2(s)$ s'achève par 2 1. Cette remarque suffit à démontrer la propriété du diamant. Un exemple est donné par la figure 6.18.

$s_{n-1} < t_1 < t_2 < s_n$

$s = \{1, 2, \ldots, n-1\} \uplus \{s_n\}$

$t = \{n, n+1, \ldots, s_n - 1\} \uplus \{s_n + 1, \ldots, d+1\}$

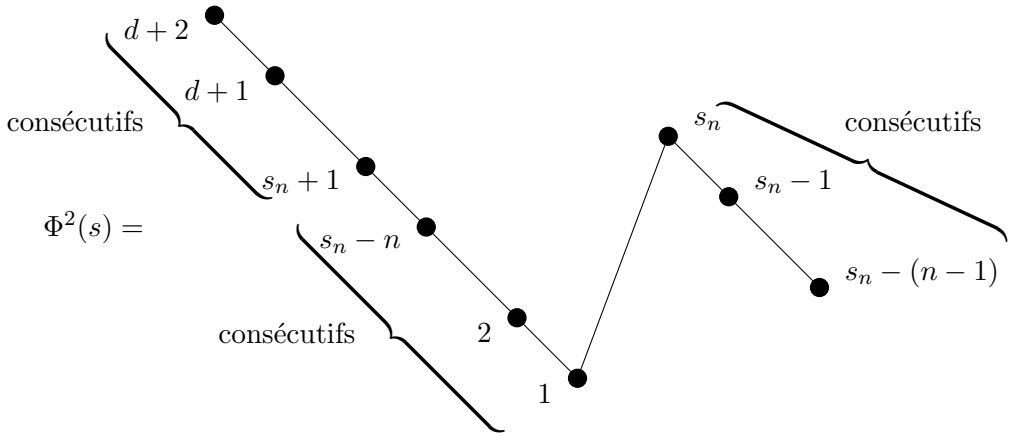


FIG. 6.18 – Définition de la bijection Φ^2 pour s tel que $s_{n-1} < t_1 < t_2 < s_n$.

4. Si $t_1 < s_{n-1} < s_n < t_2$, les éléments de $\{1, 2, \ldots, d+1\}$ sont répartis en $s \uplus t$ de la manière suivante : $s = \{1, \ldots, t_1 - 1\} \uplus \{t_1 + 1, \ldots, n+1\}$ et $t = \{t_1\} \uplus \{t_2 = n+2, \ldots, d+1\}$. L'ensemble non intervallaire s est complètement déterminé par le cardinal n de s et le nombre $p = n+1-t_1$ d'éléments de s de valeur comprise entre t_1 et t_2. Comme s_{n-1} et s_n sont compris entre t_1 et t_2, $p \geq 2$. On a aussi $p \leq n-1$ (pour $p = n$, s serait un intervalle). Dans ce cas, on associe à s la permutation $\Phi^2(s)$ de type C en procédant ainsi. La seconde séquence de descentes de $\Phi^2(s)$ est divisée en deux parties (la deuxième pouvant être vide). La première partie contient $p+1$ éléments (on a bien $3 \leq p+1 \leq n$) qui sont consécutifs et dont l'élément maximal est $d+2$, écrits en ordre décroissant. La seconde partie est composée de $n-p-1$ éléments consécutifs, en ordre décroissant, et dont l'élément minimal est 1. Jusqu'ici, cette construction laisse de côté m éléments, qui sont consécutifs : ils vont former la première séquence de descentes de $\Phi^2(s)$. Il est alors facile de démontrer la propriété du diamant, la seconde séquence de descentes de $\Phi^2(s)$ commençant forcément par $(d+2)(d+1)$. Cette remarque conclut la preuve que la permutation $\Phi^2(s)$ ainsi définie est dans S^2, et de type C (voir un exemple en figure 6.19).

$t_1 < s_{n-1} < s_n < t_2$

$s = \{1, \ldots, t_1 - 1\} \uplus \{t_1 + 1, \ldots, n+1\}$

$t = \{t_1\} \uplus \{n+2, \ldots, d+1\}$

On pose $p = n + 1 - t_1$

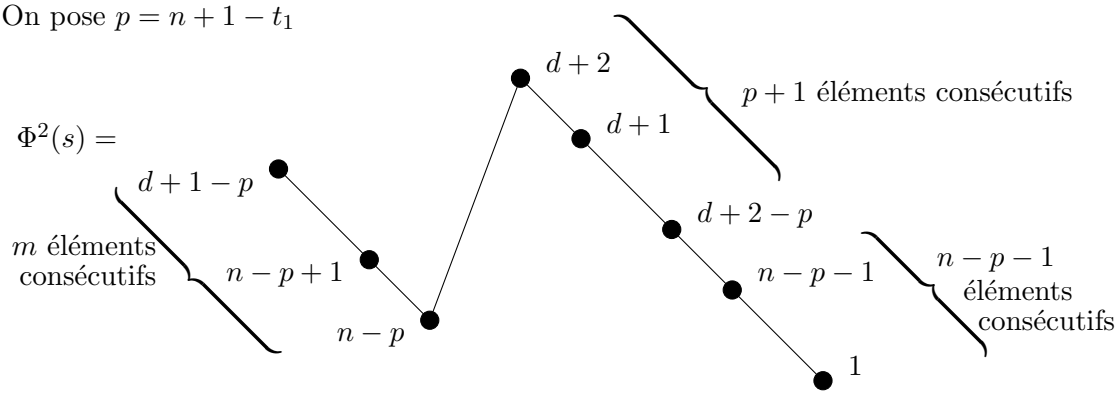


FIG. 6.19 – Définition de la bijection Φ^2 pour s tel que $t_1 < s_{n-1} < s_n < t_2$.

5. Il reste à examiner seulement le cas où $s_{n-1} < t_1 < s_n < t_2$. Ce cas est particulièrement simple car le cardinal n de s détermine s complètement : pour pouvoir satisfaire la contrainte $s_{n-1} < t_1 < s_n < t_2$, on a nécessairement $s = \{1, \ldots, n-1\} \uplus \{n+1\}$. La permutation $\Phi^2(s)$ associée à s dans ce cas est de type B. Sa seconde séquence de descentes contient n éléments : d'abord $(d+2)(d+1)$, puis une séquence décroissante d'entiers consécutifs dont la valeur minimale est 1 (cette séquence étant vide si $n = 2$). Il reste donc m éléments consécutifs, dont le maximum est d, qui (en ordre décroissant) vont former la première séquence de descentes de $\Phi^2(s)$. Comme la deuxième séquence de descentes commence par $(d+2)(d+1)$, il est clair que $\Phi^2(s)$ satisfait la propriété du diamant, ce qui justifie que $\Phi^2(s)$ est une permutation de S^2 et de type B. La figure 6.20 donne une illustration de la construction dans ce cas.

$s_{n-1} < t_1 < s_n < t_2$

$s = \{1, \ldots, n-1\} \uplus \{n+1\}$

$t = \{n\} \uplus \{n+2, \ldots, d+1\}$

$\Phi^2(s) =$

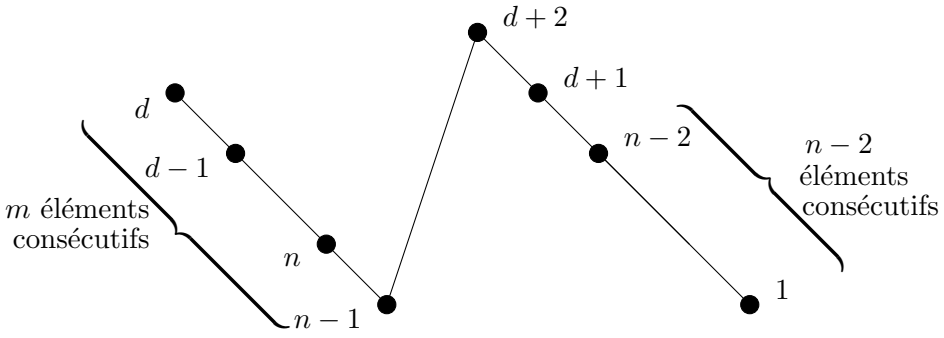


FIG. 6.20 – Définition de la bijection Φ^2 pour s tel que $s_{n-1} < t_1 < s_n < t_2$.

La figure 6.21 donne différents exemples de permutations obtenues par la bijection $\Phi^2(s)$.

Ceci termine la définition de l'application $\Phi^2 : \mathcal{NI} \to S^2$. Il reste à démontrer que c'est une bijection :

Proposition 6.23. *L'application Φ^2 définit une bijection entre $\mathcal{NI}$ et S^2.*

Démonstration. L'application inverse de Φ^2, de S^2 dans $\mathcal{NI}$, peut facilement être définie grâce à l'étude qui précède, en distinguant des cas en fonction du type (de A à E) d'une permutation de S^2.

On considère par exemple le cas d'une permutation σ de type D. On note n le nombre d'éléments de σ dans la seconde séquence de descentes, et s_{max} le second élément de la montée de σ. Parce que σ vérifie la propriété du diamant, on vérifie que $s_{max} \geq n+2$, et $(\Phi^2)^{-1}(\sigma)$ est $s = \{1, 2, \ldots, n-1\} \uplus \{s_{max}\}$. □

Pour reprendre les étapes principales de cette preuve bijective du théorème 6.19, on a d'abord décrit une partition de l'ensemble des permutations minimales à d descentes et de taille $d+2$ en $S^1 \uplus S^2$. On a ensuite donné deux bijections Φ^1 et Φ^2 respectivement de S^1 et S^2 dans $\mathcal{NI}$. En utilisant le résultat d'énumération des sous-ensembles non intervallaires de $\{1, 2, \ldots, d+1\}$ (proposition 6.20), on obtient bien la démonstration bijective annoncée du théorème 6.19.

6.2.2.3 Questions ouvertes d'énumération

On s'est intéressé dans cette partie à l'étude (caractérisation et énumération) des permutations qui sont minimales pour la propriété d'avoir d descentes, la minimalité étant entendue au sens de la relation de motif. Pour $d = 2^p$, ces permutations apparaissent naturellement dans le modèle d'évolution de génomes dit de "duplication complète avec perte aléatoire", où elles forment la base de motifs exclus caractérisant la classe des permutations obtenues en au plus p étapes dans ce modèle.

La caractérisation locale des permutations minimales à d descentes décrite au théorème 6.13 se concentre sur les éléments de ces permutations qui entourent leurs montées. D'un point de vue algorithmique, on voit immédiatement que cette caractérisation fournit un outil pour tester en temps linéaire si une permutation donnée est minimale à d descentes ou non. D'un point de vue combinatoire, elle permet de représenter les permutations minimales à d descentes par des posets, représentation grâce à laquelle nous avons obtenu des résultats partiels d'énumération.

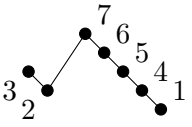
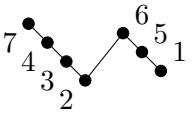
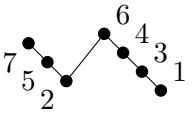
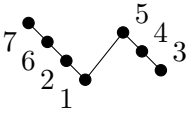
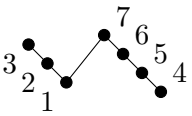
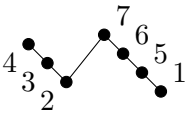
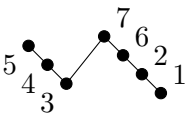
Cas pour s	Exemple de s	$\Phi^2(s)$	Type
(1) avec $t = \{x\}$	$s = \{1, 2, 4, 5, 6\}$ $t = \{3\}$ $x = 3$		A
(2) avec $t_1 < t_2 < s_{n-1} < s_n$	$s = \{1, 5, 6\}$ $t = \{2, 3, 4\}$		E
(2) avec $t_1 < s_{n-1} < t_2 < s_n$	$s = \{1, 3, 4, 6\}$ $t = \{2, 5\}$		E
(3) avec $s_{n-1} < t_1 < t_2 < s_n$	$s = \{1, 2, 5\}$ $t = \{3, 4, 6\}$ $\lvert s\rvert = 3$, $s_n = 5$		D
(4) avec $t_1 < s_{n-1} < s_n < t_2$	$s = \{1, 3, 4, 5\}$ $t = \{2, 6\}$ $\lvert s\rvert = 4$, $p = 3$		C
(4) avec $t_1 < s_{n-1} < s_n < t_2$	$s = \{1, 2, 4, 5\}$ $t = \{3, 6\}$ $\lvert s\rvert = 4$, $p = 2$		C
(5) avec $s_{n-1} < t_1 < s_n < t_2$	$s = \{1, 2, 3, 5\}$ $t = \{4, 6\}$ $\lvert s\rvert = 4$		B

FIG. 6.21 – Définition de $\Phi^2(s)$ pour quelques exemples de sous-ensembles non intervallaires s de $\{1, 2, \ldots, 6\}$ ($d = 5$), qui illustrent tous les cas possibles dans la construction de Φ^2.

On a montré que la taille d'une permutation minimale à d descentes est comprise entre $d + 1$ et $2d$, et on a énuméré ces permutations pour les tailles $d + 1$, $d + 2$ et $2d$. L'énumération des permutations minimales à d descentes et de taille $n \in [(d+3)..(2d-1)]$ reste une question ouverte. Pour $n = d+3$ et $n = 2d-1$, on a calculé les premiers termes des séquences d'énumération associées, et elles ne semblent pas figurer dans l'Encyclopédie en ligne des Séquences d'Entiers [Slo07].

On peut cependant remarquer que la méthode calculatoire utilisée pour énumérer les permutations minimales à d descentes de taille $d + 2$ est (en théorie) applicable pour toutes les tailles $n \in [(d+3)..(2d-1)]$, même si les cas à considérer seraient beaucoup plus nombreux. Cette explosion combinatoire suggère que pour résoudre le problème d'énumération des permutations minimales à d descentes, soit on peut trouver une méthode automatique pour examiner tous les cas, soit il faut envisager d'autres techniques. Une tentative dans ce sens, en collaboration avec Luca Ferrari, et utilisant des tableaux de Young, est en cours d'étude. Nous obtenons le nombre de permutations minimales à d descentes et de taille n sous la forme d'une somme de déterminants, dont on ne sait pas pour l'instant extraire de formule close (ou tout au moins sympathique).

6.3 Analyse combinatoire du modèle général de "duplication en tandem avec perte aléatoire"

On revient dans cette section à l'étude des permutations obtenues à partir de l'identité en au plus p étapes de largeur au plus K dans le modèle général de duplication en tandem avec perte aléatoire. On rappelle que la notation $\mathcal{C}(K,p)$ désigne la classe des permutations pour lesquelles il existe un scénario de duplication-perte en au plus p étapes de largeur au plus K. On étudie les bases de ces classes : dans le cas $\mathcal{C}(K,1)$ on décrit complètement la base, et pour le cas général $\mathcal{C}(K,p)$ on donne une borne sur la taille des motifs exclus.

6.3.1 Permutations obtenues en une étape de largeur au plus K

Avant de s'attaquer à l'étude des classes $\mathcal{C}(K,p)$, on se concentre ici sur le cas plus simple de la classe $\mathcal{C}(K) = \mathcal{C}(K,1)$ des permutations obtenues à partir de $12\dots n$ en une étape de largeur au plus K. Pour toute cette étude, on va supposer qu'une valeur $K \geq 2$ du paramètre est fixée, et par "étape" on entendra toujours "étape de duplication-perte de largeur au plus K", sauf mention contraire.

On remarque facilement que les permutations de $\mathcal{C}(K)$ ne peuvent pas avoir plus d'une descente. Réciproquement, les permutations de taille au plus K qui ont une unique descente appartiennent à $\mathcal{C}(K)$. La proposition 6.24 est un point technique très simple mais essentiel à la preuve du théorème 6.25 :

Proposition 6.24. *Les permutations de taille $K+1$ qui n'appartiennent pas à $\mathcal{C}(K)$ et qui possèdent une unique descente sont les permutations de $\mathcal{S}_{K+1}$ avec une unique descente, qui ne commencent pas par 1 et ne se terminent pas par $K+1$.*

Démonstration. Soit $\sigma = \sigma_1\sigma_2\dots\sigma_{K+1}$ une permutation de taille $K+1$ qui n'appartient pas à $\mathcal{C}(K)$ mais possède une unique descente. Si $\sigma_1 = 1$, alors $\overline{\sigma} = \sigma_2\dots\sigma_{K+1}$ est une permutation (de $\{2,3,\dots,K+1\}$) de taille K possédant une unique descente et par conséquent $\overline{\sigma}$ peut être obtenue à partir de $23\dots K+1$ en une étape de duplication-perte. Si on applique ce scénario de duplication-perte (réduit à une étape) à $123\dots K+1$, la permutation obtenue sera σ, ce qui contredit le fait que $\sigma \notin \mathcal{C}(K)$. Par un raisonnement identique, on obtient aussi une contradiction dans le cas où $\sigma_{K+1} = K+1$. Ainsi, σ ne commence pas par 1 et ne se termine pas par $K+1$.

On considère maintenant une permutation σ de taille $K+1$ ayant une unique descente, qui ne commence pas par 1 et ne se termine pas par $K+1$. On voit facilement que σ ne peut pas être obtenue en une seule étape de duplication-perte à partir de $12\dots K+1$: en effet, aucune étape de duplication-perte de largeur K appliquée à $12\dots K+1$ ne pourra déplacer à la fois 1 et $K+1$. □

Théorème 6.25. *La classe $\mathcal{C}(K)$ des permutations obtenues à partir de $12\dots n$ (pour un certain $n \geq 1$) en une étape de duplication-perte de largeur K est une classe $\mathcal{S}(\mathcal{B})$ de permutations à motifs exclus dont la base $\mathcal{B}$ est composée de $3 + 2^{K-1}$ motifs de taille au plus $K+1$. Plus précisément, $\mathcal{B} = \{3\,2\,1, 3\,1\,4\,2, 2\,1\,4\,3\} \cup D$, D étant l'ensemble de toutes les permutations de $\mathcal{S}_{K+1}$ qui ne commencent pas par 1, ne se terminent pas par $K+1$, et contiennent exactement une descente.*

Par exemple $\mathcal{C}(4) = \mathcal{S}(3\,2\,1, 3\,1\,4\,2, 2\,1\,4\,3, 2\,3\,4\,5\,1, 2\,3\,5\,1\,4, 2\,4\,5\,1\,3, 3\,4\,5\,1\,2, 2\,5\,1\,3\,4, 3\,5\,1\,2\,4, 4\,5\,1\,2\,3, 5\,1\,2\,3\,4)$.

Démonstration. Pour prouver le théorème 6.25, on démontre plutôt sa contraposée : $\sigma \notin \mathcal{S}(\mathcal{B})$ si et seulement si σ ne peut pas être obtenue à partir d'une permutation identité en une seule étape de duplication-perte de largeur K.

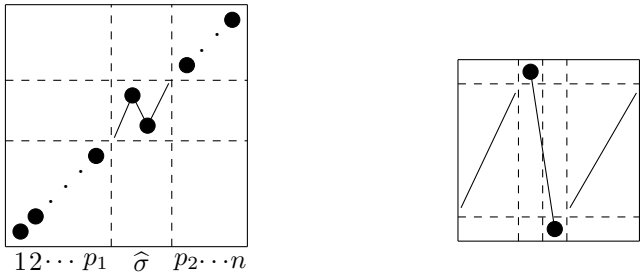
Supposons d'abord que $\sigma \notin \mathcal{S}(\mathcal{B})$. Dans ce cas, il existe $b \in \mathcal{B}$ tel que $b \preceq \sigma$. Si $b = 3\,2\,1$, $3\,1\,4\,2$ ou $2\,1\,4\,3$, alors σ a au moins 2 descentes et ne peut pas être obtenue en une seule étape de duplication-perte. Sinon, d'après la proposition 6.24, il existe $\rho \in \mathcal{S}_{K+1}$ tel que $\rho \preceq \sigma$ et $\rho \notin \mathcal{C}(K)$.

Ainsi, si σ pouvait être obtenue en une seule étape de duplication-perte de largeur au plus K, alors ρ le serait aussi, ce qui est une contradiction. On conclut que $\sigma \notin \mathcal{C}(K)$.

Réciproquement, supposons que $\sigma \notin \mathcal{C}(K)$. Ceci implique en particulier que σ n'est pas une permutation identité. Si σ contient au moins 2 descentes, alors σ contient une occurrence d'un motif parmi 3 2 1, 3 1 4 2 et 2 1 4 3, puisque ces trois permutations sont les minimales (au sens de la relation $\preceq$) ayant 2 descentes. Par conséquent, $\sigma \notin \mathcal{S}(\mathcal{B})$. Il nous reste à considérer le cas où σ a exactement une descente. On décompose $\sigma \in \mathcal{S}_n$ sous la forme $\sigma = 12\dots p_1\widehat{\sigma}p_2(p_2+1)\dots n$, où $\widehat{\sigma}$ est une permutation de l'ensemble $\{p_1+1, p_1+2\dots, p_2-1\}$ qui ne commence pas par p_1+1, ne se termine pas par p_2-1, et contient exactement une descente. Cette décomposition est illustrée en figure 6.22(a). On note $\widehat{K}$ la taille de la permutation $\widehat{\sigma}$. Comme $\sigma \notin \mathcal{C}(K)$, on a nécessairement $\widehat{K} \geq K+1$ ou alors on obtiendrait une contradiction. Si $\widehat{K} = K+1$, on trouve directement que $\widehat{\sigma}$ est une occurrence d'un motif de $D \subseteq \mathcal{B}$ dans σ. Par conséquent, $\sigma \notin \mathcal{S}(\mathcal{B})$. Il reste maintenant à étendre la preuve au cas général où $\widehat{K} > K+1$. Pour cela, il suffit de démontrer que l'on peut enlever des éléments dans $\widehat{\sigma}$ en respectant toujours les propriétés suivantes :

- la permutation ne commence pas par son élément minimal
- la permutation ne se termine pas par son élément maximal
- la permutation a exactement une descente

jusqu'à obtenir une permutation de taille $K+1$. À ce moment là, on aura mis en évidence que $\widehat{\sigma}$ contient un motif de D, ce qui sera donc aussi le cas pour σ, prouvant que $\sigma \notin \mathcal{S}(\mathcal{B})$. De par les conditions qui définissent $\widehat{\sigma}$, l'unique descente de $\widehat{\sigma}$ va nécessairement de son élément maximal vers son élément minimal. Ceci assure en particulier qu'il est possible d'enlever des éléments dans $\widehat{\sigma}$ en respectant toujours les trois propriétés ci-dessus (voir figure 6.22(b)). □



(a) Décomposition de σ (b) Forme de $\widehat{\sigma}$

FIG. 6.22 – Décomposition $\sigma = 12\dots p_1\widehat{\sigma}p_2(p_2+1)\dots n$ sur la représentation graphique de σ, et forme de $\widehat{\sigma}$.

Le théorème 6.25 donne une caractérisation complète de $\mathcal{C}(K,1)$ comme classe de permutations à motifs exclus dont la base est finie et connue. Dans une moindre mesure, on étend ce type de résultat au cas général de $\mathcal{C}(K,p)$, la classe des permutations obtenues en au plus p étapes de largeur au plus K à partir d'une permutation identité.

6.3.2 Permutations obtenues en p étapes de largeur au plus K

Comme dans le cas de $\mathcal{C}(K,1)$, on montre ici (théorème 6.34) que les permutations obtenues à partir d'une permutation identité après p étapes de duplication-perte de largeur au plus K forment une classe de permutations. Cependant, on n'obtient pas une description précise de la base de motifs exclus qui caractérise cette classe, mais seulement une borne supérieure sur la taille des motifs qui la composent. Comme précédemment, une "étape" sera toujours une "étape de duplication-perte de largeur au plus K", sauf mention contraire.

Pour démontrer le résultat annoncé, on définit quelques notations supplémentaires, et on prouve quelques lemmes techniques.

Définition 6.26. Le *vecteur* de i à j dans une permutation σ est composé de tous les éléments de σ dont la position est comprise (au sens large) entre i et j. La *taille* d'un vecteur est le nombre d'éléments qu'il contient.

Par exemple, le vecteur de 7 à 2 dans $4\,1\,2\,3\,5\,7\,6$ est $\overleftarrow{2\,3\,5\,7}$ et est de taille 4.

Définition 6.27. Soit σ une permutation de $\mathcal{S}_n$. Le *vecteur valeur-position associé à* $i \in [1..n]$ (ou *vp-vecteur* par souci de concision) est le vecteur de σ allant de i à σ_i, si i n'est pas un point fixe de σ. Dans le cas contraire, si $i = \sigma_i$, le *vp*-vecteur associé à i est vide.

Dans cette définition, il devrait apparaître que le *vp*-vecteur associé à i, qui va de l'élément de σ qui a pour valeur i à celui qui est à la position i, représente le déplacement nécessaire dans σ pour que i prenne sa place dans la permutation triée $12\ldots n$. Comme illustré en figure 6.23, sur la représentation graphique des permutations, le *vp*-vecteur associé à i est représenté par une flèche horizontal qui va de l'élément d'ordonnée i à la diagonale.

Une autre remarque évidente (mais d'importance) est qu'un *vp*-vecteur non vide contient au moins 2 éléments.

Afin de prendre en compte tous les déplacements nécessaires pour trier σ en $12\ldots n$, on définit le *domaine de valeurs-positions* :

Définition 6.28. Soit σ une permutation de $\mathcal{S}_n$. Le *domaine de valeurs-positions* de σ (ou *vp-domaine*) est composé de tous les éléments de σ qui apparaissent dans au moins un *vp*-vecteur.

Ces deux définitions sont illustrées sur la figure 6.23.

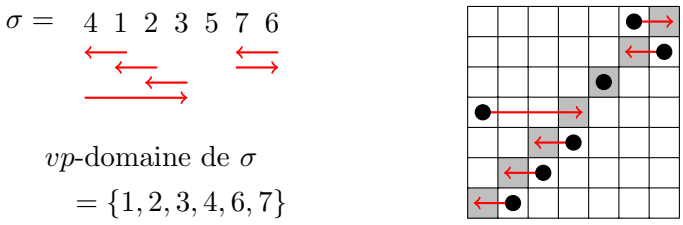


FIG. 6.23 – *vp*-vecteurs et *vp*-domaine de $\sigma = 4\,1\,2\,3\,5\,7\,6$, sur sa représentation linéaire et sa représentation graphique.

Dans un certain sens, les *vp*-vecteurs sont réversibles : si on change le sens des flèches qui leur sont associées, on obtient un ensemble de vecteurs qui représentent les mouvements nécessaires à "mélanger" $12\ldots n$ pour obtenir σ. Avec cette remarque et les définitions 6.27 et 6.28, on voit facilement que pour toute permutation $\sigma \in \mathcal{C}(K,p)$, chaque élément du *vp*-domaine de σ doit appartenir à au moins une étape dans tout scénario de duplication-perte qui mène de $12\ldots n$ à σ. Ainsi, le *vp*-domaine de σ contient au plus Kp éléments.

On rappelle que dans le contexte de cette étude, la suppression d'un élément dans une permutation est directement associée à une renormalisation, comme expliqué dans la définition 1.17. Par exemple, en supprimant 3 dans la permutation $4\,1\,2\,3\,5\,7\,6$, on obtient $3\,1\,2\,4\,6\,5$.

Lemme 6.29. *On considère une permutation σ, et la permutation τ obtenue en supprimant un élément j dans σ. Alors pour tout élément $i \neq j$ tel que $i \neq \sigma_i$, soit cet élément devient un point fixe de τ, soit la taille du vp-vecteur associé à cet élément dans τ varie de 0, 1 ou -1 par rapport à la taille du vp-vecteur associé à i dans σ.*

Démonstration. Il est facile de se convaincre de la véracité de cette affirmation en examinant la représentation graphique de σ. Les éléments (hors points fixes) de σ qui ne se trouvent pas juste au-dessus ou juste en-dessous de la diagonale ne peuvent pas devenir points fixes dans τ lors de la suppression de j. Pour les éléments qui ne sont pas points fixes dans σ et qui ne deviennent pas point fixes dans τ, la distance à la diagonale (mesurée sur l'horizontale) peut varier de 0, 1 ou -1 lors de la suppression de l'élément j. Les figures 6.24 et 6.25 montrent les différentes variations de la distance à la diagonale selon les zones de la représentation graphique de σ.

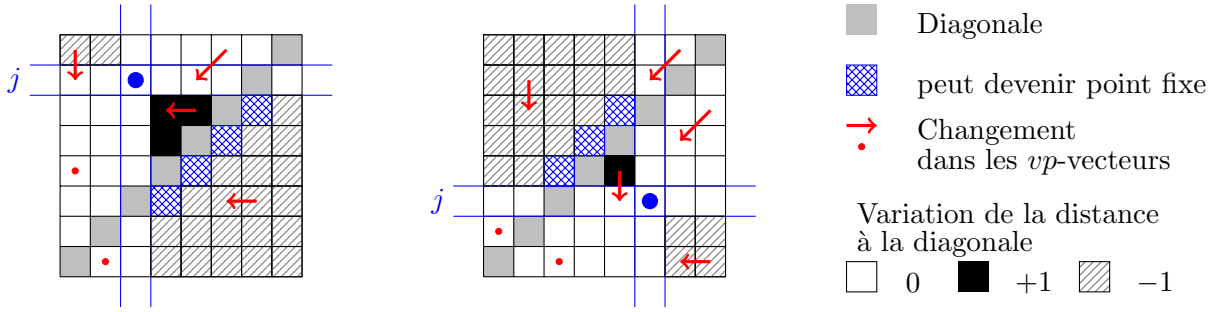


FIG. 6.24 – Cas général de la variation de la taille des *vp*-vecteurs lors de la suppression d'un élément j d'un côté ou de l'autre de la diagonale.

On peut aussi remarquer que les points fixes de σ ne restent pas forcément points fixes dans τ : il est possible que leur soit associé un *vp*-vecteur de taille 2. C'est le cas par exemple du point fixe 6 sur la figure 6.25. □

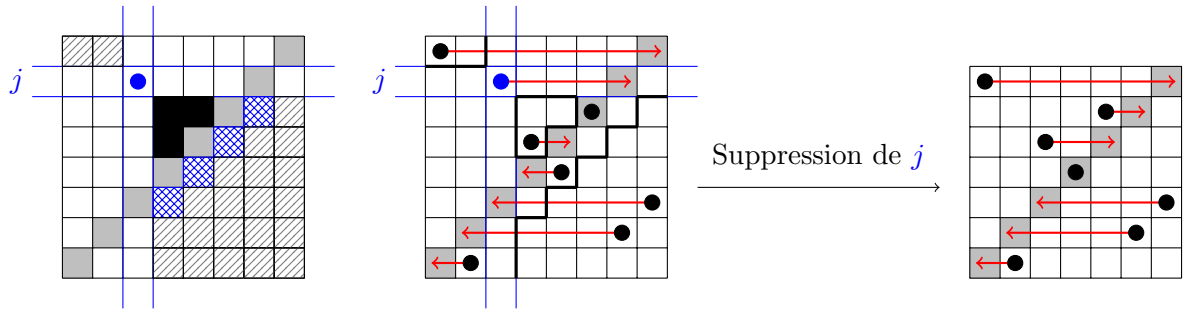


FIG. 6.25 – Variation de la taille des *vp*-vecteurs lors de la suppression d'un élément j sur un exemple.

Lemme 6.30. *Pour toute permutation σ, il existe au moins un élément j tel que la permutation τ obtenue en supprimant j dans σ contienne au plus un point fixe de plus que σ.*

Démonstration. On dira ici qu'un élément i de σ est *quasi-diagonal* lorsque $\sigma_{i-1} = i$ (dans ce cas, i est *sur-diagonal*) ou $\sigma_{i+1} = i$ (i est *sous-diagonal*). Ces deux cas correspondent respectivement aux éléments de σ situés juste au-dessus ou juste en-dessous de la diagonale dans la représentation graphique de σ. Les éléments de σ qui peuvent devenir des points fixes dans τ sont nécessairement des éléments quasi-diagonaux.

S'il n'y a pas d'élément quasi-diagonal dans σ, on peut supprimer n'importe quel élément j sans faire augmenter le nombre de points fixes. Si au contraire σ a des éléments quasi-diagonaux, on choisit j parmi eux. Alors la suppression de j provoque l'apparition d'au plus un point fixe. En effet, si $\sigma_{j-1} = j$ (l'autre cas étant similaire), alors le seul point fixe qui peut apparaître est $j-1$, dans le cas où $\sigma_j = j-1$. Cette affirmation est illustrée par la figure 6.26. □

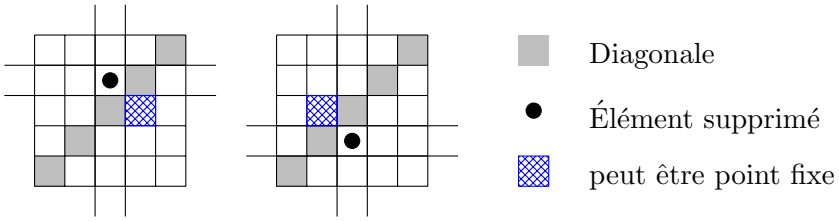


FIG. 6.26 – Le seul point fixe qui peut apparaître lors de la suppression d'un élément quasi-diagonal.

Lemme 6.31. *On considère une permutation $\sigma \notin \mathcal{C}(K,p)$ telle que pour tout motif strict τ de σ, $\tau \in \mathcal{C}(K,p)$. Alors, le vp-domaine de σ est de taille au plus $2Kp+2$.*

Démonstration. D'après le lemme 6.30, il existe $\tau \preceq \sigma$ avec $|\tau|+1 = |\sigma|$ et tel que τ a au plus un point fixe de plus que σ. On note j l'élément supprimé dans σ pour obtenir τ. Ayant fait l'hypothèse que tout motif strict de σ est dans $\mathcal{C}(K,p)$, on a en particulier $\tau \in \mathcal{C}(K,p)$. Par la remarque qui suit la définition 6.28, on sait que par conséquent, le *vp*-domaine de τ est de taille au plus Kp, donc contient au maximum Kp *vp*-vecteurs. À chacun de ces *vp*-vecteurs dans τ correspond un *vp*-vecteur dans σ (éventuellement vide) dont la taille peut rester la même, décroître, ou augmenter de 1. On va noter $\overrightarrow{V}$ l'ensemble des *vp*-vecteurs de σ obtenus ainsi à partir d'un *vp*-vecteur de τ. On obtient alors que le nombre d'éléments de σ qui appartiennent à un *vp*-vecteur de $\overrightarrow{V}$ est au plus $2Kp$. Mais $\overrightarrow{V}$ n'est pas exactement le *vp*-domaine de σ. Il doit être complété par au plus deux *vp*-vecteurs : celui associé à l'élément j qui a été supprimé, et celui associé au point fixe de τ qui n'était pas un point fixe dans σ, si un tel point existe. Si c'est le cas, alors ce point est un élément quasi-diagonal de σ et le *vp*-vecteur qui lui est associé dans σ (que l'on notera $\overrightarrow{v}$) est de taille 2, de telle sorte que l'ensemble de *vp*-vecteurs $\overrightarrow{V} \cup \{\overrightarrow{v}\}$ couvre au total au maximum $2Kp+2$ éléments. On observe alors facilement que tout élément de σ qui appartient à un *vp*-vecteur appartient nécessairement à au moins deux *vp*-vecteurs (ce qui peut être vu comme une "condition d'équilibre"). Par conséquent, tous les éléments du *vp*-vecteur associé à j sont déjà couverts par un autre *vp*-vecteur de $\overrightarrow{V} \cup \{\overrightarrow{v}\}$. Ainsi, le *vp*-domaine de σ est exactement l'ensemble des éléments couverts par $\overrightarrow{V} \cup \{\overrightarrow{v}\}$, et donc est de taille au plus $2Kp+2$. □

Lemme 6.32. *On considère une permutation $\sigma \notin \mathcal{C}(K,p)$ de taille $n > (Kp+2)^2 - 2$ telle que pour tout motif strict τ de σ, $\tau \in \mathcal{C}(K,p)$. Alors σ est de la forme $\sigma = Ii(i+1)\dots(i+Kp)J$ où I est une permutation de $[1..i-1]$ et J une permutation de $[i+Kp+1..n]$. I et J peuvent éventuellement être vides.*

Démonstration. D'après le lemme 6.31, le *vp*-domaine de σ est de taille au plus $2Kp+2$. On décompose σ en *fenêtres libres* d'éléments consécutifs extérieurs au *vp*-domaine, séparées par des fenêtres d'éléments consécutifs du *vp*-domaine (que l'on appellera *vp*-fenêtres). Cette décomposition de σ comme alternance de fenêtres libres et de *vp*-fenêtres est illustrée en figure 6.27. Les éléments du *vp*-domaine de σ (en nombre au plus $2Kp+2$) sont donc organisés en *vp*-fenêtres, de taille chacune au moins 2. Par conséquent, il y a au plus $Kp+1$ fenêtres d'éléments consécutifs du *vp*-domaine, et donc au plus $Kp+2$ fenêtres libres dans σ. En notant n la taille de σ, on a par hypothèse

$n > (Kp+2)^2 - 2 = (Kp+2)Kp + 2Kp + 2$, dont on déduit que n est strictement supérieur à la valeur : nombre de fenêtres libres de $\sigma \times Kp+$ taille du vp-domaine de σ. Comme n par définition la somme des tailles des fenêtres libres de σ à laquelle on ajoute la taille du vp-domaine de σ, cela implique qu'au moins une des fenêtres libres de σ est de taille strictement supérieure à Kp, c'est-à-dire contient au moins $Kp+1$ éléments. Par définition, ces éléments n'appartiennent pas au vp-domaine de σ et autorisent donc une décomposition de σ en $\sigma = Ii(i+1)\dots(i+Kp)J$ où I est une permutation de $[1..i-1]$ et J une permutation de $[i+Kp+1..n]$. □

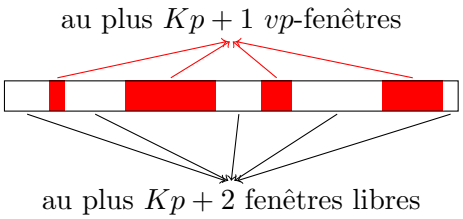


FIG. 6.27 – Décomposition d'une permutation comme alternance de fenêtres libres et de vp-fenêtres.

Ces lemmes permettent d'énoncer et de démontrer la propriété clé suivante :

Proposition 6.33. *On considère une permutation $\sigma \notin \mathcal{C}(K,p)$. Alors soit σ est de taille au plus $(Kp+2)^2-2$, soit il existe un motif strict τ de σ qui n'appartient pas à $\mathcal{C}(K,p)$.*

Démonstration. On considère une permutation $\sigma \notin \mathcal{C}(K,p)$ telle que tout motif strict τ de σ appartienne à $\mathcal{C}(K,p)$. Pour obtenir le résultat annoncé, il suffit de démontrer que σ est de taille $n \leq (Kp+2)^2-2$. On procède par l'absurde et on suppose donc le contraire. D'après le lemme 6.32, il existe $i \in [1..n]$, I une permutation de $[1..i-1]$ et J une permutation de $[i+Kp+1..n]$ tels que $\sigma = Ii(i+1)\dots(i+Kp)J$. On note $\widehat{\sigma}$ la permutation $\widehat{\sigma} = Ii(i+1)\dots(i+Kp-1)(J-1)$, où $(J-1)$ est la permutation de $[i+Kp..n-1]$ obtenue à partir de J en soustrayant 1 à chaque élément de J. La permutation $\widehat{\sigma}$ est un motif strict de σ, et ainsi $\widehat{\sigma} \in \mathcal{C}(K,p)$. Il existe donc un scénario de duplication-perte où chaque étape est de largeur au plus K, qui produit $\widehat{\sigma}$ à partir de $12\dots(n-1)$. On choisit un tel scénario avec un nombre d'étapes le plus petit possible. Ce scénario a au plus p étapes, chacune de largeur au plus K, de telle sorte que la distance totale parcourue par les éléments qui sont dupliqués est au plus Kp. Par conséquent, il est impossible de déplacer un élément de I et un élément de $J-1$ pour qu'ils soient côte à côte. Il est donc nécessaire que, dans le scénario considéré, les étapes de duplication-perte soient *internes* à I et à $J-1$. On peut reproduire ce scénario dans I et J, et exhiber ainsi un scénario de duplication-perte de $12\dots n$ à σ, contenant au plus p étapes, chacune de largeur au plus K. Ceci contredit le fait que $\sigma \notin \mathcal{C}(K,p)$ et conclut la preuve. □

Il est maintenant aisé de conclure en prouvant le théorème 6.34 :

Théorème 6.34. *La classe $\mathcal{C}(K,p)$ de toutes les permutations obtenues à partir d'une permutation identité en au plus p étapes de duplication-perte de largeur au plus K est une classe de permutations à motifs exclus dont la base est finie et contient des motifs de taille au plus $(Kp+2)^2-2$.*

Démonstration. On a vu dans le paragraphe 6.1.1 que $\mathcal{C}(K,p)$ est bien une classe de permutations. D'après le théorème 1.25 page 17, la classe de motifs exclus qui définit $\mathcal{C}(K,p)$ est $\{\pi \notin \mathcal{C}(K,p) : \forall \tau \prec \pi, \tau \in \mathcal{C}(K,p)\}$. Dans notre cas, on pose $\mathcal{B} = \{\pi \notin \mathcal{C}(K,p) : \forall \tau \prec \pi, \tau \in \mathcal{C}(K,p) \text{ et } |\pi| \leq (Kp+2)^2-2\}$ et on démontre que $\mathcal{S}(\mathcal{B}) = \mathcal{C}(K,p)$. La spécificité de ce cas est qu'on ne perd pas de généralité en imposant la borne $(Kp+2)^2-2$ sur la taille des motifs de la base.

On considère une permutation $\sigma \notin \mathcal{C}(K,p)$. Si $|\sigma| \leq (Kp+2)^2 - 2$, alors par définition de $\mathcal{B}$, soit $\sigma \in \mathcal{B}$ soit il existe un motif strict τ de σ qui n'appartient pas à $\mathcal{C}(K,p)$. Si au contraire $|\sigma| > (Kp+2)^2-2$, alors d'après la proposition 6.33, il existe un motif strict τ de σ qui n'appartient pas à $\mathcal{C}(K,p)$.

Il y a donc deux cas à analyser : $\sigma \in \mathcal{B}$, ou il existe un motif strict τ de σ qui n'appartient pas à $\mathcal{C}(K,p)$. Dans le premier cas, on obtient directement que $\sigma \notin \mathcal{S}(\mathcal{B})$. Dans le second cas, en raisonnant par récurrence sur la taille des permutations, on démontre que $\tau \notin \mathcal{S}(\mathcal{B})$, puisque $\tau \notin \mathcal{C}(K,p)$. On en déduit directement que $\sigma \notin \mathcal{S}(\mathcal{B})$. Cela démontre que $\mathcal{S}(\mathcal{B}) \subseteq \mathcal{C}(K,p)$.

Réciproquement, on considère $\sigma \in \mathcal{C}(K,p)$. Alors, tout motif τ de σ appartient aussi à $\mathcal{C}(K,p)$. Ainsi, σ ne contient aucune occurrence d'aucun des motifs de $\mathcal{B}$, c'est-à-dire que $\sigma \in \mathcal{S}(\mathcal{B})$. Cela démontre que $\mathcal{C}(K,p) \subseteq \mathcal{S}(\mathcal{B})$, terminant la preuve du théorème. □

La caractérisation des permutations de $\mathcal{C}(K,p)$ obtenue au théorème 6.34 est seulement partielle, au sens où l'on n'obtient pas de description précise et spécifique de la base de motifs exclus qui caractérise cette classe. Cependant, on obtient tout de même certaines informations sur la structure des permutations obtenues en p étapes de duplication-perte de largeur au plus K. Cela suggère donc la question suivante, qui reste ouverte : un résultat partiel comme celui du théorème 6.34 peut-il être utilisé pour réduire l'espace de recherche dans une procédure de recherche de scénarios d'évolution possibles, ou de scénarios optimaux (par exemple en devinant une borne sur le nombre d'étapes) ?

6.4 Algorithmes de calcul de scénarios, et étude de complexité

On revient dans cette partie à des considérations plus algorithmiques de calcul de scénarios de duplication-perte. On commence par décrire l'algorithme de [CCMR06] qui calcule un scénario optimal (c'est-à-dire avec un nombre d'étapes le plus petit possible) dans le modèle de duplication complète avec perte aléatoire. On se concentre ensuite sur le modèle où la duplication est restreinte à une fenêtre de largeur bornée par K, en envisageant que K puisse être une fonction $K(n)$ de la taille n de la permutation pour laquelle on cherche un scénario. On propose un algorithme calculant un scénario de duplication-perte dans ce modèle, en utilisant comme sous-procédure l'algorithme de [CCMR06]. Le scénario calculé n'est pas a priori optimal. Aussi, on évalue la qualité des scénarios calculés par l'algorithme en donnant une borne inférieure sur le nombre d'étapes nécessaires dans un scénario de duplication-perte.

6.4.1 Étude algorithmique du modèle de duplication complète avec perte aléatoire

On rappelle pour commencer l'énoncé du théorème 6.2 (page 116) :

Théorème. *Soit σ une permutation de taille n. Dans le modèle de duplication complète avec perte aléatoire, $\lceil \log_2($nombre de sous-séquences croissantes maximales de $\sigma)\rceil$ étapes de duplication-perte sont nécessaires et suffisantes pour obtenir σ à partir de $12\ldots n$.*

Dans l'étude plus combinatoire qui précède, on a privilégié la version de ce théorème reformulée en termes de descentes dans les permutations (théorème 6.8 page 117). Pour décrire les algorithmes qui suivent, la formulation avec des sous-séquences croissantes maximales est plus adaptée, mais on pourra utiliser la version en termes de descentes pour leur analyse.

La preuve du théorème 6.2 ci-dessus figure dans [CCMR06]. Sans entrer dans les détails, on se convainc assez facilement que $\lceil \log_2($nombre de sous-séquences croissantes maximales de $\sigma)\rceil$ étapes sont nécessaires, chaque étape de duplication-perte pouvant au mieux doubler le nombre de sous-séquences croissantes maximales. Pour montrer que ce nombre d'étapes est aussi suffisant, K. Chaudhuri, K. Chen, R. Mihaescu et S. Rao décrivent dans [CCMR06] un algorithme qui calcule pour toute permutation σ un scénario de duplication-perte (dans le modèle de duplication complète

avec perte aléatoire) menant de l'identité à σ, et dont le nombre d'étapes est $\lceil \log_2(\text{nombre de sous-séquences croissantes maximales de } \sigma) \rceil$.

Cet algorithme est en fait un tri par base ou tri *radix* (de l'anglais *radix sort*). Il est reproduit dans l'algorithme 6.1 ci-dessous.

Algorithme 6.1 Calcul d'un scénario optimal de duplication complète avec perte aléatoire de $12 \dots N$ vers $\sigma \in \mathcal{S}_N$.

1: $\pi = 12 \dots N$
2: Partitionner σ en sous-séquences croissantes maximales, de gauche à droite.
3: Chaque élément $[1..N]$ qui apparaît dans la i-ème sous-séquence croissante maximale reçoit comme étiquette la représentation binaire de i.
4: **pour** j allant de 1 à $\lceil \log_2(desc(\sigma) + 1) \rceil$ **faire**
5: Effectuer une étape de duplication-perte sur π qui garde dans la première copie de π les éléments dont j-ème bit le moins significatif de leur étiquette vaut 1 (et dans la seconde copie de π ceux pour lesquels la valeur de ce bit est 1).
6: **fin pour**

Afin d'examiner tous les bits des étiquettes attribuées aux éléments de $[1..N]$ par l'algorithme, le nombre de passages dans la boucle de la ligne 4 est $\lceil \log_2(\text{nombre de sous-séquences croissantes maximales de } \sigma) \rceil = \lceil \log_2(desc(\sigma) + 1) \rceil$. On en déduit que le nombre d'étapes d'un scénario de duplication complète avec perte aléatoire allant de $12 \dots n$ à $\sigma \in \mathcal{S}_n$ est $\Theta(\log n)$ dans le pire des cas. C'est aussi le cas du nombre moyen de ces étapes, comme le montre l'équation 6.1 page 148.

Dans le paragraphe 6.4.2, nous allons utiliser l'algorithme 6.1 comme sous-procédure dans un algorithme de calcul de scénarios de duplication-perte avec largeur de duplication bornée. Dans cette optique, il est intéressant de faire les quelques remarques suivantes.

Tout d'abord, l'algorithme 6.1 est décrit avec une permutation $\sigma \in \mathcal{S}_N$ en entrée, mais il s'applique également au cas plus général du calcul d'un scénario allant de $i_1 i_2 \dots i_k$ à σ, lorsque σ est une permutation de $\{i_1, i_2, \dots, i_k\}$, avec $i_1 < i_2 < \dots < i_k$.

On peut aussi remarquer que tout scénario dans notre modèle avec largeur de duplication bornée peut être vu comme un scénario dans le modèle de duplication complète avec perte aléatoire. Ainsi, le nombre d'étapes dans un scénario optimal de duplication complète avec perte aléatoire est une borne inférieure sur le nombre d'étapes d'un scénario optimal dans notre modèle. Des bornes inférieures plus fines seront décrites dans le paragraphe 6.4.3.

D'autre part, on voit facilement que le modèle de duplication complète avec perte aléatoire et notre modèle avec largeur de duplication bornée par K coïncident si on se restreint aux permutations de largeur au plus K. Nous allons tirer parti de cette remarque dans l'algorithme 6.2.

Enfin, dans le cas où $K = K(n) = n$, les deux modèles coïncident encore. Ainsi, un scénario optimal pour le modèle avec largeur de duplication bornée par K est obtenu par un tri *radix* dans ce cas. À l'extrême inverse, si $K = 2$, un scénario optimal est obtenu par un mécanisme de tri bulle. L'algorithme du paragraphe 6.4.2 calcule des scénarios dont les nombres d'étapes sont égaux aux valeurs optimales dans ces cas extrêmes. Du point de vue des algorithmes de tri, on peut donc voir le travail présenté ci-après comme une façon de décrire une famille d'algorithmes de tri entre le tri bulle et le tri *radix*.

6.4.2 Borne supérieure dans le modèle général

On décrit dans cette partie un algorithme qui calcule, étant donnée une permutation $\sigma \in \mathcal{S}_n$ en entrée, un scénario possible de duplication-perte partant de $12 \dots n$ et aboutissant à σ. Il s'agit de l'algorithme 6.2 ci-après. On se restreindra bien sûr aux étapes de duplication-perte de largeur bornée par K (constante ou fonction de n). On analysera le nombre d'étapes des scénarios calculés par l'algorithme, dans le pire des cas et en moyenne.

Algorithme 6.2 Calcul d'un scénario de duplication-perte de $12\dots n$ à $\sigma \in S_n$, avec des étapes de largeur bornée par K.

1: $\pi \leftarrow 12\dots n$
2: **pour** i allant de 1 à $\lceil \frac{n-K}{\lfloor K/2 \rfloor} \rceil$ **faire**
3: On pose $L^i = \{\sigma_j : n - i\lfloor K/2 \rfloor + 1 \leq j \leq n - (i-1)\lfloor K/2 \rfloor\}$.
4: Effectuer des étapes de duplication-perte sur π pour déplacer de la gauche vers la droite les éléments de L^i jusqu'aux positions $n - i\lfloor K/2 \rfloor + 1$ à $n - (i-1)\lfloor K/2 \rfloor$ de π, sans changer l'ordre relatif de ces éléments.
5: **fin pour**
6: **pour** i allant de 1 à $\lceil \frac{n-K}{\lfloor K/2 \rfloor} \rceil$ **faire**
7: Exécuter l'algorithme 6.1 sur la fenêtre de π entre les positions $n - i\lfloor K/2 \rfloor + 1$ et $n - (i-1)\lfloor K/2 \rfloor$.
8: **fin pour**
9: Exécuter l'algorithme 6.1 sur la fenêtre de π entre les positions 1 et $n - \lceil \frac{n-K}{\lfloor K/2 \rfloor} \rceil \lfloor K/2 \rfloor$.

Les remarques suivantes devraient aider à la compréhension de l'algorithme 6.2.

L'ensemble de valeurs L^i défini à la ligne 3 représente les $\lfloor K/2 \rfloor$ éléments les plus à droite dans σ parmi ceux qui n'ont pas encore été examinés. L'algorithme est composé de deux boucles, la première aux lignes 2 à 5, la seconde aux lignes 6 à 8.

À la fin de la première boucle (ligne 5), π est décomposée en fenêtres de largeur $\lfloor K/2 \rfloor$ (sauf la plus à gauche, qui est de largeur au plus K). Chacune de ces fenêtres contient une sous-séquence croissante dont les éléments sont ceux qui apparaissent dans la même fenêtre de positions dans σ. On peut remarquer qu'il serait possible de travailler sur des fenêtres de taille K plutôt que $\lfloor K/2 \rfloor$, mais la description et l'analyse de cette première boucle, qui aurait pour résultat une décomposition de π en sous-séquences croissantes de taille K, est alors plus complexe. En outre, cette modification de l'algorithme n'affecte pas le nombre d'étapes du scénario calculé (à un facteur multiplicatif près.)

Dans la seconde boucle, on considère de droite à gauche les fenêtres calculées par la première boucle, et comme elles sont de largeur au plus K, on appelle l'algorithme 6.1 sur chaque fenêtre pour obtenir un scénario de duplication-perte transformant π en σ.

Exemple 6.35. On donne un exemple du déroulement de l'algorithme 6.2 sur la permutation $\sigma = 2\,10\,1\,7\,6\,5\,8\,9\,3\,4$ et pour la valeur $K = 6$ du paramètre de largeur d'étape.

- D'abord, on coupe σ en fenêtres de taille $\lfloor K/2 \rfloor = 3$ et on obtient $2\,10\,1\,7 \,|\, 6\,5\,8 \,|\, 9\,3\,4$.
- Ensuite, la première boucle de l'algorithme part de $1\,2\,3\,4\,5\,6\,7\,8\,9\,10$ et transporte les éléments (toujours en ordre croissant) dans la fenêtre qui leur correspond dans σ. On obtient $1\,2\,7\,10 \,|\, 5\,6\,8 \,|\, 3\,4\,9$ en sortie de cette boucle.
- Enfin, la seconde boucle trie chaque fenêtre séparément en utilisant le tri *radix* de l'algorithme 6.2 pour obtenir σ.

On peut remarquer que dans la seconde boucle (sauf pour la fenêtre la plus à gauche), on utilise seulement des étapes de duplication-perte de largeur au plus $\lfloor K/2 \rfloor$. Une amélioration possible de l'algorithme que nous avons envisagée consiste à appliquer l'algorithme 6.2 sur des fenêtres de largeur K, qui sont malgré tout des sous-séquences croissantes. L'analyse de cette variante de l'algorithme donne les mêmes bornes théoriques sur le nombre d'étapes des scénarios calculés. Cependant, on remarque sur des simulations que le nombre d'étapes produit par cet algorithme alternatif est strictement inférieur à celui fournit par l'algorithme original, sans qu'on n'ait de preuve pour appuyer ce constat.

On passe maintenant à l'analyse du nombre d'étapes dans un scénario calculé par l'algorithme 6.2.

Proposition 6.36. *Le nombre d'étapes de duplication-perte d'un scénario calculé par l'algorithme 6.2 sur une permutation de taille n est dans le pire des cas $\Theta(\frac{n}{K}\log K+\frac{n^2}{K^2})$ asymptotiquement.*

Démonstration. On se place à la i-ème itération de la première boucle. On doit déplacer les $\lfloor K/2\rfloor$ éléments de L^i vers leurs positions (qui vont de $n-i\lfloor K/2\rfloor+1$ à $n-(i-1)\lfloor K/2\rfloor$) par des étapes de duplication-perte de largeur au plus K. La pire situation se présente lorsque les éléments de L^i sont tous au début de π. Même dans ce cas, on peut déplacer les éléments de L^i de $\lceil K/2\rceil$ positions vers la droite à chaque étape (puisque $\lfloor K/2\rfloor + \lceil K/2\rceil = K$) jusqu'à ce qu'ils atteignent leur position. Le nombre total d'étapes de duplication-perte dans cette première partie du scénario calculé est alors au plus

$$\sum_{i=1}^{\lceil \frac{n-K}{\lfloor K/2\rfloor}\rceil} \left\lceil \frac{n-i\lfloor K/2\rfloor}{\lceil K/2\rceil}\right\rceil = \Theta(\frac{n^2}{K^2}).$$

On s'intéresse maintenant à la seconde boucle de l'algorithme 6.2. D'après l'analyse de l'algorithme 6.1, dans chaque fenêtre de taille $\lfloor K/2\rfloor$, le scénario produit a au plus $\lceil\log\lfloor K/2\rfloor\rceil$ étapes (ligne 7), et dans la fenêtre la plus à gauche (ligne 7), il en a au plus $\lceil\log K\rceil$. Ainsi, le nombre d'étapes de duplication-perte produites par la deuxième boucle est

$$\left\lceil \frac{n-K}{\lfloor K/2\rfloor}\right\rceil \lceil\log\lfloor K/2\rfloor\rceil + \lceil\log K\rceil = \Theta(\frac{n}{K}\log K).$$

Ce qui précède justifie que le nombre total d'étapes de duplication-perte dans un scénario calculé par l'algorithme 6.2 est au plus $\mathcal{O}(\frac{n}{K}\log K + \frac{n^2}{K^2})$ asymptotiquement, dans le pire des cas.

On voit aussi facilement, en suivant les mêmes étapes de raisonnement, que dans le cas de la permutation miroir de l'identité $n(n-1)\dots 21$, le nombre d'étapes dans le scénario calculé par l'algorithme 6.2 atteint l'ordre de $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2})$.

On en conclut donc que le nombre total d'étapes de duplication-perte dans un scénario calculé par l'algorithme 6.2 est dans le pire des cas de l'ordre de $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2})$ asymptotiquement.

On remarque au passage que le pire des cas correspond donc à la permutation miroir de l'identité $n(n-1)\dots 21$, ce qui rejoint l'intuition de ce qu'est un mauvais cas dans ce contexte. □

On peut aussi remarquer que $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2}) = \Theta(\frac{n^2}{K^2})$ pour des "petites" valeurs de K, plus précisément tant que $K = o(\frac{n}{\log n})$. Si au contraire $\frac{n}{\log n} = o(K)$ alors $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2}) = \Theta(\frac{n}{K}\log K)$. Quand $K = \Theta(\frac{n}{\log n})$, les deux termes sont du même ordre de grandeur.

On peut aussi calculer la valeur moyenne du nombre d'étapes de duplication-perte dans un scénario calculé par l'algorithme 6.2.

Proposition 6.37. *Le nombre d'étapes de duplication-perte dans un scénario calculé par l'algorithme 6.2 sur une permutation de taille n est en moyenne $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2})$ asymptotiquement.*

Démonstration. On commence par définir quelques notations. On considère une permutation σ de taille n, et on la décompose de droite à gauche en $p = \left\lceil \frac{n-K}{\lfloor K/2\rfloor}\right\rceil + 1$ fenêtres, toutes de taille $\lfloor K/2\rfloor$, sauf la plus à gauche dont la largeur est $n - \left\lceil \frac{n-K}{\lfloor K/2\rfloor}\right\rceil \lfloor K/2\rfloor \leq K$. On note $\sigma = \sigma^1\sigma^2\dots\sigma^p$ cette décomposition.

Nombre d'étapes de duplication-perte produites par la première boucle de l'algorithme 6.2.

On note $c(\sigma)$ le nombre d'étapes de duplication-perte du scénario produit par la première boucle de l'algorithme 6.2 sur l'entrée σ. En particulier, on note aussi $c_p(\sigma)$ le nombre d'étapes produites par la première itération de cette boucle, c'est-à-dire le nombre d'étapes pour déplacer

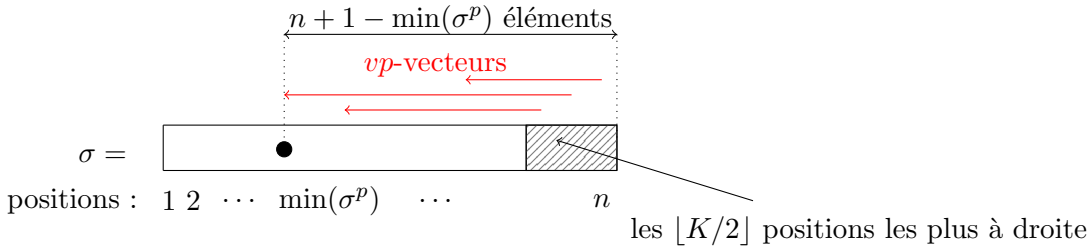


FIG. 6.28 – Trouver des bornes sur $c_p(\sigma)$.

les éléments de L^1 à la fin de la permutation. Pour calculer le nombre moyen de ces étapes, on pose $u_n = \sum_{\sigma \in S_n} c(\sigma)$. On se convainc facilement que

$$\begin{aligned} u_n &= \sum_{\sigma \in S_n} c_p(\sigma) + c(\sigma^1 \dots \sigma^{p-1}) \\ &= \sum_{\sigma \in S_n} c_p(\sigma) + n(n-1)\dots(n-\lfloor K/2 \rfloor + 1) \sum_{\sigma \in S_{n-\lfloor K/2 \rfloor}} c(\sigma) \\ &= \sum_{\sigma \in S_n} c_p(\sigma) + \frac{n!}{(n-\lfloor K/2 \rfloor)!} u_{n-\lfloor K/2 \rfloor}. \end{aligned}$$

On concentre nos efforts sur $\sum_{\sigma \in S_n} c_p(\sigma)$. La figure 6.28 rend l'inégalité suivante évidente :

$$\frac{n+1-\lfloor K/2 \rfloor - \min(\sigma^p)}{K} \leq c_p(\sigma) \leq \frac{n+1-\lfloor K/2 \rfloor - \min(\sigma^p)}{\lfloor K/2 \rfloor}.$$

On vérifie ensuite facilement que le nombre de permutations σ de taille n telles que $\min(\sigma^p) = i$ est $\binom{n-i}{\lfloor K/2 \rfloor - 1} (n - \lfloor K/2 \rfloor)! \lfloor K/2 \rfloor !$. Cela donne alors

$$\begin{aligned} \sum_{\sigma \in S_n} n+1-\lfloor K/2 \rfloor - \min(\sigma^p) &= \sum_{i=1}^{n-\lfloor K/2 \rfloor + 1} (n+1-\lfloor K/2 \rfloor - i) \binom{n-i}{\lfloor K/2 \rfloor - 1} (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \\ &= (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \sum_{i=\lfloor K/2 \rfloor - 1}^{n-1} (i+1-\lfloor K/2 \rfloor) \binom{i}{\lfloor K/2 \rfloor - 1} \\ &= (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \lfloor K/2 \rfloor \sum_{i=\lfloor K/2 \rfloor}^{n-1} \binom{i}{\lfloor K/2 \rfloor} \\ &= (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \lfloor K/2 \rfloor \binom{n}{\lfloor K/2 \rfloor + 1}. \end{aligned}$$

Par conséquent,

$$\begin{aligned} \sum_{\sigma \in S_n} c_p(\sigma) &\leq (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \binom{n}{\lfloor K/2 \rfloor + 1} \\ \sum_{\sigma \in S_n} c_p(\sigma) &\geq \frac{\lfloor K/2 \rfloor}{K} (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \binom{n}{\lfloor K/2 \rfloor + 1} \geq \frac{1}{3} (n-\lfloor K/2 \rfloor)! \lfloor K/2 \rfloor ! \binom{n}{\lfloor K/2 \rfloor + 1}, \end{aligned}$$

ce qui donne après quelques calculs

$$\frac{1}{3} \frac{n-\lfloor K/2 \rfloor}{\lfloor K/2 \rfloor + 1} + \frac{u_{n-\lfloor K/2 \rfloor}}{(n-\lfloor K/2 \rfloor)!} \leq \frac{u_n}{n!} \leq \frac{n-\lfloor K/2 \rfloor}{\lfloor K/2 \rfloor + 1} + \frac{u_{n-\lfloor K/2 \rfloor}}{(n-\lfloor K/2 \rfloor)!}.$$

Il devient donc naturel de considérer deux suites (v_n) et (w_n) vérifiant les relations de récurrence $v_n = \frac{1}{3}\frac{n-\lfloor K/2\rfloor}{\lfloor K/2\rfloor+1} + v_{n-\lfloor K/2\rfloor}$ et $w_n = \frac{n-\lfloor K/2\rfloor}{\lfloor K/2\rfloor+1} + w_{n-\lfloor K/2\rfloor}$ respectivement si $n > K$, et $v_n = w_n = \frac{u_n}{n!}$ pour tout $n \leq K$. On a alors $v_n \leq \frac{u_n}{n!} \leq w_n \forall n \in \mathbb{N}$. On peut résoudre les équations de récurrence définissant v_n et w_n, et en notant $n = \lceil \frac{n-K}{\lfloor K/2\rfloor} \rceil \lfloor K/2\rfloor + r$ (ce qui induit $\lfloor K/2\rfloor \leq r \leq K$), on obtient :

$$v_n = \frac{1}{3}\sum_{i=1}^{\lceil \frac{n-K}{\lfloor K/2\rfloor}\rceil} \frac{n - i\lfloor K/2\rfloor}{\lfloor K/2\rfloor+1} + v_r = \frac{1}{3(\lfloor K/2\rfloor+1)}\Big\lceil \frac{n-K}{\lfloor K/2\rfloor}\Big\rceil \Big(n - \lfloor K/2\rfloor \frac{\lceil \frac{n-K}{\lfloor K/2\rfloor}\rceil + 1}{2}\Big) + v_r = \Theta(\frac{n^2}{K^2})$$

et

$$w_n = \sum_{i=1}^{\lceil \frac{n-K}{\lfloor K/2\rfloor}\rceil} \frac{n - i\lfloor K/2\rfloor}{\lfloor K/2\rfloor+1} + w_r = \Theta(\frac{n^2}{K^2}).$$

On déduit donc que le nombre moyen d'étapes de duplication-perte produites par la première boucle de l'algorithme 6.2 sur les permutations de taille n est $\frac{u_n}{n!} = \Theta(\frac{n^2}{K^2})$.

Nombre d'étapes de duplication-perte produites par la seconde boucle de l'algorithme 6.2.

Il reste à calculer le nombre moyen d'étapes de duplication-perte du scénario qui sont produites par la deuxième boucle de l'algorithme 6.2 sur les permutations de taille n. Ce nombre est donné par

$$\begin{aligned}
\frac{1}{n!}\sum_{\sigma\in S_n}\sum_{i=1}^{p}\lceil \log(desc(\sigma^i)+1)\rceil &= \frac{1}{n!}\Big(\sum_{i=2}^{p}\sum_{\sigma\in S_n}\lceil \log(desc(\sigma^i)+1)\rceil + \sum_{\sigma\in S_n}\lceil \log(desc(\sigma^1)+1)\rceil\Big)\\
&= \frac{1}{n!}\Big(\sum_{i=2}^{p}(n-\lfloor K/2\rfloor)!\binom{n}{\lfloor K/2\rfloor}\sum_{\sigma\in S_{\lfloor K/2\rfloor}}\lceil \log(desc(\sigma)+1)\rceil\\
&\quad +(n-|\sigma^1|)!\binom{n}{|\sigma^1|}\sum_{\sigma\in S_{|\sigma^1|}}\lceil \log(desc(\sigma)+1)\rceil\Big)\\
&= \frac{1}{\lfloor K/2\rfloor!}(p-1)\sum_{\sigma\in S_{\lfloor K/2\rfloor}}\lceil \log(desc(\sigma)+1)\rceil\\
&\quad +\frac{1}{|\sigma^1|!}\sum_{\sigma\in S_{|\sigma^1|}}\lceil \log(desc(\sigma)+1)\rceil.
\end{aligned}$$

Comme $p = \left\lceil \frac{n-K}{\lfloor K/2\rfloor}\right\rceil + 1$, on en déduit que le nombre moyen d'étapes de duplication-perte produites par la seconde boucle de l'algorithme 6.2 sur les permutations de taille n est

$$\Theta(\frac{1}{\lfloor K/2\rfloor!}\Big\lceil \frac{n-K}{\lfloor K/2\rfloor}\Big\rceil \sum_{\sigma\in S_{\lfloor K/2\rfloor}}\lceil \log(desc(\sigma)+1)\rceil).$$

Il reste donc à trouver une estimation asymptotique de $\frac{1}{k!}\sum_{\sigma\in S_k}\lceil \log(desc(\sigma)+1)\rceil)$ pour $k = \lfloor K/2\rfloor$. La fonction log étant concave, et sachant par la remarque 6.6 que $\frac{1}{k!}\sum_{\sigma\in S_k} desc(\sigma)+1 = \frac{k+1}{2}$, on obtient

$$\frac{1}{k!}\sum_{\sigma\in S_k}\lceil \log(desc(\sigma)+1)\rceil \geq \frac{1}{k!}\sum_{\sigma\in S_k}\log(desc(\sigma)+1) \geq \log(\frac{k+1}{2}).$$

En outre, il est clair que

$$\frac{1}{k!}\sum_{\sigma\in S_k}\lceil \log(desc(\sigma)+1)\rceil \leq \lceil \log(k)\rceil,$$

et on peut en déduire que

$$\frac{1}{k!}\sum_{\sigma\in S_k}\lceil\log(desc(\sigma)+1)\rceil=\Theta(\log(k)). \tag{6.1}$$

En conséquence, le nombre moyen d'étapes de duplication-perte produites par la seconde boucle de l'algorithme 6.2 sur les permutations de taille n est $\Theta(\lceil\frac{n-K}{\lfloor K/2\rfloor}\rceil\log(\lfloor K/2\rfloor))=\Theta(\frac{n}{K}\log K)$.

En conclusion, le nombre total d'étapes de duplication-perte dans un scénario calculé par l'algorithme 6.2 est en moyenne $\Theta(\frac{n}{K}\log K+\frac{n^2}{K^2})$ sur les permutations de taille n. □

6.4.3 Borne inférieure dans le modèle général

Les algorithmes présentés dans le paragraphe 6.4.2 ci-dessus permettent de donner des bornes supérieures sur le nombre d'étapes (dans le pire des cas et en moyenne) dans un scénario optimal de duplication-perte produisant une permutation σ de taille n à partir de l'identité de même taille, dans le modèle où la largeur de duplication est bornée par un paramètre K (qui peut être fonction de n). On s'intéresse maintenant à décrire des bornes inférieures sur ce nombre d'étapes. Nous verrons que les bornes inférieures décrites coïncident avec les bornes supérieures fournies par l'algorithme 6.2, à un facteur multiplicatif près, pour de nombreuses valeurs de K : c'est plus précisément le cas lorsque K est une constante, ou lorsque K est une fonction de n qui ne vérifie pas que $\frac{n}{\log n}\ll K(n)\ll n$. On discutera aussi de la valeur du facteur multiplicatif qui sépare la borne inférieure de la borne supérieure.

Pour ce calcul de bornes inférieures, on utilise un paramètre bien connu sur les permutations : leur nombre d'inversions. On en rappelle la définition ainsi que quelques propriétés. Pour plus de détails, on pourra consulter [Com74] par exemple.

Définition 6.38. On considère une permutation $\sigma\in\mathcal{S}_n$, et deux positions $i<j\in[1..n]$. On dit qu'il y a une *inversion* entre σ_i et σ_j si $\sigma_i>\sigma_j$.

Exemple 6.39. Par exemple, les inversions de $\sigma=3\,1\,2\,4\,6\,5$ sont obtenues pour les paires $(\sigma_i,\sigma_j)\in\{(3,1),(3,2),(6,5)\}$.

Remarque 6.40. La valeur maximale du nombre d'inversions dans une permutation de taille n est $\frac{n(n-1)}{2}$, ce qui correspond à la permutation $n(n-1)\dots1$. Le nombre moyen d'inversions parmi les permutations de taille n est $\frac{n(n-1)}{4}$.

La preuve de cette remarque, très similaire à celle de la remarque 6.6 concernant le nombre moyen de descentes, est omise.

Proposition 6.41. *Dans le pire des cas, $\Omega(\log n+\frac{n^2}{K^2})$ étapes de duplication-perte de largeur K sont nécessaires dans un scénario donnant une permutation de $\mathcal{S}_n$ à partir de $123\dots n$.*

Démonstration. On examine d'abord le nombre d'inversions que peut créer une étape de duplication-perte s de largeur K. Il est clair que les inversions créées ne peuvent qu'impliquer deux éléments de s. On note i le nombre d'éléments de s qui sont conservés dans la première copie du fragment dupliqué. Alors s crée au maximum $i(K-i)\leq\frac{K^2}{4}$ inversions dans la permutation. Une permutation de σ taille n pouvant avoir jusqu'à $\frac{n(n-1)}{2}$ inversions, une borne inférieure sur le nombre d'étapes nécessaires pour transformer $123\dots n$ en σ est $\frac{2n(n-1)}{K^2}$.

Pour l'autre terme de la borne inférieure, on hérite du résultat de [CCMR06], énonçant que $\log n$ étapes sont nécessaires dans un scénario pour le modèle de duplication complète avec perte aléatoire, où les étapes de duplication-perte sont moins contraintes.

On obtient finalement la borne inférieure $\Omega(\log n+\frac{n^2}{K^2})$ sur le nombre d'étapes nécessaires dans un scénario de duplication-perte avec largeur de duplication bornée par K pour obtenir une permutation de taille n à partir de $123\dots n$ dans le pire des cas. □

Proposition 6.42. *En moyenne, dans un scénario de duplication-perte avec largeur de duplication bornée par K allant de $123\dots n$ à $\sigma \in \mathcal{S}_n$, $\Omega(\log n + \frac{n^2}{K^2})$ étapes sont nécessaires.*

Démonstration. Comme précédemment, une étape de duplication-perte peut créer au plus $\frac{K^2}{4}$ inversions dans une permutation. Le nombre moyen d'inversions dans une permutation de taille n étant $\frac{n(n-1)}{4}$, on en déduit qu'en moyenne au moins $\frac{n(n-1)}{K^2}$ étapes de duplication-perte sont nécessaires pour passer de $123\dots n$ à $\sigma \in \mathcal{S}_n$.

La partie logarithmique de la borne nous est encore fournie par [CCMR06] : la borne inférieure relative au modèle de duplication complète avec perte aléatoire, qui autorise plus d'opérations que notre modèle, s'applique donc dans notre contexte.

En conclusion, on obtient la borne annoncée $\Omega(\log n + \frac{n^2}{K^2})$ sur le nombre d'étapes nécessaires en moyenne pour passer de $123\dots n$ à $\sigma \in \mathcal{S}_n$. □

On propose pour terminer une comparaison des bornes supérieures et inférieures obtenues aux paragraphes 6.4.2 et 6.4.3 respectivement. En général, il n'y a pas égalité entre ces bornes, mais nous allons voir qu'elles coïncident en dehors du cas où $K = K(n)$ vérifie $\frac{n}{\log n} \ll K \ll n$.

En effet, lorsque $K = o(\frac{n}{\log n})$, on a $\frac{n}{K}\log K = o(\frac{n^2}{K^2})$, et par conséquent une autre écriture de la borne supérieure est $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2}) = \Theta(\frac{n^2}{K^2})$, ce qui correspond à un facteur multiplicatif près à la borne inférieure $\Omega(\log n + \frac{n^2}{K^2}) = \Omega(\frac{n^2}{K^2})$. Une relecture attentive des preuves des propositions 6.36 à 6.42 montre que ce facteur multiplicatif vaut 1 dans l'analyse au pire des cas, et 2 dans l'analyse en moyenne.

Lorsque $K = \Theta(\frac{n}{\log n})$, on obtient de la même manière que les bornes inférieures et supérieures sont du même ordre de grandeur, mais la valeur du facteur multiplicatif n'est pas aussi immédiate à calculer.

Enfin, si $K = \Theta(n)$, alors $\Theta(\frac{n}{K}\log K + \frac{n^2}{K^2}) = \Theta(\log n)$ et $\Omega(\log n + \frac{n^2}{K^2}) = \Omega(\log n)$, et ainsi les bornes inférieures et supérieures sont dans ce cas aussi du même ordre de grandeur. En notant $K \sim \frac{n}{C}$, on déduit des propositions 6.36 à 6.42 que le facteur constant qui sépare la borne inférieure de la borne supérieure est $2C$, aussi bien pour le pire des cas que pour le cas moyen.

À l'inverse, lorsque $\frac{n}{\log n} \ll K \ll n$, les bornes supérieures et inférieures ne sont pas du même ordre de grandeur. Une question ouverte est donc naturellement de trouver, lorsque $\frac{n}{\log n} \ll K \ll n$, l'ordre de grandeur du nombre d'étapes d'un scénario de duplication-perte optimal dans le modèle avec largeur de duplication bornée par K (pour le pire des cas et pour le cas moyen), et un algorithme de calcul de ces scénarios optimaux.

Enfin, l'algorithme 6.2 ne calcule pas a priori, étant donnée une permutation σ, un scénario de duplication-perte optimal : les résultats "d'optimalité" des scénarios produits sont faibles, car ils ne s'appliquent qu'au cas moyen et au pire des cas, et à un facteur multiplicatif près. Un élargissement possible de ce travail pourrait donc consister à trouver des algorithmes de calcul de scénarios optimaux. Au delà de l'étude des modèles de duplication-perte, décrire de tels algorithmes serait un moyen de décrire une famille d'algorithmes de tri (paramétrée par $K = K(n)$) qui offrirait un continuum entre le tri bulle (pour $K = 2$) et le tri *radix* (pour $K(n) = n$).

Chapitre 7

Tri par renversement de permutations signées

Sommaire

7.1 Généralités sur le tri par renversements des permutations 152
7.2 Tri parfait par renversements des permutations signées 154
7.2.1 Algorithme de tri parfait par renversements 154
7.2.2 Analyse de complexité en moyenne 155
7.3 Le cas particulier des permutations séparables 158
7.3.1 Nombre moyen de renversements 158
7.3.2 Taille moyenne d'un renversement 161

7.1 Généralités sur le tri par renversements des permutations

Dans ce chapitre, on envisage un autre modèle d'évolution de génomes, représentés par des permutations, où les opérations autorisées sont des renversements de fragments continus de la permutation. Il existe en réalité de nombreuses variantes de ce modèle, et on s'intéressera principalement au problème du tri *parfait* par renversements de permutations *signées*. Avant de présenter cette étude, on propose un bref survol des résultats essentiels concernant les variantes les plus étudiées du problème de tri par renversements de permutations. Pour plus de détails, on renvoie à l'état de l'art réalisé dans [Ber05].

Définition 7.1. Un *renversement* dans une permutation σ, entre les positions i et $j \geq i$, consiste à "couper" le fragment de σ allant de la position i à la position j (incluses) dans σ et à "retourner" ce fragment. Plus formellement, le résultat du renversement entre les positions i et j dans $\sigma = \sigma_1\sigma_2\cdots\sigma_i\sigma_{i+1}\cdots\sigma_{j-1}\sigma_j\cdots\sigma_n$ est la permutation $\sigma_1\sigma_2\cdots\sigma_j\sigma_{j-1}\cdots\sigma_{i+1}\sigma_i\cdots\sigma_n$.

Exemple 7.2. Dans la permutation $\sigma = 2\,1\,7\,5\,3\,8\,4\,6$, le renversement entre les positions 3 et 6 aboutit à la permutation $2\,1\,\overleftrightarrow{8\,3\,5\,7}\,4\,6$.

Définition 7.3. Le problème du *tri par renversements* d'une permutation σ consiste à trouver un scénario de renversements parcimonieux transformant σ en la permutation identité de même taille, c'est-à-dire une suite de renversements de longueur minimale (la longueur étant mesurée en nombre de renversements) qui transforme $\sigma \in \mathcal{S}_n$ en $12\ldots n$.

Il faut bien remarquer que contrairement au cas du modèle de duplication en tandem avec perte aléatoire étudié au chapitre précédent, on cherche ici un scénario *partant de* σ et *arrivant à* une permutation identité.

Le problème du tri par renversements de permutations (non signées) est malheureusement NP-difficile [Ber05]. Cependant, la version du problème sur des permutations signées est polynomiale.

Définition 7.4. Une permutation *signée* est une permutation dans laquelle chaque élément reçoit un signe $+$ ou $-$.

Par convention, on notera une permutation signée avec une barre sur les éléments de signe $-$ (et rien pour les éléments de signe $+$).

Exemple 7.5. La permutation signée $+3\,-4\,-2\,+5\,-8\,+1\,-6\,+7$ est aussi notée $3\,\overline{4}\,\overline{2}\,5\,\overline{8}\,1\,\overline{6}\,7$.

Cette variante signée du problème de tri par renversements prend tout son sens avec les motivations biologiques sous-jacentes d'évolution de génomes. En effet, biologiquement, les gènes ont un "sens", et un renversement change le sens des gènes renversés. De la même manière, un renversement change le signe des éléments renversés, qui codent le sens des gènes associés :

Définition 7.6. Un *renversement* dans une permutation signée σ, entre les positions i et $j \geq i$, consiste à "couper" le fragment de σ allant de la position i à la position j (incluses) dans σ et à "retourner" ce fragment, en changeant les signes des éléments du fragment. Plus formellement, avec l'identité $\overline{\overline{x}} = x$, le résultat du renversement entre les positions i et j dans $\sigma = \sigma_1\sigma_2\cdots\sigma_i\sigma_{i+1}\cdots\sigma_{j-1}\sigma_j\cdots\sigma_n$ est la permutation $\sigma_1\sigma_2\cdots\overline{\sigma_j}\,\overline{\sigma_{j-1}}\cdots\overline{\sigma_{i+1}}\,\overline{\sigma_i}\cdots\sigma_n$.

On appellera *taille* d'un renversement le nombre d'éléments de la permutation dans le fragment renversé.

Définition 7.7. Le problème du *tri par renversements* d'une permutation signée σ consiste à trouver un scénario de renversements parcimonieux transformant σ en la permutation identité de même taille $12\ldots n$ ou en $\overline{n}\ldots\overline{2}\,\overline{1}$, c'est-à-dire une suite de renversements de longueur minimale (la longueur étant mesurée en nombre de renversements) qui transforme $\sigma \in \mathcal{S}_n$ en $12\ldots n$ ou en $\overline{n}\ldots\overline{2}\,\overline{1}$.

Exemple 7.8. Un scénario parcimonieux pour $\sigma = 4\,\overline{1}\,6\,7\,\overline{3}\,\overline{2}\,5$ est le suivant (les parties de la permutation entre | indiquent les fragments renversés) :

$$\begin{aligned}\sigma = |\,4\,\overline{1}\,6\,7\,|\,\overline{3}\,\overline{2}\,5 &\rightsquigarrow \overline{7}\,\overline{6}\,|\,1\,\overline{4}\,\overline{3}\,\overline{2}\,5\,|\\ &\rightsquigarrow \overline{7}\,\overline{6}\,\overline{5}\,|\,2\,3\,4\,|\,\overline{1}\\ &\rightsquigarrow \overline{7}\,\overline{6}\,\overline{5}\,\overline{4}\,\overline{3}\,\overline{2}\,\overline{1}\end{aligned}$$

Comme annoncé plus haut, le problème du tri par renversements peut être résolu en temps polynomial. Le premier algorithme polynomial de calcul de scénarios parcimonieux pour les permutations signées a été décrit en 1995 par S. Hannenhalli et P. A. Pevzner [HP99]. Depuis, d'autres algorithmes ont été développés [BMS05, TBS07], et sont regroupés dans ce qu'on appelle désormais la théorie de Hannenhalli et Pevzner. Pour une introduction à cette théorie, on pourra consulter l'article d'Anne Bergeron [Ber05].

Un raffinement de ce problème a été récemment introduit, pour prendre un compte un peu mieux la réalité biologique des phénomènes mis en jeu [FV04]. En comparant les génomes deux espèces, on s'aperçoit que certains ensembles de gènes homologues (c'est-à-dire ayant un même ancêtre commun) apparaissent regroupés dans les deux espèces. On considère alors que ces groupes de gènes ont toutes les chances d'apparaître regroupés dans le génome de l'ancêtre commun, et de ne jamais être séparés dans le processus d'évolution.

Dans notre modélisation des génomes par des permutations signées (l'une des deux permutations étant toujours l'identité, modulo réétiquetage), ces groupes de gènes sont représentés par les *intervalles* des permutations signées.

Définition 7.9. Un *intervalle* dans une permutation signée σ est un ensemble d'éléments de σ dont les positions et les *valeurs absolues* des valeurs forment deux intervalles de $\mathbb{N}$.
Un intervalle sera identifié par les valeurs de ses éléments, en valeur absolue.

Exemple 7.10. Parmi les intervalles de $3\,\overline{4}\,\overline{2}\,5\,\overline{8}\,1\,\overline{6}\,7$, on peut citer $\{3,4\}$, $\{2,3,4,5\}$ ou $\{6,7\}$.

Parmi tous les scénarios de tri par renversements, on va donc envisager seulement ceux qui ne coupent jamais aucun intervalle :

Définition 7.11. Un scénario *parfait* de tri par renversements d'une permutation signée σ est un scénario de renversements transformant σ en la permutation identité de même taille $12\ldots n$ ou en $\overline{n}\ldots\overline{2}\,\overline{1}$ qui ne casse à aucun moment un intervalle de σ.

L'existence d'un scénario parfait pour une permutation σ est une question naturelle : il a été démontré dans [FV04] que toute permutation admet un scénario parfait de tri par renversements. Le problème étudié dans ce chapitre est celui du calcul, étant donnée une permutation σ, d'un scénario parfait de tri par renversements pour σ, qui soit parcimonieux, c'est-à-dire le plus court possible (en nombre de renversements) parmi tous les scénarios parfaits. Il faut remarquer qu'il est possible qu'un scénario parfait parcimonieux soit plus long qu'un scénario parcimonieux (sans contrainte de perfection).

Il a été démontré dans [FV04] que le calcul de scénario de tri parfait par renversements parcimonieux est un problème NP-difficile, mais des algorithmes FPT ont été proposés.

Les algorithmes FPT (de l'anglais *fixed parameter tractable*) ont une complexité dépendant d'un paramètre, de telle façon qu'ils deviennent polynomiaux lorsqu'on restreint le problème aux instances correspondant à une valeur fixée du paramètre. Pour illustrer cette méthode, on présentera l'algorithme de [BBCP07] dans le paragraphe 7.2.1, et on montrera qu'il est en fait polynomial en moyenne.

Une autre façon d'aborder l'étude de ce problème NP-difficile est de se restreindre à des classes de permutations sur lesquelles le problème est polynomial. Toujours suivant [BBCP07], on se restreindra à la classe des permutations séparables (dites aussi commutantes dans ce contexte), et on analysera les scénarios produits par l'algorithme, en particulier le nombre moyen de renversements et la taille moyenne d'un renversement dans ces scénarios.

7.2 Tri parfait par renversements des permutations signées

7.2.1 Algorithme de tri parfait par renversements

On décrit ici l'algorithme de S. Bérard, A. Bergeron, C. Chauve et C. Paul tiré de [BBCP07]. Dans cet algorithme, c'est l'arbre de décomposition des permutations (enrichi de signes) qui sert de guide au calcul de scénarios parfaits de tri par renversements.

Définition 7.12. L'*arbre signé* d'une permutation signée σ est l'arbre de décomposition de la permutation non signée sous-jacente à σ où certains nœuds reçoivent un signe + ou − de la manière suivante :

- une feuille reçoit le signe + (resp. −) lorsqu'elle correspond à un élément de signe + (resp. −) dans σ,
- un nœud interne linéaire d'étiquette $\oplus$ (resp. $\ominus$) reçoit le signe + (resp. −),
- un nœud interne premier reçoit l'étiquette de son père, si celui-ci existe et est un nœud linéaire.

Il faut donc remarquer que certains nœuds dans l'arbre signé d'une permutation n'ont pas de signe : c'est le cas des nœuds premier dont le père et premier, et de la racine lorsque le nœud racine est premier.

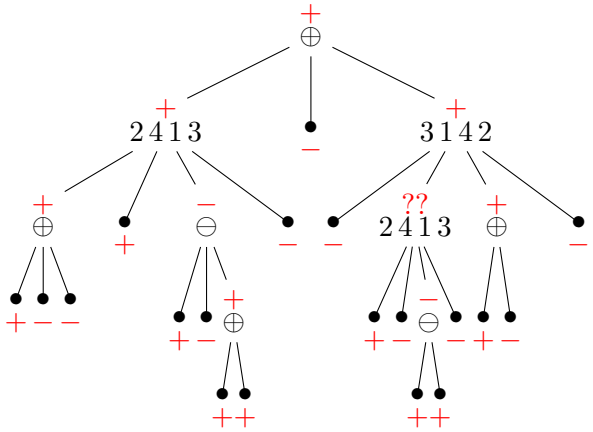


FIG. 7.1 – Arbre signé de la permutation $\sigma = 5\,\overline{6}\,\overline{7}\,9\,4\,\overline{3}\,1\,2\,\overline{8}\,\overline{10}\,\overline{17}\,13\,\overline{15}\,12\,11\,\overline{14}\,18\,\overline{19}\,\overline{16}$.

Comme pour l'arbre de décomposition d'une permutation (voir le paragraphe 2.3.3 page 52), le coût de calcul de l'arbre signé d'une permutation signée σ est linéaire en la taille de σ.

L'algorithme 7.1 explicite la procédure de [BBCP07].

Aux lignes 6 et 9 de l'algorithme 7.1, on utilise un algorithme de calcul de scénario parcimonieux de tri par renversements, sans contrainte de perfection. Le meilleur algorithme connu résolvant ce problème a une complexité $\mathcal{O}(n\sqrt{n \log n})$, en fonction de la taille n de la permutation en entrée.

La correction de l'algorithme 7.1 est démontrée dans [BBCP07], et on l'admettra ici. On s'intéresse plutôt à l'évaluation de la complexité de cet algorithme. On rappelle pour commencer le résultat de complexité de [BBCP07] :

Théorème 7.13. *La complexité de l'algorithme 7.1 est $\mathcal{O}(2^k \times n\sqrt{n \log n})$, où n est la taille de la permutation σ en entrée, et k le nombre de nœuds premiers dans l'arbre signé de σ.*

Démonstration. On considère σ une permutation signée de taille n, dont l'arbre signé $\mathcal{T}_\sigma$ à k nœuds premiers. Ainsi, il y a au plus 2^k nœuds non signés dans $\mathcal{T}_\sigma$. Le coût du calcul de $\mathcal{T}_\sigma$ à la ligne 1 est $\mathcal{O}(n)$. Il y a ensuite au plus 2^k affectations possibles des signes dans $\mathcal{T}_\sigma$ à examiner, c'est-à-dire au plus 2^k passages dans la boucle de la ligne 2. Pour chaque passage dans cette boucle, le calcul des renversements associés aux nœuds premiers $p_1, p_2, \ldots$ de taille $n_1, n_2, \ldots$ coûte

Algorithme 7.1 Calcul d'un scénario parfait parcimonieux de tri par renversements pour une permutation signée σ de taille n.

1: $\mathcal{T}_\sigma \leftarrow$ l'arbre signé associé à σ.
2: **pour** chaque affectation de signes aux nœuds de $\mathcal{T}_\sigma$ qui n'en ont pas **faire**
3: $S \leftarrow$ le scénario vide.
4: **pour** tout nœud premier p dans $\mathcal{T}_\sigma$ **faire**
5: **si** p a le signe + **alors**
6: Calculer un scénario parcimonieux de tri par renversements S' depuis la permutation quotient qui étiquette le nœud p vers $1\,2\ldots n$.
7: **sinon**
8: /* p a alors le signe $-$/*
9: Calculer un scénario parcimonieux de tri par renversements S' depuis la permutation quotient qui étiquette le nœud p vers $\overline{n}\ldots\overline{2}\,\overline{1}$.
10: **finsi**
11: Ajouter les renversements de S' à S.
12: **fin pour**
13: Ajouter à S les renversements des feuilles et des nœuds internes linéaires de $\mathcal{T}_\sigma$ dont le signe est différent de celui de leur père.
14: **fin pour**
15: Retourner le scénario S le plus court parmi tous ceux calculés dans la boucle précédente.

$\sum \mathcal{O}(n_i\sqrt{n_i \log n_i}) = \mathcal{O}(n\sqrt{n \log n})$. Le calcul du nombre de renversements associés aux nœuds internes linéaires et aux feuilles se fait à la ligne 13 en temps $\mathcal{O}(n)$. □

On remarque en particulier que l'algorithme 7.1 est *FPT*, pour le paramètre $k =$ le nombre de nœuds premiers dans l'arbre signé de la permutation signée en entrée. On s'intéressera au cas où $k = 0$, c'est-à-dire à la classe des permutations séparables dans le paragraphe 7.3.

Avant cela, on raffine l'analyse de la complexité de l'algorithme 7.1 en démontrant dans le paragraphe 7.2.2 que sa complexité en moyenne est polynomiale.

7.2.2 Analyse de complexité en moyenne

Un premier résultat est immédiatement déduit du théorème 7.13 et de la forme en moyenne des arbres de décomposition des permutations donnée par le théorème 2.70 page 60 :

Théorème 7.14. *Avec probabilité* 1*, l'algorithme 7.1 fonctionne en temps* $\mathcal{O}(n\sqrt{n \log n})$ *sur les permutations signées de taille* $n \to \infty$*.*

Démonstration. Le seul point qu'il peut être nécessaire de préciser est que la forme en moyenne des arbres *signés* des permutations signées est identique à la forme en moyenne des arbres de décomposition des permutations non signées sous-jacentes. En particulier, asymptotiquement et avec probabilité 1, ces arbres ont un unique nœud premier (voir le théorème 2.70 page 60). □

Cela démontre en particulier que l'algorithme 7.1 fonctionne en temps polynomial avec probabilité 1, asymptotiquement en taille des entrées, mais cela ne prouve pas qu'il est polynomial en moyenne. C'est pourtant ce qu'on constate sur des simulations, et c'est le résultat qu'on se propose de démontrer maintenant.

L'analyse en moyenne de l'algorithme 7.1 est menée sous une loi de probabilité uniforme sur l'ensemble des permutations d'une taille donnée. Évidemment, cette hypothèse ne rend pas compte d'une quelconque réalité biologique, qui estimerait que tous les ordres possibles entre les gènes sont équiprobables. Il est cependant admis qu'elle fournit un garde-fou sur la complexité de l'algorithme lorsqu'appliqué à des données biologiques réelles.

On étudie d'abord la proportion d'arbres de décomposition (de permutations non signées donc) qui ont un nombre donné de nœuds premiers. On introduit pour cela la notation suivante :

Définition 7.15. On note $T_{n,p}$ le nombre d'arbres de décomposition à n feuilles qui ont p nœuds premiers.

La correspondance entre permutations (non signées) et arbres de décomposition étant bijective, $T_{n,p}$ est aussi le nombre de permutations de taille n dont l'arbre de décomposition a p nœuds premiers.

Lemme 7.16. *Le nombre $T_{n,p}$ de permutations de taille n dont l'arbre de décomposition contient p nœuds premiers, avec $p \geq 2$, est au plus $48\frac{(n-1)!}{2^p}$.*

On rappelle que d'après le théorème 2.70, pour $p = 1$ on a $T_{n,1} \sim n!$, et que pour $p = 0$, $T_{n,0}$ est le $(n-1)$-ème nombre de Schröder d'après le théorème 3.11.

Démonstration. La preuve est par récurrence sur le nombre p de nœuds premiers. L'hypothèse de récurrence est la suivante : $(\mathcal{H}_p) : \forall n, T_{n,p} \leq 48\frac{(n-1)!}{2^p}$.

$(\mathcal{H}_p)$ est vraie par vacuité pour $n < 3p+1$ puisqu'un arbre avec p nœuds premiers a au moins $3p+1$ feuilles.

On commence par le cas de base $p = 2$, en supposant donc que $n \geq 7$. Bien qu'il n'y ait pas unicité de cette décomposition, un arbre de taille n ayant deux nœuds premiers peut toujours être décomposé de la manière suivante : un arbre $\mathcal{T}_1$ avec un nœud premier où une feuille est choisie et est dilatée par un deuxième arbre $\mathcal{T}_2$ ayant aussi un nœud premier. On remarque que $|\mathcal{T}_1|+|\mathcal{T}_2| = n+1$. On rappelle aussi que le nombre d'arbres à k feuilles ayant un nœud premier ne peut excéder $k!$. Ainsi,

$$T_{n,2} \leq \sum_{k=4}^{n-3} k!k(n+1-k)! \leq (n+1)! \sum_{k=4}^{n-3} \frac{k}{\binom{n+1}{k}} \leq \frac{(n+1)!}{\binom{n+1}{4}} \sum_{k=4}^{n-3} k \leq 48\frac{(n-1)!}{2^2}$$

On démontre maintenant l'hérédité de la propriété $(\mathcal{H}_p)$: on suppose que $(\mathcal{H}_p)$ est vraie et on prouve $(\mathcal{H}_{p+1})$. La démarche est similaire à celle employée pour le cas de base. En effet, un arbre ayant $p+1$ nœuds premiers peut être décomposé (même si cette décomposition n'est pas unique) en un arbre $\mathcal{T}_1$ avec p nœuds premiers dont une feuille est dilatée par un autre arbre $\mathcal{T}_2$ qui a un nœud premier. Comme expliqué plus haut, on peut supposer que $n \geq 3(p+1)+1$. On obtient ainsi que

$$\begin{aligned}
T_{n,p+1} &\leq \sum_{k=3p+1}^{n-3} T_{k,p}k(n+1-k)! \\
&\leq \frac{48}{2^p} \sum_{k=3p+1}^{n-3} (k-1)!k(n+1-k)! \\
&\leq \frac{48(n+1)!}{2^p} \sum_{k=3p+1}^{n-3} \frac{1}{\binom{n+1}{k}} \\
&\leq \frac{48(n+1)!}{2^p} \left[\frac{1}{\binom{n+1}{n-3}} + \sum_{k=3p+1}^{n-4} \frac{1}{\binom{n+1}{k}} \right] \\
&\leq \frac{48(n+1)!}{2^p} \left[\frac{1}{\binom{n+1}{4}} + (n-4-3p)\frac{1}{\binom{n+1}{5}} \right] \\
&\leq \frac{48(n+1)!}{2^p} \left[\frac{24}{(n+1)n(n-1)(n-2)} + (n-4-3p)\frac{120}{(n+1)n(n-1)(n-2)(n-3)} \right] \\
&\leq \frac{48(n-1)!}{2^{p+1}} \left[\frac{48}{(n-1)(n-2)} + (n-10)\frac{240}{(n-1)(n-2)(n-3)} \right]
\end{aligned}$$

Une analyse classique et sans piège donne facilement que $\left[\frac{48}{(n-1)(n-2)} + (n-10)\frac{240}{(n-1)(n-2)(n-3)}\right] \leq 1$ pour tout $n \geq 3(p+1)+1 \geq 10$.

Ceci achève la preuve du lemme 7.16. □

Théorème 7.17. *La complexité en moyenne de l'algorithme 7.1 est $\mathcal{O}(n\sqrt{n\log n})$. En particulier, cet algorithme fonctionne en temps polynomial en moyenne.*

Démonstration. Il faut ici revenir aux permutations *signées*. Il y a $2^n n!$ permutations signées de taille n, et un unique arbre signé étant associé à chacune de ces permutations, il y a $2^n T_{n,p}$ arbres signés à n feuilles et p nœuds premiers. Si on note C la constante telle que l'algorithme 7.1 fonctionne en temps $C2^p n\sqrt{n\log n}$ sur les permutations signées de taille n dont l'arbre signé a p nœuds premiers, la complexité en moyenne de cet algorithme est donc

$$\frac{C}{2^n n!}\sum_{p=0}^{n} 2^n T_{n,p} 2^p n\sqrt{n\log n} = \frac{C}{n!}\sum_{p=0}^{n} T_{n,p} 2^p n\sqrt{n\log n}.$$

En notant (Sch_n) la séquence des nombres de Schröder, on déduit facilement du lemme 7.16 que

$$\frac{C}{n!}\sum_{p=0}^{n} T_{n,p} 2^p n\sqrt{n\log n} \leq \frac{Cn\sqrt{n\log n}}{n!}\left(Sch_{n-1} + 2n! + \sum_{p=2}^{n} 48(n-1)!\right) \leq 50Cn\sqrt{n\log n},$$

concluant ainsi la preuve du théorème 7.17. □

On peut remarquer que la constante en facteur de $(n\sqrt{n\log n})$ dans la complexité en moyenne de l'algorithme 7.1 reste inconnue. On démontre qu'elle est au plus 50 dans la preuve du théorème 7.17, mais les expériences suggèrent plutôt une valeur légèrement supérieure à 2, comme illustré sur la figure 7.2.

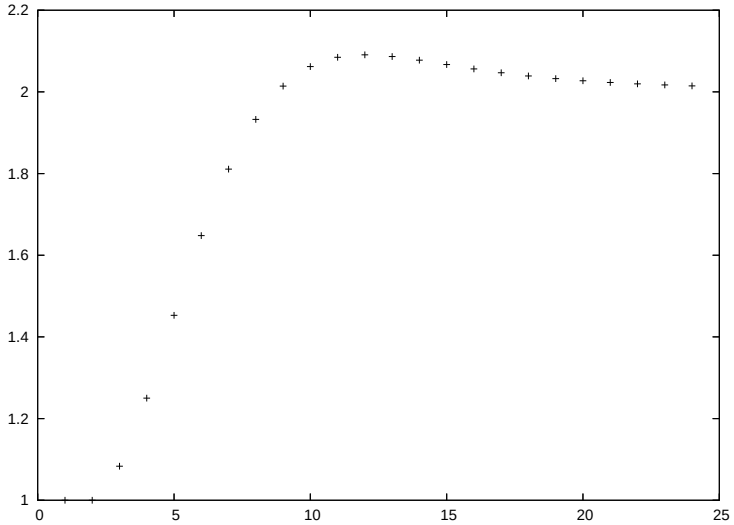


FIG. 7.2 – Graphique représentant les valeurs $p_n = \frac{\sum_p 2^p T_{n,p}}{n!}$, jusqu'à $n = 25$.

7.3 Le cas particulier des permutations séparables

On s'intéresse dans ce paragraphe à quelques propriétés des scénarios parcimonieux de tri parfait par renversements calculés par l'algorithme 7.1, sur une classe restreinte d'entrées : la classe des permutations séparables signées. La définition et quelques propriétés de cette classe (dans sa version non signée) sont rappelées page 71. En particulier, on étudiera dans ce cadre restreint le nombre moyen de renversements, et la taille moyenne d'un renversement dans un scénario parfait parcimonieux calculé par l'algorithme 7.1.

Remarquons d'abord que les permutations séparables sont exactement celles dont les arbres (signés ou de décomposition) n'ont aucun nœud premier. Cela implique en particulier que sur cette classe d'entrée, l'algorithme 7.1 est polynomial, pas seulement en moyenne. On s'aperçoit facilement qu'il est même de complexité linéaire dans ce cas.

En outre, les renversements du scénario parcimonieux parfait calculé par l'algorithme *commutent* deux à deux : pour tous renversements I et J, soit $I \subset J$, soit $J \subset I$, soit $I \cap J = \emptyset$. En particulier, l'ordre entre ces renversements n'importe pas : un scénario est composé dans ce cas d'un ensemble de renversements qui peuvent être effectués dans n'importe quel ordre. En fait, il est même démontré dans [BBCP07] que les scénarios parcimonieux parfaits pour les permutations séparables sont tous composés des renversements décrits à la ligne 13 de l'algorithme 7.1. Dans le cas des permutations séparables, tout nœud interne linéaire a un signe différent de son père (si ce père existe), et ainsi, on peut énoncer le théorème suivant :

Théorème 7.18. *Dans un scénario parcimonieux de tri parfait par renversements d'une permutation séparable σ, les renversements correspondent à tous les nœuds internes de l'arbre signé $T_{\mathcal{S}}(\sigma)$ associé à σ, sauf la racine, auxquels on ajoute tous ceux correspondant à des feuilles de $T_{\mathcal{S}}(\sigma)$ dont le signe est différent de celui de leur père.*

La description des permutations séparables à travers des propriétés de leurs scénarios parcimonieux de tri parfait par renversements peut même être poussée plus loin : les permutations séparables sont caractérisées par le fait que dans tout scénario parcimonieux de tri parfait par renversements, tous les renversements de ce scénario commutent deux à deux. Cela vaut l'autre nom de permutations *commutantes* (ou *commuting permutations*) à la classe des permutations séparables dans le contexte du tri parfait par renversements.

Pour obtenir des estimations du nombre moyen de renversements et de la taille moyenne d'un renversement dans un scénario parcimonieux parfait pour les permutations séparables, on utilise les techniques de combinatoire analytique de [FS08] dont les principes de base sont rappelés dans le chapitre 10.

7.3.1 Nombre moyen de renversements

On considère une permutation séparable σ (non signée) de taille n, dont l'arbre de décomposition $T_{\mathcal{S}}(\sigma)$ n'a donc pas de nœud premier. Du point de vue des structures combinatoires, on peut donc voir $T_{\mathcal{S}}(\sigma)$ comme un arbre plan (les fils d'un nœud sont ordonnés) dont les nœuds internes ont arité au moins 2, et où on ajoute un signe $\oplus$ ou $\ominus$ à la racine. Les signes des nœuds internes de $T_{\mathcal{S}}(\sigma)$ sont implicitement contenus dans cette description, sachant que les fils non feuilles d'un nœud $\oplus$ (resp. $\ominus$) sont tous d'étiquette $\ominus$ (resp. $\oplus$).

Les arbres de décomposition des permutations séparables (non signées) sont donc des *arbres de Schröder* avec un signe à la racine (voir page 73 et suivantes). De la même manière, les arbres signés des permutations séparables signées sont des arbres de Schröder dont la racine et les feuilles sont toutes affectées d'un signe.

Théorème 7.19. *Le nombre moyen de renversements dans un scénario parcimonieux parfait pour une permutation séparable signée de taille n est asymptotiquement*

$$\frac{1+\sqrt{2}}{2}n \simeq 1.2n.$$

Démonstration. D'après le théorème 7.18, on peut reformuler le problème d'estimer le nombre moyen de renversements dans un scénario parcimonieux de tri parfait par renversements pour une permutation séparable signée σ en celui de calculer le nombre moyen de nœuds internes (sauf la racine) dans l'arbre signé $T_{\mathcal{S}}(\sigma)$ associé, ainsi que le nombre moyen de feuilles dans cet arbre dont le signe diffère de celui de leur père.

Pour une permutation de taille n, le nombre de feuilles dans $T_{\mathcal{S}}(\sigma)$ qui ont un signe différent de celui de leur père est en moyenne $n/2$, puisque les signes d'une feuille et de son père sont indépendants.

Le nombre moyen de nœuds internes dans un arbre de Schröder ne varie pas selon qu'on considère des arbres de Schröder signés à la racine ou non. On s'intéresse donc au calcul du nombre moyen de nœuds internes dans un arbre de Schröder standard (non signé). Pour ce faire, on utilise la *méthode symbolique* de [FS08].

On définit la série génératrice bivariée $S(x,y) = \sum_{k,n} S_{n,k} x^n y^k$ où $S_{n,k}$ est le nombre d'arbres de Schröder ayant n feuilles et k nœuds internes. Le nombre moyen de nœuds internes dans un arbre de Schröder à n feuilles est donné par

$$\frac{\sum_k k S_{n,k}}{\sum_k S_{n,k}} = \frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)}.$$

Comme illustré en figure 7.3, on peut décrire récursivement un arbre de Schröder comme étant une feuille ou un nœud interne ayant au moins deux fils, qui sont eux-mêmes des arbres de Schröder.

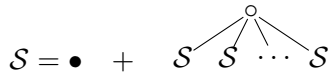


FIG. 7.3 – Description récursive de la classe combinatoire des arbres de Schröder.

Par conséquent, la série $S(x,y)$ satisfait l'équation

$$S(x,y) = x + y\frac{S(x,y)^2}{1-S(x,y)},$$

et la résolution de cette équation donne

$$S(x,y) = \frac{(x+1) - \sqrt{(x+1)^2 - 4x(y+1)}}{2(y+1)}. \tag{7.1}$$

On étudie d'abord le comportement asymptotique de la séquence $([x^n]S(x,1))$, qui correspond aux (petits) nombres de Schröder, dont la séquence porte la référence A001003 dans [Slo07]. Ces petits nombres de Schröder (r_n) sont reliés aux nombres de Schröder standards (Sch_n) par l'identité $2r_n = Sch_n$ pour tout $n \geq 1$.

Étude asymptotique de $[x^n]S(x,1)$. D'après l'équation 7.1, on a

$$S(x,1) = \frac{(x+1) - \sqrt{(x+1)^2 - 8x}}{4} = \frac{(x+1) - \sqrt{(1-\frac{x}{3+2\sqrt{2}})(1-\frac{x}{3-2\sqrt{2}})}}{4},$$

ce qui nous donne l'équivalent suivant lorsque $x \to 3-2\sqrt{2}$, $x < 3-2\sqrt{2}$

$$S(x,1) \sim \frac{2-\sqrt{2}}{2} - \frac{\sqrt{3\sqrt{2}-4}}{2}(1-\frac{x}{3-2\sqrt{2}})^{1/2}.$$

En appliquant les techniques de [FS08, chapitres 4 et 6], on obtient l'équivalent suivant des coefficients $[x^n]S(x,1)$ quand $n \to \infty$:

$$[x^n]S(x,1) \sim \frac{\sqrt{3\sqrt{2}-4}}{4}(3+2\sqrt{2})^n \frac{1}{\sqrt{\pi n^3}}.$$

Étude asymptotique de $[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}$.

D'après l'équation 7.1, on a

$$\frac{\partial S(x,y)}{\partial y}|_{y=1} = \frac{(x-1)^2-(x+1)\sqrt{(x+1)^2-8x}}{8\sqrt{(1-\frac{x}{3+2\sqrt{2}})(1-\frac{x}{3-2\sqrt{2}})}}.$$

L'expression ci-dessus permet de calculer un équivalent de $\frac{\partial S(x,y)}{\partial y}|_{y=1}$ lorsque $x \to 3-2\sqrt{2}$, $x < 3-2\sqrt{2}$. Plus précisément,

$$\frac{\partial S(x,y)}{\partial y}|_{y=1} \sim \frac{3-2\sqrt{2}}{4\sqrt{3\sqrt{2}-4}}(1-\frac{x}{3-2\sqrt{2}})^{-1/2}.$$

Avec la même technique que précédemment, on en déduit un équivalent des coefficients $[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}$ quand $n \to \infty$:

$$[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1} \sim \frac{3-2\sqrt{2}}{4\sqrt{3\sqrt{2}-4}}(3+2\sqrt{2})^n \frac{1}{\sqrt{\pi n}}$$

Combiner ces résultats pour conclure.

On obtient alors facilement l'équivalent asymptotique attendu du nombre moyen de nœuds internes dans un arbre de Schröder à n feuilles :

$$\frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)} \sim \frac{3-2\sqrt{2}}{3\sqrt{2}-4}n \sim \frac{n}{\sqrt{2}}.$$

On peut remarquer que ce nombre moyen de nœuds internes tient compte de la racine (dans le cas où elle n'est pas une feuille, c'est-à-dire dès que $n > 1$.)

Étant donné un arbre de Schröder à n feuilles, il y a 2^{n+1} manières de lui ajouter des signes pour en faire un arbre signé d'une permutation séparable signée (choix d'un signe parmi 2 pour la racine et pour chacune des feuilles). On a déjà remarqué que cette décoration par des signes ne change pas le nombre moyen de nœuds internes. Ainsi, le nombre moyen de nœuds internes dans l'arbre signé $T_{\mathcal{S}}(\sigma)$ d'une permutation séparable signée σ de taille n est asymptotiquement équivalent à $\frac{n}{\sqrt{2}}$. Tous ces nœuds internes correspondent à des renversements dans un scénario parcimonieux de tri parfait par renversements pour σ, sauf la racine qu'il faut soustraire, ce qui n'affecte pas l'asymptotique. Enfin, il suffit d'ajouter les renversements associés aux feuilles de $T_{\mathcal{S}}(\sigma)$ pour obtenir que le nombre moyen de renversements dans un scénario parcimonieux de tri parfait par renversements pour une permutation séparable signée est

$$\frac{1+\sqrt{2}}{2}n.$$

□

Il faut remarquer dans la preuve du théorème 7.19 qu'il apparaît qu'une grosse proportion (presque la moitié) des renversements dans un scénario parcimonieux de tri parfait par renversements d'une permutation séparable signée sont de taille 1. Cette propriété a été observée en pratique sur des génomes de bactéries (voir [LEMTS03]).

7.3.2 Taille moyenne d'un renversement

Toujours dans les scénarios parcimonieux de tri parfait par renversements pour les permutations séparables signées, on s'intéresse maintenant à l'estimation de la valeur moyenne d'un autre paramètre : la taille moyenne d'un renversement.

Théorème 7.20. *La taille moyenne d'un renversement dans un scénario parcimonieux parfait pour une permutation séparable signée de taille n est asymptotiquement*

$$\frac{2^{7/4}\sqrt{3-2\sqrt{2}}}{1+\sqrt{2}}\sqrt{\pi n} \simeq 1.02\sqrt{n}.$$

Démonstration. On veut évaluer le ratio entre la valeur moyenne de la somme des tailles des renversements dans un scénario parcimonieux parfait pour une permutation séparable, et le nombre moyen de renversements dans un tel scénario. Le théorème 7.19 donne la deuxième de ces valeurs, on se concentre donc sur le calcul de la première.

La taille d'un renversement associé à un nœud x dans l'arbre signé d'une permutation séparable signée est égale au nombre de feuilles dans le sous-arbre de racine x. On dira que la taille d'un sous-arbre est le nombre de feuilles qu'il contient. On commence donc par calculer la valeur moyenne de la somme des tailles de tous les sous-arbres dans un arbre de Schröder (standard). Comme avant, on obtiendra ainsi la valeur moyenne de la somme des tailles de tous les sous-arbres dans un arbre signé d'une permutation séparable signée, l'étiquetage par des signes n'influençant pas cette valeur moyenne. En revanche, on compte dans l'ensemble des sous-arbres celui enraciné à la racine de l'arbre, et aussi toutes ses feuilles, qui ne correspondront pas à des renversements.

On définit une nouvelle série génératrice bivariée (notée encore $S(x,y)$, bien qu'elle soit légèrement différente de celle définie dans la preuve du théorème 7.19) par $S(x,y) = \sum_{k,n} S_{n,k}x^n y^k$ où $S_{n,k}$ représente le nombre d'arbres de Schröder à n feuilles et dont les tailles des sous-arbres (y compris l'arbre entier et les sous-arbres réduits à des feuilles) somment à k. La valeur moyenne de la somme des tailles de tous les sous-arbres dans un arbre de Schröder à n feuilles est donc donnée par :

$$\frac{\sum_k kS_{n,k}}{\sum_k S_{n,k}} = \frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)}.$$

Comme dans la preuve du théorème 7.19, un arbre de Schröder peut être décrit récursivement comme étant soit une feuille, soit un nœud interne racine ayant au moins deux fils, qui sont eux-mêmes des arbres de Schröder. Dans le deuxième de ces cas, les sous-arbres à prendre en compte sont les sous-arbres contenus dans chacun des fils de la racine, plus l'arbre lui-même, ce qui conduit à l'équation fonctionnelle suivante

$$S(x,y) = xy + \frac{S(xy,y)^2}{1-S(xy,y)}. \tag{7.2}$$

Cette équation faisant intervenir à la fois $S(x,y)$ et $S(xy,y)$, on ne peut pas en extraire une expression de $S(x,y)$ comme dans la preuve du théorème 7.19. Mais la quantité qui nous intéresse (la valeur moyenne de la somme des tailles de tous les sous-arbres dans un arbre de Schröder à n feuilles) est obtenue par la formule

$$\frac{\sum_k kS_{n,k}}{\sum_k S_{n,k}} = \frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)},$$

de sorte qu'il n'est pas nécessaire de calculer $S(x,y)$ mais seulement $S(x,1)$ et $\frac{\partial S(x,y)}{\partial y}|_{y=1}$.

Étude asymptotique de $[x^n]S(x,1)$.

D'après l'équation 7.2, on a $S(x,1) = \frac{(x+1)-\sqrt{(x+1)^2-8x}}{4}$. C'est la même fonction que dans la preuve du théorème 7.19, et on obtient donc que

$$[x^n]S(x,1) \sim \frac{\sqrt{3\sqrt{2}-4}}{4}(3+2\sqrt{2})^n \frac{1}{\sqrt{\pi n^3}}.$$

Étude asymptotique de $\frac{\partial S(x,y)}{\partial y}|_{y=1}$.

En dérivant l'équation 7.2 et en évaluant le résultat de cette dérivation en $y=1$, on obtient le système :

$$\begin{cases} \frac{\partial S}{\partial x}(x,1) = 1 + \frac{\partial S}{\partial x}(x,1) \cdot \frac{2S(x,1)-S(x,1)^2}{(1-S(x,1))^2} \\ \frac{\partial S}{\partial y}(x,1) = x + \left(x\frac{\partial S}{\partial x}(x,1) + \frac{\partial S}{\partial y}(x,1)\right) \cdot \frac{2S(x,1)-S(x,1)^2}{(1-S(x,1))^2}. \end{cases}$$

De ce système, on peut extraire l'expression suivante, où $S(x,1)$ vaut $\frac{(x+1)-\sqrt{(x+1)^2-8x}}{4}$ comme calculé plus haut :

$$\frac{\partial S(x,y)}{\partial y}|_{y=1} = \frac{\partial S}{\partial y}(x,1) = \frac{x}{(1-C)^2}, \text{ avec } C = \frac{2S(x,1)-S(x,1)^2}{(1-S(x,1))^2}.$$

La singularité la plus proche de l'origine est $3-2\sqrt{2}$, et le développement de Taylor autour de cette singularité donne :

$$\frac{\partial S(x,y)}{\partial y}|_{y=1} \sim \frac{3-2\sqrt{2}}{2(1-\frac{x}{3-2\sqrt{2}})}.$$

En appliquant les techniques de [FS08], on aboutit à un équivalent des coefficients $[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}$ quand $n \to \infty$ donné par la formule :

$$[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1} \sim \frac{3-2\sqrt{2}}{2}(3+2\sqrt{2})^n.$$

La moyenne de la somme des tailles de tous les sous-arbres d'un arbre de Schröder à n feuilles est alors donnée par :

$$\frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)} \sim 2^{3/4}\sqrt{3-2\sqrt{2}}\ \sqrt{\pi n^3}.$$

Cet équivalent est donc aussi celui de la valeur moyenne de la somme des tailles des sous-arbres de $T_{\mathcal{S}}(\sigma)$ pour une permutation séparable signée σ de taille n.

Pour obtenir la valeur moyenne de la somme des tailles des renversements dans un scénario parcimonieux parfait pour σ, il faut lui soustraire la taille des sous-arbres qui ont été comptés en trop : l'arbre lui-même (de taille n) et les feuilles de l'arbre (n feuilles de taille 1). Enfin, il n'y a pas seulement les renversements associés aux nœuds internes (sauf la racine) de $T_{\mathcal{S}}(\sigma)$ à prendre en compte : il faut aussi ajouter la contribution des renversements de taille 1 associés aux feuilles dont le signe est différent de celui de leur père ($n/2$ en moyenne). Ces ajustements sont négligeables devant l'asymptotique obtenue.

Ainsi, en divisant finalement par le nombre moyen de renversements dans un scénario parcimonieux parfait pour une permutation séparable signée σ de taille n (obtenu au théorème 7.19), on aboutit au résultat annoncé dans le théorème 7.20. □

Ce chapitre illustre bien les raffinements dans l'étude des algorithmes que peut apporter une analyse combinatoire. En effet, bien que l'algorithme 7.1 que nous avons étudié dans tout ce chapitre

était déjà connu, on disposait de peu de résultats sur sa complexité, et sur les scénarios qu'il calcule. Avec l'outil combinatoire des arbres de décomposition, on a démontré que sa complexité, bien qu'exponentielle en générale, est polynomiale en moyenne. Les techniques de combinatoire analytique ont aussi permis d'évaluer la valeur moyenne de deux paramètres (nombre de renversements et taille d'un renversement) des scénarios calculés par cet algorithme dans le cas restreint des permutations séparables. Il serait intéressant d'utiliser cette approche combinatoire pour l'analyse d'autres problèmes et algorithmes, en particulier issus du champ bio-informatique. Un premier problème qu'on pourrait envisager serait l'étude des scénarios parfaits de réarrangements dans le modèle de *Double-Cut-and-Join* (défini dans [BCC$^+$08]) pour lequel les instances faciles à résoudre (c'est-à-dire polynomiales) sont celles difficiles à résoudre dans le modèle du tri parfait par renversements, et vice-versa.

Quatrième partie

Étude de la nature des séries génératrices des classes de permutations

« Vous ne les comblerez point de formules qui sont vides, mais d'images qui charrient des structures. »

Antoine de Saint-Exupéry, *Citadelle*

Automatisation du calcul des séries génératrices dans des classes de permutations “structurées”

L’idée de “structure” dans les classes de permutations

Les guillemets dans le titre de cette introduction à la quatrième partie se veulent porteurs du message que la structure d’une classe de permutations n’est pas une propriété formellement définie, mais plutôt un ensemble de caractéristiques qui suggèrent qu’une structure intrinsèque est présente dans les permutations de cette classe, lui conférant en particulier de “bonnes” propriétés du point de vue combinatoire.

On pourrait dire que lorsqu’on y est confronté, la structure d’une classe de permutations “se voit bien”, par exemple parce qu’il y a une signification profonde pour l’appartenance d’une permutation à la classe. Cependant, il n’existe pas pour l’instant de manière canonique de définir formellement qu’une classe de permutations est structurée (et on peut douter qu’il en existera une un jour).

L’étude de la structure des classes de permutations est une manière de prendre du recul sur les travaux combinatoires existants concernant les classes de permutations. Plutôt que de s’attacher aux propriétés particulières d’une classe donnée, elle explore des propriétés générales des classes, et les conséquences que peuvent avoir ces propriétés, en combinatoire ou en algorithmique. Les développements de cette partie illustrent qu’une bonne connaissance de la structure des classes de permutations permet de concevoir des algorithmes plus efficaces, et d’obtenir systématiquement certaines propriétés combinatoires.

Sans définir l’idée de structure d’une classe de permutations, on propose de l’expliciter, à travers quelques exemples de critères rencontrés dans les chapitres précédents de cette thèse qui peuvent indiquer qu’une classe de permutations est structurée.

Lorsque les permutations appartenant à une classe $\mathcal{C}$ peuvent être décrites récursivement comme composées de permutations de $\mathcal{C}$ plus petites, cela laisse apparaître une structure dans la classe $\mathcal{C}$. C’est ce qui se produit par exemple pour les permutations triables par pile, définies page 84 comme celles qui s’écrivent InJ, pour deux permutations I et J triables par pile. Le même type de description récursive justifie aussi que la classe des permutations séparables (voir page 71 et suivantes) est structurée, puisque ces permutations se décomposent en une concaténation (positive ou négative) de permutations séparables plus petites.

Les nombreuses caractérisations des permutations séparables autorisent d’autres manières de faire apparaître la structure de cette classe. Les permutations séparables peuvent par exemple être codées par leurs arbres de séparation, qui sont des objets possédant une structure explicite, transmise aux permutations.

La base de motifs exclus qui caractérise la classe des permutations séparables est $\{2\,4\,1\,3, 3\,1\,4\,2\}$, c’est-à-dire que cette base est composée des deux permutations simples de taille 4, qui apparaissent l’une ou l’autre comme motif dans toute permutation simple de taille supérieure. Ainsi, la base de cette classe satisfait une propriété particulière qui confère une structure aux permutations

séparables. Parmi les classes dont la structure est justifiée par une base de motifs exclus possédant de bonnes propriétés, on peut penser aussi à toutes les classes fermées par produit de substitution, caractérisées par le fait que leur base est composée exclusivement de permutations simples.

Sans aller jusqu'aux classes fermées par produit de substitution, le produit de substitution est un moyen de structurer une classe : une classe décrite comme produit de substitution de deux classes dont on comprend bien la structure est évidemment elle aussi structurée. Ce principe n'est pas valable seulement pour le produit de substitution mais pour toute méthode de construction de classes à partir de classes plus petites,

On peut encore mentionner les classes de permutations qui sont décrites comme celles calculées par des *permuting machines*, ou d'autres machines de mélange, tels des dispositifs de tri à partir de piles et de files. Des exemples de cette nature apparaissent dès l'article [AS02], où est défini le produit de substitution, et où la question de la structure des classes de permutations est posée pour la première fois, dans un but d'énumération par séries génératrices.

On cite enfin le cas où la structure d'une classe réside dans un codage de ses permutations par des mots. Il peut s'agir par exemple d'un codage par un langage sur un alphabet fini (comme on va le voir pour les permutations en épingles) ou par utilisation du principe d'*insertion encoding* qu'on a brièvement présenté à la fin du chapitre 1.

Dans certains cas, la manière dont émerge la structure d'une classe de permutations peut être directement mise à profit pour obtenir des résultats sur la séquence d'énumération de cette classe, à travers sa série génératrice.

C'est le cas par exemple pour les descriptions récursives associées aux formalismes des arbres de génération ou de la méthode *ECO*. Les règles de production des permutations de taille $n+1$ à partir de celles de taille n donnent des équations satisfaites par les séries génératrices, et le calcul de ces équations à partir du système de règles de réécriture est automatique.

On s'intéresse dans la suite à une autre façon de faire émerger la structure dans les classes de permutations, qui utilise la décomposition par substitution, et qui permet aussi d'extraire automatiquement (des équations satisfaites par) les séries génératrices des classes.

On s'aperçoit que dans ces deux cas, la structure des classes se retrouve dans leurs séries génératrices, et les propriétés de ces séries génératrices (comme être rationnelle ou algébrique) deviennent des témoins de la structure des classes considérées.

Les étapes du calcul automatique des séries génératrices

On décrit plus en détails ici un outil pour rechercher une forme de structure dans une classe de permutations. Les travaux exposés dans cette partie IV seront, à des degrés plus ou moins directs, consacrés à la mise en œuvre de cet outil.

Le théorème fondamental sur lequel repose le cheminement exposé dans la suite est dû à M. H. Albert et M. D. Atkinson dans [AA05]. On a déjà mentionné ce résultat à plusieurs reprises dans les pages qui précèdent. Il énonce que :

Théorème. [AA05]. *Pour toute classe de permutations $\mathcal{C}$, si $\mathcal{C}$ contient un nombre fini de permutations simples, alors la base de $\mathcal{C}$ est finie et la série génératrice de $\mathcal{C}$ est algébrique.*

Outre le caractère très général du résultat, une grande force de ce théorème est que sa preuve fournit une méthodologie pour poser un système d'équations fonctionnelles satisfait par la série génératrice de la classe, et offre ainsi les moyens de calculer cette série automatiquement, connaissant les permutations simples de la classe. La démonstration repose sur la décomposition par substitution des permutations, et même si elle n'est pas formulée en termes d'arbres de décomposition, ce formalisme est très adapté ici pour automatiser la méthodologie proposée pour le calcul de la série génératrice.

Une question algorithmique naturelle est soulevée par le théorème de [AA05] : peut-on tester l'hypothèse de ce théorème, c'est-à-dire peut-on algorithmiquement décider si une classe de permutations contient un nombre fini de permutations simples ?

Une première réponse est donnée dans [AA05], en utilisant un résultat de J. H. Schmerl et W. T. Trotter dans [ST93], qui figure dans cette thèse comme le théorème 2.27. Ce théorème énonce que toute permutation simple de taille $n \geq 5$ contient (en tant que motif) une permutation simple de taille $n-1$ ou $n-2$. Il assure donc que la classe $\mathcal{C}$ contient un nombre fini de permutations simples si et seulement s'il existe deux tailles consécutives n et $n+1$ telles que $\mathcal{C}$ ne contienne aucune permutation simple de taille n ou $n+1$.

Une manière de tester si une classe $\mathcal{C}$ contient un nombre fini de permutations simples est donc de tester par taille n croissante si $\mathcal{C}$ contient des permutations simples de taille n (en mémorisant au passage lesquelles pour pouvoir calculer effectivement la série génératrice dans le cas où le nombre de permutations simples serait fini). Si l'on trouve deux tailles consécutives n et $n+1$ pour lesquelles $\mathcal{C}$ ne contient pas de permutations simples, le résultat de J. H. Schmerl et W. T. Trotter assure que toutes les permutations simples de $\mathcal{C}$ sont de taille inférieure à n. On a donc mis en évidence que $\mathcal{C}$ contient un nombre fini de permutations simples, et on connait ces permutations.

Le premier problème de cette procédure est qu'il s'agit seulement d'un semi-algorithme : la procédure ne termine que lorsque $\mathcal{C}$ contient un nombre fini de permutations simples, et rien ne permet de l'assurer à l'avance. Le second problème est la complexité, fortement exponentielle, de cette procédure naïve, qui interdit de la voir fonctionner sur des classes contenant des permutations simples de taille supérieure à 9 ou 10, quand bien même ces classes contiendraient un nombre fini de permutations simples.

R. Brignall, N. Ruškuc et V. Vatter se sont penchés dans [BRV08] sur le premier de ces deux problèmes. Dans cet article, sous l'hypothèse que la classe $\mathcal{C}$ est décrite par sa base finie de motifs exclus, les trois auteurs fournissent une procédure effective pour tester si le nombre de permutations simples dans la classe est fini. L'idée est bien sûr d'appliquer ensuite le théorème de [AA05] pour obtenir l'algébricité de la série génératrice.

Les différentes étapes de la procédure de [BRV08] seront présentées dans le chapitre 9. On peut d'ores et déjà annoncer que cette procédure est effective, mais qu'aucune étude de complexité n'est faite dans [BRV08]. Après une analyse attentive, la procédure décrite dans [BRV08] s'avère exponentielle. On proposera dans le chapitre 9 un algorithme polynomial adapté de celui de [BRV08] pour décider si une classe donnée par sa base finie contient un nombre fini de permutations simples ou non.

Le second problème, à savoir celui de calculer les permutations simples qui appartiennent à une classe sachant que leur nombre est fini, n'a pas été étudié jusqu'ici. Reste bien sûr la méthodologie utilisant le théorème de J. H. Schmerl et W. T. Trotter, qui est promue au rang d'algorithme par la procédure de décision de [BRV08] : si l'on sait décider *a priori* qu'une classe $\mathcal{C}$ contient un nombre fini de permutations simples, on obtient une garantie sur la terminaison de la procédure cherchant les permutations simples de $\mathcal{C}$ par taille croissante. Mais pour achever de rendre vraiment effectif le théorème de [AA05], il faudrait trouver un algorithme plus efficace que cette procédure naïve calculant les permutations simples qui appartiennent à une classe.

La classe des permutations en épingles

Cette classe de permutations a été introduite dans la procédure de décision de [BRV08] permettant de savoir si une classe de permutations donnée par sa base finie contient un nombre fini de permutations simples. Elle y joue un rôle fondamental.

Les deux chapitres de la partie IV se concentrent sur les permutations en épingles, dans le but d'améliorer la procédure de décision de [BRV08].

Il faut remarquer que la classe des permutations en épingles est à la fois un exemple de classe dont la structure interne peut être mise en évidence grâce aux arbres de décomposition, et un outil pour exhiber une structure dans d'autres classes de permutations, en montrant qu'elles contiennent un nombre fini de permutations simples.

Dans le chapitre 8, on mène une étude approfondie de la classe des permutations en épingles. Ces permutations n'étant décrites que par une propriété graphique, on en donne une caractérisation plus utilisable aussi bien combinatoirement qu'algorithmiquement : on caractérise les arbres de décomposition qui correspondent à des permutations en épingles. La preuve de ce résultat est plutôt technique, et repose sur une analyse fine des représentations en épingles des permutations. Une première conséquence de cette caractérisation des permutations en épingles est qu'elle permet le calcul de la série génératrice associée à cette classe. Il était conjecturé dans [BRV08] que la série génératrice des permutations en épingles était rationnelle, puisque ces permutations peuvent être codées par un langage rationnel de mots, mais sans qu'il y ait unicité du codage. Le résultat obtenu est une série génératrice rationnelle, répondant ainsi par l'affirmative à cette conjecture. On termine le chapitre 8 en s'intéressant à la base de la classe des permutations en épingles, et en démontrant que cette base est infinie. Ceci répond à une autre remarque de [BRV08], qui jugeait "peu évident" le caractère fini de la base de la classe des permutations en épingles. Les résultats du chapitre 8 sont présentés dans [BBR09].

Le chapitre 9 reprend la procédure de [BRV08] permettant de décider si le nombre de permutations simples dans une classe $\mathcal{C}$ est fini, et s'attache à transformer cette procédure en un véritable algorithme, de complexité polynomiale. Le principe de fonctionnement reste fondamentalement inchangé : construire un automate qui accepte les mots d'épingles stricts des permutations en épingles propres appartenant à $\mathcal{C}$, et tester si le langage accepté par cet automate est fini. Mais il y a plusieurs points qu'il faut analyser avec attention pour éviter une explosion combinatoire, et conserver le caractère polynomial. Un premier point, évident avec la caractérisation du chapitre 8, consiste à proposer un algorithme testant si une permutation donnée est ou non une permutation en épingle. Les deux principales difficultés sont de savoir décrire précisément l'ensemble des mots d'épingles associés à une permutation en épingles σ, puis de construire en temps polynomial un automate *déterministe* qui accepte les mots d'épingles stricts associés aux permutations en épingles propres dont σ est motif. Pour résoudre la première, on utilise la caractérisation du chapitre 8, et une étude là encore assez technique permet de comprendre comment sont formés tous les mots d'épingles associés à une permutation en épingles. Pour la seconde, on construira les automates cherchés par récurrence, en utilisant l'astuce d'une lecture de droite à gauche des mots d'épingles pour obtenir des automates qui soient déterministes. En mettant bout à bout ces différents points, on obtient un algorithme testant en temps polynomial si une classe donnée par sa base finie de motifs exclus contient un nombre fini de permutations simples. L'article [BBPR] en préparation reprendra le contenu du chapitre 9.

Chapitre 8

Arbres de décomposition des permutations en épingles, et série génératrice de la classe

Sommaire

8.1 Définitions et propriétés autour des permutations en épingles . . . 172
8.1.1 Représentations en épingles et permutations en épingles 172
8.1.2 Représentations en épingles et mots d'épingles 175
8.1.3 Familles des épis et des quasi-épis. 176
8.2 Caractérisation des arbres de décomposition des permutations en épingles 180
8.2.1 Remarques préliminaires 180
8.2.2 Propriétés des nœuds linéaires 181
8.2.3 Propriétés des nœuds premiers 183
8.2.4 Preuve de la condition nécessaire 185
8.2.5 Preuve de la condition suffisante 188
8.3 Série génératrice des permutations en épingles simples 189
8.3.1 Propriétés des premiers points d'une représentation en épingles d'une permutation en épingles simple 189
8.3.2 Énumération des représentations en épingles simples 195
8.3.3 Énumération des permutations en épingles simples 198
8.4 Série génératrice de la classe des permutations en épingles 199
8.4.1 Une équation définissant les arbres de décomposition des permutations en épingles 199
8.4.2 Séries génératrices simples mises en jeu 200
8.4.3 Série génératrice de la classe 201
8.5 Discussion sur la base de la classe des permutations en épingles . . 202

Les permutations en épingles sont au cœur de la procédure développée par R. Brignall, N. Ruškuc et V. Vatter dans [BRV08] pour décider si une classe donnée par sa base finie contient un nombre fini ou non de permutations simples, ce qui est un critère suffisant pour que la série génératrice de la classe soit algébrique. Bien que le travail [BRV08] soit très récent, les permutations en épingles en elles-même, mais aussi des techniques consistant à insérer des épingles dans des permutations, ont été utilisées dans d'autres travaux contemporains de [BRV08]. On peut citer par exemple [Bri07], où les épingles sont un outil pour l'étude du produit de substitution sur les permutations, et permettent de construire des classes ayant des bases infinies, [Bria] qui s'intéresse à une famille particulière de classes de permutations (les *grid classes*) et propose aussi des constructions d'antichaînes infinies à base d'épingles, ou encore [ARS] qui s'attache à donner une compréhension approfondie des classes de permutations fermées par produit de substitution.

Ce chapitre est entièrement consacré à l'étude de la classe des permutations en épingles. On en donne au théorème 8.25 une caractérisation à partir de leurs arbres de décomposition, permettant de calculer la série génératrice de cette classe (théorème 8.58), qui se trouve être rationnelle. Enfin, on verra avec le théorème 8.64 que la base de motifs exclus caractérisant les permutations en épingles est infinie.

8.1 Définitions et propriétés autour des permutations en épingles

Le début de ce chapitre rappelle les définitions et propriétés essentielles sur les permutations en épingles et les notions associées.

8.1.1 Représentations en épingles et permutations en épingles

Définition 8.1. La *boîte englobante* (ou *bounding box*) d'un ensemble E de points d'une permutation, donnée par sa représentation graphique, est le plus petit rectangle (dont les côtés sont horizontaux et verticaux) qui contienne tous les points de E (voir un exemple en figure 8.1). Cette boîte englobante définit neuf régions du plan :

- les *côtés* de la boîte englobante, notés U, L, R et D (pour *up*, *left*, *right* et *down*), comme sur la figure 8.1),
- les coins de la boîte englobante, aussi appelés *quadrants*, et numérotés $1, 2, 3$ et 4 comme indiqué sur la figure 8.1).
- et la boîte englobante elle-même.

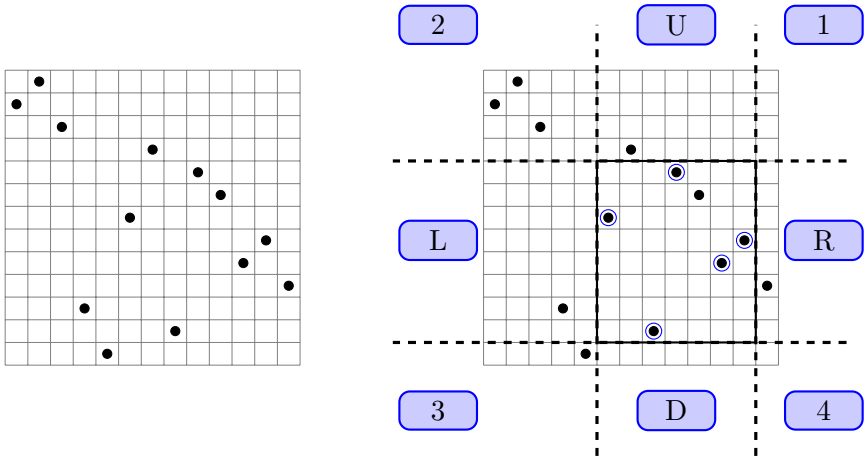


FIG. 8.1 – Représentation graphique de la permutation $\sigma = 12\,13\,11\,3\,1\,7\,10\,2\,9\,8\,5\,6\,4$ et boîte englobante de l'ensemble des points $\{7, 2, 9, 5, 6\}$.

On généralise très légèrement la notion de représentation graphique d'une permutation. Un *diagramme* est un ensemble de points du quart de plan positif, à coordonnées entières, tels que deux d'entre eux ne sont jamais alignés horizontalement ou verticalement. Tout diagramme est isomorphe à la représentation graphique d'une permutation, en effaçant les lignes et les colonnes ne contenant aucun point. Les points d'un diagramme sont aussi appelés des *épingles*. Dans un diagramme, on dit qu'une épingle p_i *sépare* deux ensembles de points E et F disjoints lorsque E et F se trouve de part et d'autre d'une ligne, soit horizontale soit verticale, passant par p_i.

Définition 8.2. Soit $\sigma \in \mathcal{S}_n$ une permutation. Une *représentation en épingles* (en anglais, *pin representation*) de σ est une séquence de points $(p_1, p_2, \ldots, p_n)$ de la représentation graphique de σ (qui couvre tous ses points) et telle que chaque point $p_i, i \geq 3$ satisfasse les deux conditions suivantes :

– la condition d'*extériorité* : p_i se trouve à l'extérieur de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$,

– $\begin{cases} \text{soit la condition de } \textit{séparation} : \\ \qquad p_i \text{ sépare } p_{i-1} \text{ de } \{p_1, \ldots, p_{i-2}\}, \\ \text{soit la condition d'} \textit{indépendance} : \\ \qquad p_i \text{ n'est pas sur les côtés de la boîte englobante de } \{p_1, \ldots, p_{i-1}\}. \end{cases}$

On dit qu'une épingle satisfaisant les conditions d'extériorité et d'indépendance (resp. de séparation) est une épingle *indépendante* (resp. *séparante*). Un exemple de représentation en épingles est donné sur la figure 8.2.

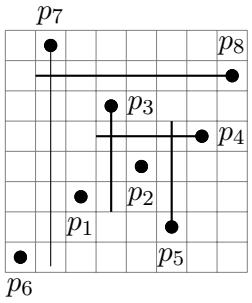


FIG. 8.2 – Une représentation en épingles de la permutation $\sigma = 1\,8\,3\,6\,4\,2\,5\,7$. Toutes les épingles $p_3, \ldots, p_8$ sont séparantes, sauf p_6 qui est indépendante.

Remarque 8.3. Il apparaît clairement dans la définition 8.2 que les deux premiers points p_1 et p_2 d'une représentation en épingles jouent le même rôle, et peuvent être intervertis dans la séquence $(p_1, p_2, \ldots, p_n)$ sans qu'on altère la propriété de cette séquence d'être une représentation en épingles : si $(p_1, p_2, p_3, \ldots, p_n)$ est une représentation en épingles, alors $(p_2, p_1, p_3, \ldots, p_n)$ aussi.

Les représentations en épingles selon la définition 8.2 sont plus restreintes que les *séquences d'épingles* (*pin sequences*) au sens de [BHV08a, BRV08] : une représentation en épingles doit couvrir tous les points d'une permutation, alors que cette condition n'est pas forcément vérifiée par une séquence d'épingles. Cette différence justifie le changement de terminologie. Remarquons cependant que les représentations en épingles *propres* sont les mêmes objets que les séquences d'épingles propres définies dans [BHV08a].

Définition 8.4. Soit $\sigma \in \mathcal{S}_n$ une permutation. Une *représentation en épingles propre* de σ est une représentation en épingles $(p_1, p_2, \ldots, p_n)$ de σ où toute épingle p_i, pour $i \geq 3$, est une épingle séparante.

Certaines permutations n'admettent pas de représentation en épingles : c'est le cas par exemple de $\sigma = 7\,1\,2\,3\,8\,4\,5\,6$. Cette remarque autorise à définir les *permutations en épingles*.

Définition 8.5. On appelle *permutation en épingles* toute permutation admettant une représentation en épingles.

Remarque 8.6. Une même permutation en épingles peut admettre plusieurs représentations en épingles, comme l'illustre la figure 8.3

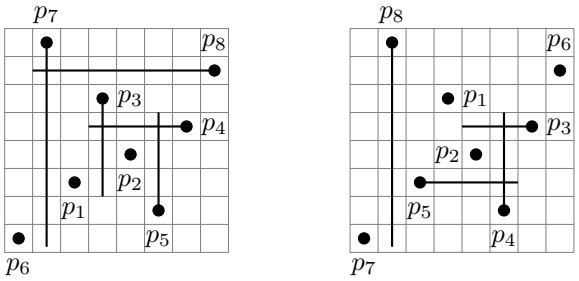


FIG. 8.3 – Deux représentations en épingles de $\sigma = 1\,8\,3\,6\,4\,2\,5\,7$.

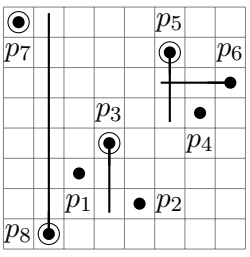
Les permutations en épingles forment, comme on pouvait s'y attendre, une classe de permutations.

Lemme 8.7. *L'ensemble des permutations en épingles est une classe de permutations. En outre, si p est une représentation en épingles d'une permutation σ, alors pour tout motif π de σ, une représentation en épingles pour π peut être extraite de p, en conservant les points p_i qui forment une occurrence de π dans σ.*

Ce résultat est mentionné sans preuve dans [BRV08]. Par souci d'exhaustivité, on en donne une preuve ci-dessous.

Démonstration. Soit σ une permutation en épingles, $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ, et π un motif de σ. On note $p_{i_1}, p_{i_2}, \ldots, p_{i_k}$, avec $i_1 < i_2 < \ldots < i_k$, les points de p qui forment une occurrence de π dans σ. En posant $q_j = p_{i_j}$ pour tout j, on affirme que $q = (q_1, \ldots, q_k) = (p_{i_1}, \ldots, p_{i_k})$ est une représentation en épingles de π, ce qui démontrera le lemme 8.7.

On prouve que pour tout $j > 1$, q_j satisfait soit les conditions d'extériorité et d'indépendance, soit les conditions d'extériorité et de séparation, par rapport à $\{q_1, \ldots, q_{j-1}\}$. On distingue deux cas, selon les conditions satisfaites dans p par p_{i_j} par rapport à $\{p_1, \ldots, p_{i_j-1}\}$.



- p_5 satisfait la condition de séparation dans p et la condition d'indépendance dans q
- p_7 satisfait la condition d'indépendance dans p et dans q
- p_8 satisfait la condition de séparation dans p et dans q

FIG. 8.4 – Les différents cas dans la preuve du lemme 8.7 sur une représentation en épingles p de $\sigma = 8\,1\,3\,4\,2\,7\,5\,6$. Les éléments 8, 1, 4 et 7 forment dans σ une occurrence du motif $\pi = 4\,1\,2\,3$. La représentation en épingles extraite $q = (q_1, q_2, q_3, q_4) = (p_3, p_5, p_7, p_8)$ correspondant à cette occurrence de π est indiquée par des points entourés.

– Si p_{i_j} satisfait les conditions d'extériorité et d'indépendance par rapport à $\{p_1, \dots, p_{i_j-1}\}$, alors q_j satisfait aussi ces conditions par rapport à $\{p_{i_1}, \dots, p_{i_{(j-1)}}\} = \{q_1, \dots, q_{j-1}\}$.
– Si p_{i_j} satisfait les conditions d'extériorité et de séparation par rapport à $\{p_1, \dots, p_{i_j-1}\}$, on distingue encore deux sous-cas, selon que $i_{(j-1)} = i_j - 1$ ou $i_{(j-1)} < i_j - 1$. Si $i_{(j-1)} = i_j - 1$, alors p_{i_j} sépare $p_{i_j-1} = p_{i_{(j-1)}}$ de $\{p_1, \dots, p_{i_{(j-1)}-1}\}$, de sorte que q_j sépare q_{j-1} de $\{q_1, \dots, q_{j-2}\}$. Ainsi q_j satisfait les conditions d'extériorité et de séparation dans q. Sinon, c'est-à-dire si $i_{(j-1)} < i_j - 1$, alors p_{i_j} les conditions d'extériorité et d'indépendance par rapport à $\{p_1, \dots, p_{i_j-2}\}$, de sorte que q_j satisfait les conditions d'extériorité et d'indépendance par rapport à $\{q_1, \dots, q_{j-1}\}$.

□

Parmi les motifs d'une permutation en épingles σ, on utilisera souvent le lemme 8.7 sur les motifs qui correspondent à des blocs (ou intervalles) de σ, et on propose donc un énoncé du lemme 8.7 pour ce cas restreint :

Conséquence 8.8. *Si σ est une permutation en épingles, alors toute permutation associée à un bloc de σ est aussi une permutation en épingles.*

Les permutations en épingles correspondent aux permutations qui sont codées par des mots d'épingles (voir définition 8.11) dans la terminologie de [BHV08a, BRV08]. Dans cet article, R. Brignall, N. Ruškuc et V. Vatter formulent la conjecture suivante :

Conjecture 8.9. *La classe des permutations en épingles a une série génératrice rationnelle.*

Ce chapitre (sauf le paragraphe 8.5) est consacré à la preuve de cette conjecture, qui est obtenue en donnant une formule explicite pour la série génératrice des permutations en épingles.

8.1.2 Représentations en épingles et mots d'épingles

On décrit tout d'abord quelques propriétés générales des représentations en épingles.

Lemme 8.10. *Soit $(p_1, \dots, p_n)$ une représentation en épingles de $\sigma \in \mathcal{S}_n$. Alors pour tout $i \in \{2, \dots, n-1\}$, s'il existe un point x sur les côtés de la boîte englobante de $\{p_1, \dots, p_i\}$, alors il est unique et $x = p_{i+1}$.*

Démonstration. On s'intéresse à la boîte englobante de $\{p_1, p_2, \dots, p_i\}$, et on considère un point x sur les côtés de cette boîte englobante. Sans perte de généralité, on peut supposer que x est au-dessus de la boîte englobante considérée. Par définition d'une boîte englobante, et comme elle contient au moins deux points, x sépare $\{p_1, \dots, p_i\}$ en deux sous-ensembles S_1 et S_2 non vides.

D'autre part, il existe un entier $\ell \geq i$ tel que $x = p_{\ell+1}$. En supposant que $\ell > i$, la boîte englobante de $\{p_1, \dots, p_\ell\}$ contient celle de $\{p_1, \dots, p_i\}$, mais ne contient pas x. Le point x est donc toujours au-dessus de la boîte englobante de $\{p_1, \dots, p_\ell\}$. Par conséquent, x ne satisfaisant pas la condition d'indépendance, il doit satisfaire la condition de séparation : x sépare p_ℓ de $p_1, \dots, p_{\ell-1}$. Or, $S_1, S_2 \subseteq \{p_1, \dots, p_{\ell-1}\}$ et x sépare S_1 de S_2, conduisant à une contradiction. □

Dans [BRV08], R. Brignall, N. Ruškuc et V. Vatter utilisent un codage des représentations en épingles par des mots : les *pin words*. Dans ce codage, à toute représentation en épingles correspond un mot sur l'alphabet $\{1, 2, 3, 4\} \cup \{R, L, U, D\}$, que l'on appellera *mot d'épingles* (en anglais *pin word*), et défini de la façon suivante :

Définition 8.11. Soit $(p_1, p_2, \dots, p_n)$ une représentation en épingles d'une permutation en épingles σ. Pour tout $k \geq 2$, l'épingle p_{k+1} est codée par une lettre selon les règles suivantes :
– si elle sépare p_k de l'ensemble $\{p_1, p_2, \dots, p_{k-1}\}$, alors elle se trouve sur l'un des côtés de la boîte englobante de $\{p_1, p_2, \dots, p_k\}$, et p_{k+1} est codé par L, R, U ou D, selon sa position, comme indiqué sur la figure 8.1,

– si elle satisfait les conditions d'extériorité et d'indépendance, et par là même se trouve dans un des quadrants $1, 2, 3$ ou 4 définis sur la figure 8.1, alors le numéro du quadrant est la lettre codant p_{k+1} dans le mot d'épingles.

Pour le codage de p_1 et p_2, on choisit un point fictif p_0 dans le plan, puis on code p_1 par la lettre (ici, un chiffre entre 1 et 4) correspondant à la position relative de p_1 par rapport à p_0, et enfin, on code p_2 comme les points p_k pour $k \geq 3$, selon sa position par rapport à la boîte englobante de $\{p_0, p_1\}$.

Le choix de p_0 doit être fait de telle sorte qu'aucun point de la permutation σ, sauf p_1, ne soit à l'intérieur de la boîte englobante de $\{p_0, p_1\}$. Même en satisfaisant cette condition, il y a plusieurs choix possibles pour placer p_0 par rapport aux points de σ. Ainsi, il est possible d'avoir plusieurs mots d'épingles codant pour une même représentation en épingles.

Exemple 8.12. Certains des mots d'épingles associés à la représentation en épingles de la permutation $\sigma = 1\,8\,3\,6\,4\,2\,5\,7$ donnée en figure 8.2 sont $11URD3UR, 3RURD3UR, \ldots$

Définition 8.13. Un mot d'épingles $w = w_1 \ldots w_n$ est un *mot d'épingles strict* si et seulement si
– w a une unique lettre qui est un chiffre, et c'est w_1,
– pour tout $i \in [2..(n-1)]$, si $w_i \in \{L, R\}$, alors $w_{i+1} \in \{U, D\}$,
– pour tout $i \in [2..(n-1)]$, si $w_i \in \{U, D\}$, alors $w_{i+1} \in \{L, R\}$.

On peut remarquer que les deux derniers points ci-dessus sont la conséquence directe du fait que w_1 est le seul chiffre dans w. En effet, par définition, dans toute représentation en épingles, il est impossible d'avoir deux épingles séparantes consécutives qui aient la même orientation (verticale ou horizontale).

Un mot d'épingles strict est le codage d'une représentation en épingles propre. Cependant, parmi les mots d'épingles codant une représentation en épingles propre, certains sont stricts et d'autres non. On reviendra plus en détails sur les mots d'épingles au chapitre 9.

Lemme 8.14. *Soit $(p_1, \ldots, p_n)$ une représentation en épingles propre de $\sigma \in \mathcal{S}_n$. Alors, pour tout $i \in [2..n-1]$, l'épingle p_i est à distance 2 exactement de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. La distance est mesurée en nombre de cases sur la représentation graphique de σ entre la frontière de la boîte englobante considérée, et la case où se trouve p_i, cette dernière case comprise.*

Démonstration. La définition 8.4 des représentations en épingles propres assure que pour tout $i \in [2..n-1]$, p_{i+1} sépare p_i de $\{p_1, \ldots, p_{i-1}\}$. Ainsi, p_i est à distance au moins 2 de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. En outre, d'après la définition 8.4 et le lemme 8.10, pour tout $i \in [2..n-1]$, p_i est sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$ et p_{i+1} est le seul point sur les côtés de la boîte englobante de $\{p_1, \ldots, p_i\}$. On en déduit que pour tout $i \in [2..n-1]$, p_i est à distance 2 exactement de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. □

Lemme 8.15. *Soit $p = (p_1, \ldots, p_n)$ une représentation en épingles propre de $\sigma \in \mathcal{S}_n$. Si l'épingle p_i est dans un coin de la boîte englobante de $\{p_1, \ldots, p_j\}$, alors $i = 1$ ou 2.*

Démonstration. Si l'épingle p_i est dans un coin de la boîte englobante de $\{p_1, \ldots, p_j\}$ pour un certain $j \geq i$, alors p_i n'est pas sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. Comme p est une représentation en épingles propre, cela peut se produire seulement lorsque $i = 1$ ou 2. □

8.1.3 Familles des épis et des quasi-épis.

Parmi les permutations en épingles simples, certaines jouent un rôle clé dans la caractérisation des arbres de décomposition des permutations en épingles (voir le théorème 8.25). Il s'agit des permutations que l'on appellera les *épis* et les *quasi-épis*.

Famille des épis

Définition 8.16. La famille des *épis* (ou *weaving permutations*) est une famille de permutations définies par les conditions suivantes. Pour $n \geq 5$, il y a exactement quatre épis de taille n :

– si $n \geq 5$ est pair, un épi de taille n est

$$\sigma = 2\ \underbrace{4\,1}\ \ldots\ \underbrace{(2p+2)\,(2p-1)}\ \ldots\ \underbrace{n\,(n-3)}\ (n-1),$$

– si $n \geq 5$ est impair, un épi de taille n est

$$\sigma = 2\ \underbrace{4\,1}\ \underbrace{6\,3}\ \ldots\ \underbrace{(2p+2)\,(2p-1)}\ \ldots\ \underbrace{(n-1)\,(n-4)}\ n\,(n-2).$$

Cette permutation σ a seulement trois permutations symétriques, obtenues par inverse, miroir et complément (car par exemple $(\sigma^{-1})^r = (\sigma^r)^{-1}$). Ce sont les trois autres épis de taille n.

Il y a en outre un épi de taille 1, deux épis de taille 2 (1 2 et 2 1), quatre épis de taille 3 (1 3 2, 2 1 3, 2 3 1 et 3 1 2) et deux épis de taille 4 (2 4 1 3 et 3 1 4 2).

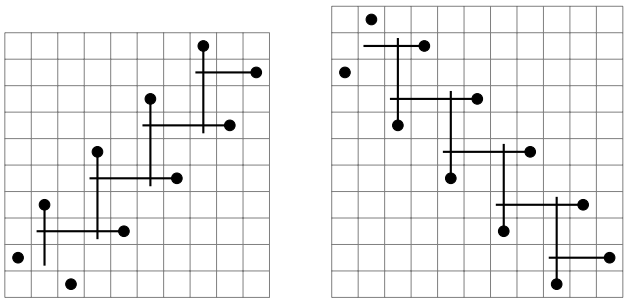
Les propriétés suivantes des épis sont facilement démontrées par un examen exhaustif des différents cas, et sont illustrées sur la figure 8.5.

Lemme 8.17.

(*i*) *Les épis de taille supérieure ou égale à* 4 *sont des permutations en épingles simples.*

(*ii*) *Pour tout épi de taille au moins* 5*, une et une seule des deux diagonales est telle que tout point de la permutation est à distance au plus* 2 *de cette diagonale.*

On peut remarquer que pour $n \leq 4$, les deux diagonales satisfont la condition dans le point (*ii*) du lemme 8.17.



2 4 1 6 3 8 5 10 7 9 9 11 7 10 5 8 3 6 1 4 2

FIG. 8.5 – Un épi montant de taille 10 et un épi descendant de taille 11, chacun donnés avec une représentation en épingles.

Le point (*ii*) du lemme 8.17 autorise la définition de la direction d'un épi :

Définition 8.18. Un épi de taille $n \geq 5$ est dit *montant* lorsque tout point de sa représentation graphique est à distance au plus 2 de la diagonale principale. Sinon, l'épi est dit *descendant*. Pour $n \geq 4$, les épis montants sont 1, 2 1, 2 3 1, 3 1 2, 2 4 1 3 et 3 1 4 2, et les épis descendants sont 1, 1 2, 1 3 2, 2 1 3, 2 4 1 3 et 3 1 4 2. Il faut remarquer au passage que 1, 2 4 1 3 et 3 1 4 2 sont des épis à la fois montants et descendants.

Lemme 8.19. *Un épi montant (resp. descendant) possède des représentations en épingles propres dont les points de départ peuvent être choisis soit dans le coin en bas à gauche, soit dans le coin en haut à droite (resp. soit dans le coin en haut à gauche, soit dans le coin en bas à droite) de sa représentation graphique.*

Démonstration. On prouve le résultat annoncé par vérification exhaustive pour $n \leq 4$. Pour $n \geq 5$, les représentations en épingles propres annoncées sont obtenues comme illustrées sur la figure 8.5. Pour les épis σ de taille respectivement paire et impaire donnés à la définition 8.16, on vérifie que $1RURU\ldots$ et $3LDLD\ldots$ (resp. $1RURU\ldots$ et $3DLDL\ldots$) sont des mots d'épingles codant ces représentations en épingles. Pour les autres épis, on conclut par des arguments de symétrie. □

Famille des quasi-épis

Définition 8.20. On appelle *quasi-épis* (ou *quasi-weaving permutations*) les permutations définies comme suit. Pour $n \geq 6$, il y a exactement huit quasi-épis de taille n.

– Si $n \geq 6$ est pair, un quasi-épi de taille n est

$$\sigma = 2\,n\ \underbrace{4\,1}\ \underbrace{6\,3}\ \ldots\ \underbrace{(2p+2)\,(2p-1)}\ \ldots\ \underbrace{(n-2)\,(n-5)}\,(n-3)\,(n-1).$$

On définit le *point de substitution principal* (resp. *point de substitution auxiliaire*) de σ comme étant le point de coordonnées $(n-1, n-3)$ (resp. $(n, n-1)$) dans la représentation graphique de σ, et le *point extérieur* de σ comme étant le point de coordonnées $(2, n)$ dans sa représentation graphique.

– Si $n \geq 6$ est impair, un quasi-épi de taille n est

$$\sigma = \underbrace{4\,1}\ \underbrace{6\,3}\ \ldots\ \underbrace{(2p+2)\,(2p-1)}\ \ldots\ \underbrace{(n-1)\,(n-4)}\,(n-2)\,n\,2.$$

On définit le *point de substitution principal* (resp. *point de substitution auxiliaire*) de σ comme étant le point de coordonnées $(n-2, n-2)$ (resp. $(n-1, n)$) dans la représentation graphique de σ, et le *point extérieur* de σ comme étant le point de coordonnées $(n, 2)$ dans sa représentation graphique.

Les sept permutations obtenues par symétrie à partir de la permutation σ ci-dessus sont les sept autres quasi-épis de taille n.

Pour $n = 4$ ou 5, il y a deux quasi-épis de taille n : 2 4 1 3, 3 1 4 2, 2 5 3 1 4 et 4 1 3 5 2. Pour chacun d'eux, il y a quatre choix possibles pour la paire de points de substitution principal et auxiliaire. Les détails sont donnés sur la figure 8.7. Les points extérieurs ne sont pas définies pour les quasi-épis de taille inférieure à 6.

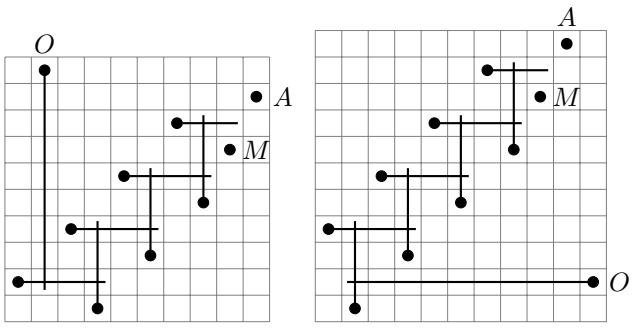


FIG. 8.6 – Deux exemples de quasi-épis de taille 10 et 11. Les points notés M, A et O représentent respectivement les points de substitution principal (*main substitution point*), auxiliaire (*auxiliary substitution point*), et le point extérieur (*outer point*).

On donne à présent quelques propriétés des quasi-épis.

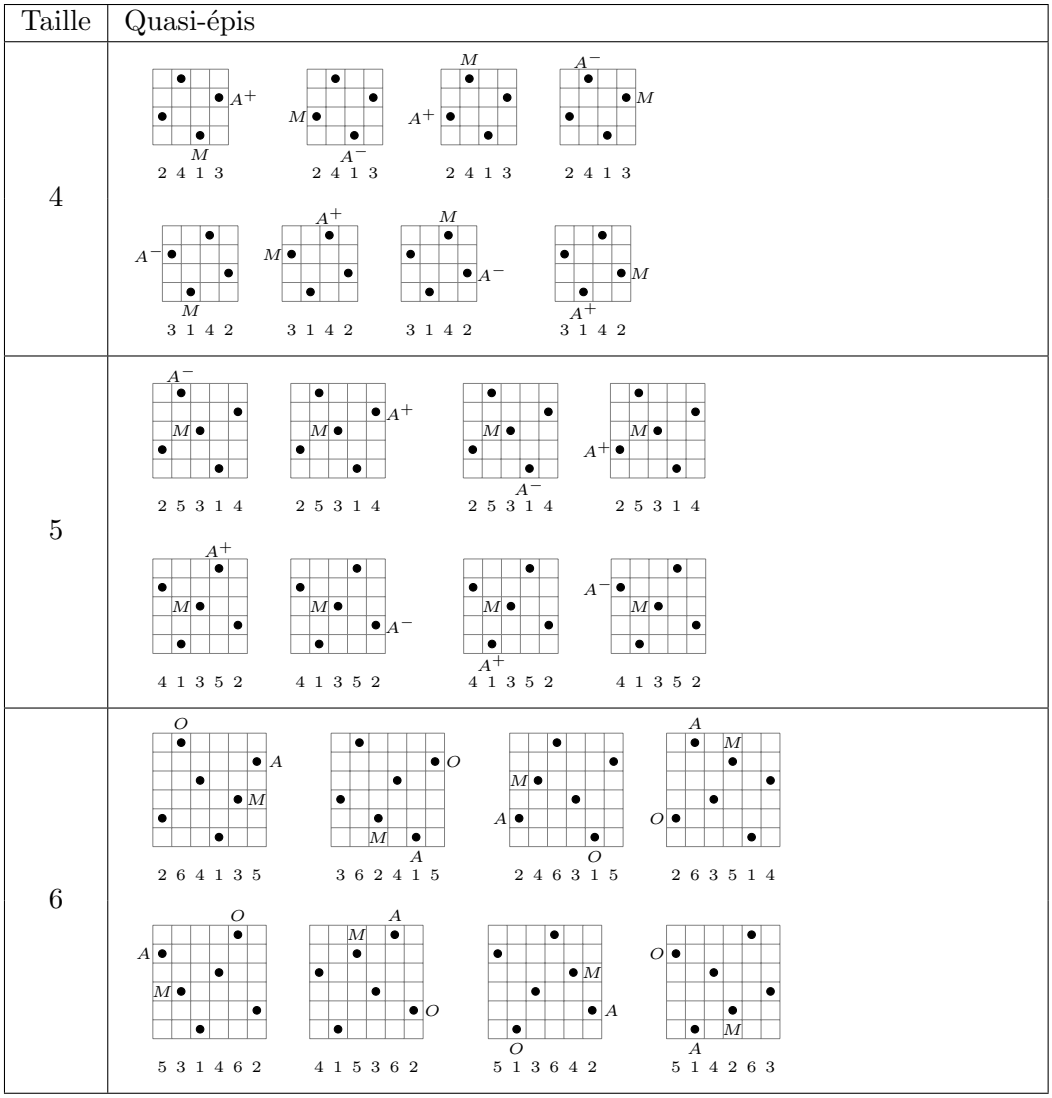


FIG. 8.7 – Représentations graphiques des quasi-épis de taille 4, 5 et 6. Les points notés M, A (ou A^+ ou A^-, voir page 187 pour ces notations) et O représentent respectivement les points de substitution principal, auxiliaire, et le point extérieur (lorsqu'il est défini) de chaque quasi-épi.

Lemme 8.21.

(*i*) *Tout quasi-épi est une permutation en épingles simple.*

(*ii*) *Pour tout quasi-épi de taille au moins* 6*, une et une seule des deux diagonales est telle que tout point de la permutation, excepté le point extérieur, est à distance au plus* 3 *de cette diagonale.*

Démonstration. (*i*) La simplicité des quasi-épis est prouvée par une simple vérification exhaustive. Une représentation en épingles d'un quasi-épi peut être décrit en commençant par son point de substitution principal, en lisant ensuite son point de substitution auxiliaire, et en continuant à parcourir le quasi-épi avec des épingles séparantes, jusqu'à son point extérieur (lorsqu'il est défini).

(*ii*) Sur les quasi-épis σ de taille n donnés à la définition 8.20, on peut vérifier que la différence entre i et σ_i, pour tout $i \in [1..n]$, est au plus 3, sauf pour le point extérieur. Cela prouve que la diagonale principale satisfait la propriété annoncée. Il apparaît aussi clairement que l'autre diagonale ne satisfait pas cette propriété. Pour les autres quasi-épis, on conclut par des arguments de symétrie. □

Le point (*ii*) du lemme 8.21 autorise la définition de la direction d'un quasi-épi.

Définition 8.22. Un quasi-épi de taille $n \geq 6$ est dit *montant* lorsque tout point de sa représentation graphique (sauf le point extérieur) est à distance au plus 3 de la diagonale principale. Sinon, l'autre diagonale satisfait cette propriété, et le quasi-épi est dit *descendant*.

On peut remarquer que la direction de la diagonale du point (*ii*) du lemme 8.21 est la même que celle définie par l'alignement des points M et A de substitution principal et auxiliaire. On peut donc reformuler la définition 8.22, en la généralisant en même temps aux quasi-épis de taille 4 et 5. On rappelle que la définition 8.20 autorise quatre choix possibles pour les points de substitution principal et auxiliaire quand la taille du quasi-épi est 4 ou 5.

Définition 8.23. Un quasi-épi muni d'un choix de points de substitution principal et auxiliaire est dit *montant* (resp. *descendant*) lorsque ces deux points forment une occurrence du motif 1 2 (resp. 2 1) (voir la figure 8.6).

8.2 Caractérisation des arbres de décomposition des permutations en épingles

Comme on l'a vu au théorème 2.44 page 46, les permutations sont en bijection avec les arbres de décomposition. Pour caractériser les permutations en épingles, on donne ici des conditions nécessaires et suffisantes sur les arbres de décomposition pour qu'ils soient associés, par cette bijection, à des permutations en épingles.

Remarque 8.24. Dans tout ce chapitre, lorsqu'on fait référence à un fils V' d'un nœud V dans un arbre de décomposition, on entend le plus souvent le sous-arbre enraciné en V' (ou la permutation correspondante), plutôt que le nœud V' en lui-même.

Théorème 8.25. *Une permutation σ est une permutation en épingles si et seulement si son arbre de décomposition $\mathcal{T}_\sigma$ satisfait les conditions suivantes :*

- (C_1) *tout nœud linéaire de signe $\oplus$ (resp. $\ominus$) dans $\mathcal{T}_\sigma$ a au plus un fils qui n'est pas un épi montant (resp. descendant)*[f],
- (C_2) *tout nœud premier dans $\mathcal{T}_\sigma$ est étiqueté par une permutation en épingles simple α et satisfait l'une des conditions suivantes :*
  - *il a au plus un fils qui n'est pas une feuille ; en outre, le point de α dilaté par l'unique fils non trivial (s'il existe) est un* point actif *de α,*
  - *α est un quasi-épi montant (resp. descendant) et le nœud a exactement deux fils qui ne sont pas des feuilles : l'un dilate le point de substitution principal de α, et l'autre correspond à la permutation* 1 2 *(resp.* 2 1*) et dilate le point de substitution auxiliaire de α.*

Une vision schématique de ce théorème est proposée page 200, pour le calcul de la série génératrice de la classe des permutations en épingles.

8.2.1 Remarques préliminaires

On considère une permutation en épingles σ, qui possède donc une représentation en épingles. Mais tous les points de σ ne sont pas susceptibles d'être le premier point d'une de ses représentations en épingles. Pour cela, on définit :

Définition 8.26. Un *point actif* d'une permutation en épingles σ est un point qui est le premier point d'une des représentations en épingles de σ.

[f] Précisons que les fils de la racine sont ici entendus au sens de la remarque 8.24, et que par conséquent ces épis n'ont aucun de leurs points qui est dilaté.

La remarque suivante sera utilisée à plusieurs reprises dans les démonstrations qui suivent :

Remarque 8.27. On considère une permutation en épingles σ dont l'arbre de décomposition a pour racine un nœud V, et on considère aussi un bloc B de σ correspondant à un fils de V. Si une représentation en épingles de σ vérifie qu'il existe des indices $i < j < k$ tels que $p_i \in B$, $p_j \in B$, et p_k est une épingle séparant p_i de p_j, alors p_k appartient aussi au bloc B.

On suppose que σ est une permutation en épingles, et on considère les nœuds de l'arbre de décomposition $\mathcal{T}_\sigma$ de σ. Ils sont racines de sous-arbres de $\mathcal{T}_\sigma$ correspondant à des permutations qui sont des blocs de σ, et qui sont donc aussi des permutations en épingles. Par conséquent, pour trouver des propriétés de tous les nœuds des arbres de décomposition des permutations en épingles, il suffit de s'intéresser aux propriétés des racines de ces arbres. Avant de s'attaquer à ce problème, on définit deux notions utiles pour décrire le comportement d'une représentation en épingles de σ sur les fils de la racine de $\mathcal{T}_\sigma$.

Définition 8.28. Soit σ une permutation en épingles et $p = (p_1, \dots, p_n)$ une représentation en épingles de σ. Pour tout ensemble B de points de σ, si k est le nombre de facteurs maximaux $p_i, p_{i+1}, \dots, p_{i+j}$ de p qui contiennent exclusivement des points de B, on dit que B est *lu en k fois par p*.

On utilise surtout la définition 8.28 sur les ensembles B qui sont des blocs de σ, et même encore plus précisément sur les blocs correspondant à un fils de la racine dans l'arbre de décomposition de σ.

Définition 8.29. On considère une permutation en épingles σ dont l'arbre de décomposition a pour racine V, et on note $p = (p_1, \dots, p_n)$ une représentation en épingles de σ. On dit qu'un fils B de V est le *k-ème fils lu par p* si, en notant i l'indice minimal tel que p_i appartienne à B, les points $p_1, \dots, p_{i-1}$ appartiennent à exactement $k-1$ fils différents de V.

Exemple 8.30. On considère la permutation $\sigma = 5\,4\,1\,2\,6\,3$. Son arbre de décomposition $\mathcal{T}_\sigma$ est donné sur la figure 8.8. Sa racine V a quatre fils $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ et $\mathcal{T}_4$, de gauche à droite. Pour la représentation en épingles p représentée sur la figure, $\mathcal{T}_2$ est le premier fils de V à être lu par p, puis vient $\mathcal{T}_1$ (par l'épingle p_3, et même si p_6 sera lu plus tard) qui est lu en deux fois par p. On termine enfin par $\mathcal{T}_4$ puis $\mathcal{T}_3$.

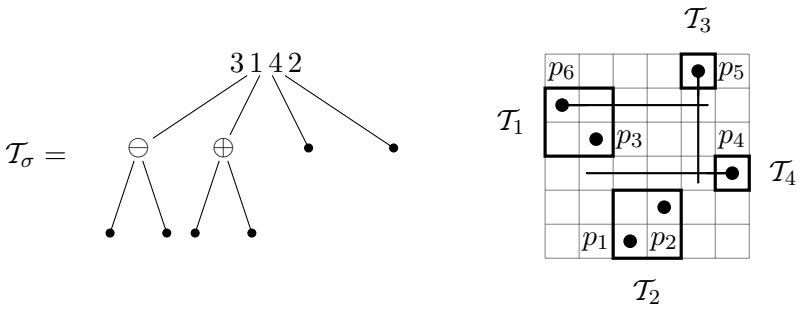


FIG. 8.8 – L'arbre de décomposition $\mathcal{T}_\sigma$ et une représentation en épingles de la permutation $\sigma = 5\,4\,1\,2\,6\,3$.

8.2.2 Propriétés des nœuds linéaires

On analyse d'abord la structure des représentations en épingles des permutations en épingles σ dont l'arbre de décomposition a une racine linéaire V. On démontre dans le lemme 8.31 que certaines représentations en épingles lisent la permutation σ *un fils après l'autre*. Cela nous autorise à donner dans le lemme 8.32 une description précise des fils de V.

Lemme 8.31. *Si σ est une permutation en épingles dont l'arbre de décomposition a une racine linéaire V, alors il existe une représentation en épingles de σ qui lit chacun des fils de V en une fois.*

Démonstration. On suppose que V est de signe $\oplus$, l'autre cas étant similaire. On note $\mathcal{T}_1, \dots, \mathcal{T}_k$ les fils de V, de gauche à droite. Comme σ est une permutation en épingles, il existe une représentation en épingles p de σ. On note i_0 l'indice du fils $\mathcal{T}_i$ auquel le premier point de p appartient.

On suppose maintenant que p est en train de lire à la fois $\mathcal{T}_i$ et $\mathcal{T}_j$ (avec $i < j$), c'est-à-dire que p a lu certains points mais pas tous dans chacun des blocs $\mathcal{T}_i$ et $\mathcal{T}_j$. On considère la boîte englobante B des points déjà lus par p à cet instant : tout fils $\mathcal{T}_\ell$ de V est alors soit complètement inclus dans B, soit complètement à l'extérieur de B. Par conséquent, p a déjà terminé la lecture de chaque $\mathcal{T}_\ell$ pour $i < \ell < j$, et n'a commencé à lire aucun des $\mathcal{T}_\ell$ pour $\ell < i$ ou $\ell > j$. Voir la figure 8.9 pour une vision schématisée de cette affirmation.

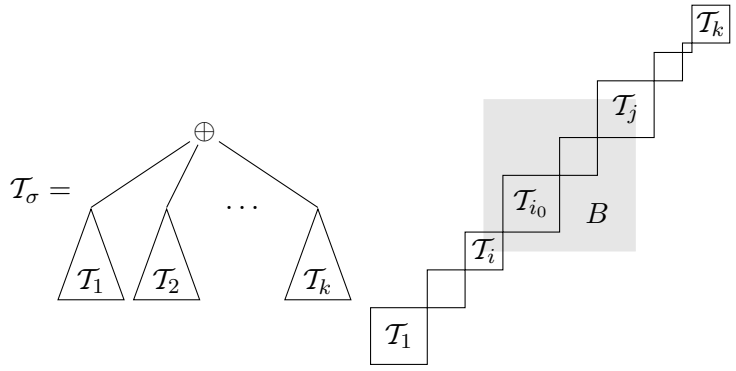


FIG. 8.9 – Une vision graphique de la manière dont une représentation en épingles p parcourt une permutation σ, lorsque la racine de $\mathcal{T}_\sigma$ est un nœud linéaire de signe $\oplus$.

À partir de la représentation en épingles p de σ, on peut donc construire, d'après le lemme 8.7 :
- une représentation en épingles q^{i_0} de $\mathcal{T}_{i_0}$,
- pour tout $i < i_0$ (resp. $i > i_0$), une représentation en épingles q^i de $\mathcal{T}_i$ dont le premier point correspond à une épingle satisfaisant dans p la condition d'indépendance, et qui se trouve dans le coin en bas à gauche (resp. en haut à droite) de la boîte englobante des point déjà lus par p, qui contient au moins un point de $\mathcal{T}_{i_0}$, et tous les points des $\mathcal{T}_\ell$ si $i < \ell < i_0$ (resp. $i_0 < \ell < i$).

On vérifie alors facilement que $q = q^{i_0} q^{i_0-1} \dots q^1 q^{i_0+1} \dots q^k$ est une représentation en épingle de σ, qui lit chaque fils de V en une fois. □

Lemme 8.32. *Soit σ une permutation en épingles dont l'arbre de décomposition a une racine linéaire V de signe $\oplus$ (resp. $\ominus$). Alors au plus un des fils de V n'est pas un épi montant (resp. descendant).*

Démonstration. On suppose que le nœud V est d'étiquette $\oplus$, l'autre cas étant similaire. On note $\mathcal{T}_1, \dots, \mathcal{T}_k$ les fils de V, de gauche à droite. D'après le lemme 8.31, il existe une représentation en épingles p de σ qui lit chacun des fils de V en une fois. On note $\mathcal{T}_{i_0}$ le premier fils de V lu par p.

On suppose que la lecture par p d'un fils $\mathcal{T}_\ell$ (pour $\ell \geq i_0$) de V vient de s'achever. Alors les points de $\mathcal{T}_{\ell+1}$ sont dans le quadrant 1 (en haut à droite) par rapport aux point déjà lus par p (qui sont tous les fils $\mathcal{T}_m, \dots, \mathcal{T}_{i_0}, \dots, \mathcal{T}_\ell$ de V pour un certain $m \leq i_0$, puisque p lit chaque fils de V en une fois). Ainsi, ils correspondent à des épingles codées par les lettres 1, U ou R dans un mot d'épingles. Plus encore, le seul point qui est codé par la lettre 1 est le premier point de $\mathcal{T}_{\ell+1}$ qui est lu par p. En effet, tout autre lettre 1 signifie que p commence la lecture du fils suivant de V,

c'est-à-dire $\mathcal{T}_{\ell+2}$. Par conséquent, $\mathcal{T}_{\ell+1}$ est une permutation représentée par un mot d'épingles de la forme soit $1URURUR\ldots$ soit $1RURURURU\ldots$, c'est-à-dire que $\mathcal{T}_{\ell+1}$ est un épi montant.

De la même manière, on démontre que tout fils $\mathcal{T}_{\ell-1}$ de V, avec $\ell \leq i_0$ est une permutation codée par un mot d'épingles de la forme soit $3LDLDLDLD\ldots$ soit $3DLDLDLDL\ldots$, et qu'ainsi $\mathcal{T}_{\ell-1}$ est aussi un épi montant. En conclusion, le seul fils de V qui est susceptible de ne pas être un épi montant est le premier fils qui est lu par une représentation en épingles de σ. □

Remarque 8.33. Les démonstrations des lemmes 8.31 et 8.32 assurent aussi que si un des fils de V n'est pas un épi montant (resp. descendant), alors toute représentation en épingles de σ commence par lire ce fils (même s'il n'est pas lu complètement).

Démonstration. Soit p une représentation en épingles de σ. On note $\mathcal{T}_{i_0}$ le premier fils de V lu par p. La preuve du lemme 8.31 autorise à construire une autre représentation en épingles p' de σ, dont $\mathcal{T}_{i_0}$ est aussi le premier fils lu, et qui lit chacun des fils de V en une fois. La preuve du lemme 8.32, appliquée à la représentation en épingles p', assure que tous les fils de V sauf peut-être $\mathcal{T}_{i_0}$ sont des épis montants (resp. descendants). □

8.2.3 Propriétés des nœuds premiers

On utilisera souvent la remarque 2.32 dans les preuves de ce paragraphe. Une formulation de cette remarque en termes d'arbres de décomposition est la suivante :

Remarque 8.34. On considère une permutation σ dont l'arbre de décomposition a une racine V qui est un nœud premier. Alors il n'y a aucun bloc dans σ qui intersecte plusieurs fils de V, excepté le bloc trivial σ.

On commence par démontrer un lemme technique :

Lemme 8.35. *Soit σ une permutation en épingles dont l'arbre de décomposition a une racine première V et soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ. Si une épingle p_i qui satisfait les conditions d'extériorité et d'indépendance est le premier point d'un fils B de V à être lu par p, alors B est soit le premier, soit le deuxième fils de V qui est lu par p.*

Démonstration. On suppose que B est un fils de V qui n'est ni le premier ni le deuxième lu par p, et on note p_i le premier point de B qui est lu par p. On note C le fils de V qui est lu (peut-être seulement partiellement) par p juste avant B, et D le fils de V qui est lu par p juste avant C. Comme B est au moins le troisième fils de V lu par p, C et D sont bien définis. Supposons maintenant que p_i satisfasse les conditions d'extériorité et d'indépendance. Alors $\{p_1, \ldots, p_{i-1}\}$ formerait un bloc dans σ qui intersecterait plusieurs fils de V (au moins C et D), mais sans contenir pourtant tous les fils de V (il ne contient pas B). Ceci apporte une contradiction à la remarque 8.34 et conclut la preuve. □

On considère une permutation en épingles σ dont l'arbre de décomposition a une racine première V. Contrairement aux nœuds linéaires, il n'existe pas toujours une représentation en épingles de σ qui lise chaque fils de V en une fois, comme le montre l'exemple de $\sigma = 5\,4\,1\,2\,6\,3$.

Exemple 8.36. On considère la permutation $\sigma = 5\,4\,1\,2\,6\,3$. Son arbre de décomposition $\mathcal{T}_\sigma$ est donné à la figure 8.8. La racine de $\mathcal{T}_\sigma$ est un nœud premier d'étiquette $3\,1\,4\,2$. Il existe deux représentations en épingles pour σ, et toutes deux lisent le fils $\mathcal{T}_1$ de V en deux fois. L'une de ces représentations en épingles est donnée sur la figure 8.8. L'autre est obtenue en échangeant ses deux premiers points.

Cependant, les situations dans lesquelles un fils d'un nœud premier V peut être lu en plusieurs fois sont très contraintes. Le lemme 8.37 et sa conséquence 8.38 identifient ces cas très spécifiques.

Lemme 8.37. *Soit σ une permutation en épingles dont l'arbre de décomposition a une racine première V et soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ. S'il existe un fils B de V que p lit en plus d'une fois, alors la deuxième partie de la lecture de B est réduite à l'épingle p_n.*

Démonstration. On décompose p en $p = (p_1, \ldots, p_i, \ldots, p_j, p_{j+1}, \ldots, p_k, \ldots, p_n)$ où p_i est le premier point de B qui est lui par p, toutes les épingles de p_i à p_j sont des points de B, p_{j+1} n'appartient pas à B, et p_k est le premier point appartenant à B après p_{j+1}. Ces points sont bien définis puisque B est lu par p en plusieurs fois. Pour obtenir le résultat annoncé, il suffit de montrer que $k = n$.

Tout d'abord, pour $k \leq h \leq n$, p_h satisfait les conditions d'extériorité et de séparation. En effet, dans le cas contraire, p_h satisferait les conditions d'extériorité et d'indépendance, et créerait un bloc $p_1, \ldots, p_{k-1}$ dans σ qui intersecterait plusieurs fils de V (plus précisément, B et le bloc auquel appartient p_{j+1}), contredisant la remarque 8.34 puisque V est premier. En outre, on peut montrer par récurrence que $p_h \in B$ pour $k \leq h \leq n$. Cette affirmation est évidemment vraie pour $h = k$. On considère maintenant $h \in \{k+1, \ldots, n\}$. Par hypothèse de récurrence, p_{h-1} appartient à B. Comme p_h satisfait la condition de séparation, il sépare p_{h-1} de $\{p_i, \ldots, p_j\} \subseteq \{p_1, \ldots, p_{h-2}\}$ et par conséquent appartient à B, par application directe de la remarque 8.27. En conclusion, tous les points $p_k, p_{k+1}, \ldots, p_n$ sont des points de B.

D'autre part, on démontre qu'au plus un fils de V est découvert avant l'épingle p_i. En effet, c'est évident dans le cas où $i \leq 2$ et quand $i \geq 3$, on montre que p_i est une épingle qui satisfait les conditions d'extériorité et d'indépendance. À supposer le contraire, comme p_i est le premier point de B qui est lu, tous les points de B seraient alors (comme p_i) sur les côtés de la boîte englobante (non réduite à un point) de $\{p_1, \ldots, p_{i-1}\}$, apportant ainsi une contradiction au lemme 8.10. On conclut par le lemme 8.35 qu'au plus un fils de V apparaît avant B. Par conséquent, comme toute permutation simple est de taille au moins 4 (voir page 40) et comme p se décompose en $p = (\underbrace{p_1, \ldots, p_{i-1}}_{\text{au plus un fils}}, \underbrace{p_i, \ldots, p_j}_{\in B}, \underbrace{p_{j+1}, \ldots, p_{k-1}}_{\notin B}, \underbrace{p_k, \ldots, p_n}_{\in B})$, il y a, parmi $p_{j+1}, \ldots, p_{k-1}$, des points appartenant à au moins deux fils différents de V, tous deux différents de B. On note C le fils de V auquel p_{k-1} appartient, et D un autre fils de V qui apparaît dans $p_{j+1}, \ldots, p_{k-1}$. Comme p_k sépare p_{k-1} des épingles précédentes, et comme p_k appartient à B, B (par p_k) sépare C (auquel appartient p_{k-1}) de D (qui contient une autre épingle précédant p_{k-1}). Mais alors tout point de B qui n'a pas encore été lu, c'est-à-dire tout point de $\{p_k, \ldots, p_n\}$, est sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{k-1}\}$. Or le lemme 8.10 assure qu'il y a au plus un point sur les côtés d'une boîte englobante. Ceci démontre que $k = n$. □

Conséquence 8.38.

(*i*) *Si un fils d'un nœud premier est lu en plus d'une fois par une représentation en épingles p, alors il est lu en exactement deux fois, la seconde partie étant réduite à la dernière épingle de p.*

(*ii*) *Au plus un fils d'un nœud premier peut être lu en deux fois par une représentation en épingles.*

On sait maintenant, étant donnée une permutation en épingles σ dont l'arbre de décomposition $\mathcal{T}_\sigma$ a une racine première V, comment toute représentation en épingles de σ parcourt les fils de V. Le lemme 8.39 et sa conséquence 8.40 s'intéressent à la structure interne de ces fils.

Lemme 8.39. *Soit σ une permutation en épingles dont l'arbre de décomposition a une racine première V et soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ. On suppose qu'il existe un fils B de V qui n'est pas réduit à une permutation de taille 1, et tel que B n'est pas le premier fils de V lu par p. Alors B est une permutation de taille 2, qui est lue en deux fois par p, et telle que la première épingle de p qui lit un point de B satisfasse les conditions d'extériorité et d'indépendance.*

Démonstration. Dans la représentation en épingles p de σ, on note p_i le premier point de B qui est lu par p. Par hypothèse, $i \geq 2$ et $i \neq n$. On suppose que p_i est une épingle séparante. Alors, nécessairement $i \geq 3$ (il est impossible que p_i sépare un ensemble de moins de deux points), et p_i est sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. Mais comme p_i est le premier point de B qui est lu, alors tout autre point de B est aussi sur les côtés de cette boîte englobante. B n'étant pas une permutation de taille 1, ceci apporte une contradiction au lemme 8.10. Par conséquent, p_i satisfait les conditions d'extériorité et d'indépendance. D'après le lemme 8.35, et comme on a supposé que B n'est pas le premier fils lu, B est donc le deuxième fils de V lu par p. On note C le premier fils de V lu par p. Comme V est un nœud premier, il doit exister un point dans σ, qui appartienne à un autre fils D de V, et qui sépare B de C. Ce point sépare en particulier p_i de p_1, et est donc nécessairement p_{i+1}, puisqu'aucune autre épingle après p_{i+1} ne pourra séparer p_1 de p_i. Ainsi, $p_{i+1} \notin B$. La conséquence 8.38(i) assure alors que soit $B = \{p_i\}$, soit $B = \{p_i, p_n\}$. Comme B n'est pas une permutation de taille n, on est dans le deuxième cas, ce qui termine la démonstration. □

Avec les résultats du lemme 8.35 et de la conséquence 8.38, on peut déduire du lemme 8.39 que le premier fils de V qui est lu par p est lu en une fois, que B est le deuxième fils lu par p, et que p_n est le second point de B.

Conséquence 8.40. *Un nœud premier a au plus deux fils qui ne sont pas des permutations de taille* 1*, c'est-à-dire des feuilles. En outre, si un nœud premier a exactement deux fils non réduits à des feuilles, alors toute représentation en épingles de la permutation en épingles σ correspondante commence par lire entièrement l'un des fils non feuilles, et l'autre fils non feuille est celui caractérisé par le lemme 8.39.*

Démonstration. Le premier fils de V à être lu par une représentation en épingles p peut être de taille supérieure à 1. Tout autre fils (noté B) de V qui ne serait pas réduit à une feuille doit satisfaire le lemme 8.39. En particulier, la dernière épingle de p appartient à B. Ceci implique qu'il y a au plus un tel fils non feuille.

Pour prouver la deuxième partie de l'énoncé, on suppose que V a exactement deux fils non feuilles, et qu'il existe une représentation en épingles p de σ qui ne commence pas par l'un des fils non feuilles de V. D'après le lemme 8.39, et comme la dernière épingle de p appartient à un seul fils de V exactement, on obtient une contradiction. Ainsi, p commence par lire l'un des fils non feuilles de V, et l'autre fils non feuille correspond au B du lemme 8.39. En particulier, B étant lu en deux fois, le fils non feuille de V qui est lu en premier par p est lu en une seule fois. □

8.2.4 Preuve de la condition nécessaire

Avec les lemmes précédents, on peut maintenant démontrer dans ce paragraphe que les conditions (C_1) et (C_2) du théorème 8.25 sont des conditions nécessaires sur l'arbre de décomposition $\mathcal{T}_\sigma$ de σ pour que σ soit une permutation en épingles.

Soit σ une permutation en épingles dont l'arbre de décomposition $\mathcal{T}_\sigma$ est représenté en figure 8.10. Comme on l'a expliqué plus haut, tout nœud V de $\mathcal{T}_\sigma$ est la racine d'un sous-arbre $\mathcal{T}$ de $\mathcal{T}_\sigma$, et $\mathcal{T}$ est en fait l'arbre de décomposition $\mathcal{T}_\pi$ d'une permutation $\pi \preceq \sigma$. Par la conséquence 8.8, π est aussi une permutation en épingles, de sorte qu'il suffit de démontrer que, pour toute permutation π dont la racine de l'arbre de décomposition est un nœud V,

– si V est un nœud linéaire, alors la condition (C_1) est satisfaite par V,
– si V est un nœud premier, la condition (C_2) est satisfaite par V.

Lorsque V est un nœud linéaire, on conclut par le lemme 8.32. On suppose donc que V est un nœud premier, étiqueté par une permutation simple α. Le lemme 8.7 assure que α est une permutation en épingles, puisque c'est un motif de π.

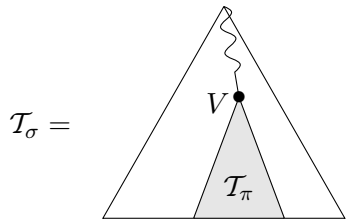


FIG. 8.10 – Tout nœud de l'arbre de décomposition $\mathcal{T}_\sigma$ d'une permutation σ est la racine de l'arbre de décomposition $\mathcal{T}_\pi$ d'un motif π de σ.

D'après la conséquence 8.40, V a au plus 2 fils qui ne sont pas des feuilles. On suppose d'abord que V a exactement un fils B qui n'est pas une feuille, et on considère une représentation en épingles $p = (p_1, \ldots, p_n)$ de π. Il faut démontrer que ce fils B dilate un point actif de α. Si p commence par la lecture de B (même partielle), alors il y a une occurrence de α dans π où B est représenté par le premier point p_1 de p, et avec le lemme 8.7, on peut extraire de p une représentation en épingles pour α dont le premier point, qui est par la définition 8.26 un point actif de α, est celui qui représente B. Lorsqu'au contraire B n'est pas le premier fils lu par p, on applique le lemme 8.39 : B contient exactement deux points, le premier étant p_2 (lu juste après le premier fils lu par p – qui est par hypothèse une permutation de taille 1, et dont la lecture est donc réduite à p_1) et le second p_n. En remarquant que les deux premiers points dans une représentation en épingles jouent des rôles symétriques (voir remarque 8.3), l'ordre dans lequel ils sont lus importe peu : par conséquent, $p_2, p_1, p_3, \ldots, p_n$ est une autre représentation en épingles admissible pour π et une occurrence de α dans π est formée de tous les points de p sauf p_n. Ainsi, $p_2, p_1, p_3, \ldots, p_{n-1}$ est une représentation en épingles de α où B est représenté par p_2, et ainsi B dilate un point actif de α.

On suppose à présent que le nœud V a exactement deux fils qui ne sont pas réduits à des feuilles. La conséquence 8.40 montre que toute représentation en épingles $p = \{p_1, \ldots, p_n\}$ de π est faite ainsi :

- p lit entièrement l'un des fils non feuille de V, que l'on note C, l'autre fils non feuille noté B contenant exactement deux points,
- le premier point de B lu par p satisfait les conditions d'extériorité et d'indépendance,
- le second point de B lu par p est p_n.

Sans perte de généralité, c'est-à-dire à symétrie près, on peut supposer que B est dans la zone en haut à droite de C. Cette situation est représentée sur la figure 8.11.

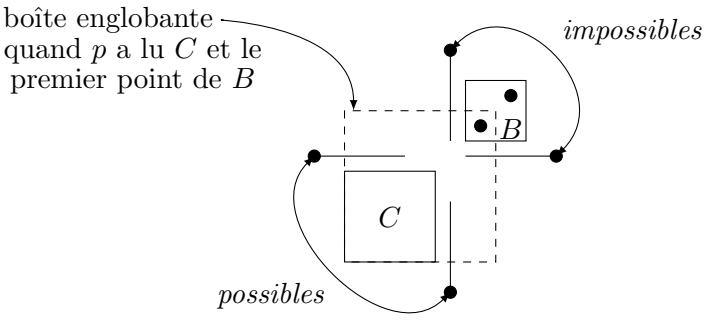


FIG. 8.11 – La permutation π autour de ses deux fils non feuilles B et C.

Si le bloc B contenait la permutation 2 1, alors le deuxième point de B serait sur les côtés de la boîte englobante quand p a lu C et le premier point de B, et d'après le lemme 8.10, ce second point

de B devrait être lu juste après le premier, contredisant que le nœud V est premier (et a donc au moins 4 fils). Ainsi, B contient la permutation 1 2.

Entre les deux points de B, p lit tous les points de π qui correspondent aux fils de V qui sont des feuilles. Comme V est premier, par la remarque 8.34, tous ces points sont des épingles séparantes, et il y a au moins deux tels points. Il y a quatre positions possibles pour la première épingle séparante de cet ensemble lue par p, mais seulement deux peuvent être choisies, puisqu'il est nécessaire que la permutation α soit simple. En effet, le choix d'une des épingles de direction U ou R sur la figure 8.11 (les épingles indiquées comme *impossibles*) impliquerait que le second point de B est sur les côtés de la boîte englobante, et doit donc être lu immédiatement, et comme ce point est la dernière épingle de p, la représentation en épingles s'arrêterait, contredisant que le nœud V est premier. On peut donc supposer que la première épingle lue par p après le premier point de B est celle dans la zone en haut à gauche de C, l'autre épingle possible conduisant à une configuration symétrique. Par le même raisonnement, la représentation en épingles p est alors une alternance d'épingles de type D et L, jusqu'à p_{n-1} qui est de type U ou R.

Par conséquent, pour ne par contredire que la permutation α est simple et que B contient deux points, il y a une unique façon de placer les épingles qui correspondent aux fils de V qui sont des feuilles. Cette configuration est représentée sur la figure 8.12, et correspond au cas où α est un quasi-épi, avec C dilatant son point de substitution principal, et B dilatant son point de substitution auxiliaire. En outre, lorsque la taille de α est au moins 6, si le quasi-épi est montant (resp. descendant), alors B contient la permutation 1 2 (resp. 2 1). La justification de ce dernier point réside dans le fait que l'alignement des blocs B et C définit une direction qui est celle du quasi-épi.

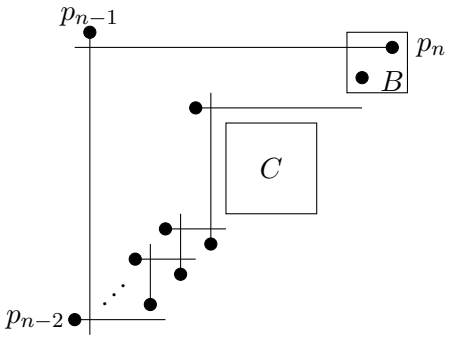


FIG. 8.12 – La seule permutation en épingles (à symétrie près) dont l'arbre de décomposition a une racine première (d'arité au moins 6) possédant deux fils qui ne sont pas des feuilles.

On peut remarquer que dans le cas d'un quasi-épi de taille 4 ou 5, dont les points de substitution principal et auxiliaire sont fixés, le contenu du bloc B est aussi déterminé, toujours par la direction qu'indique l'alignement des blocs B et C. La preuve de ce résultat est omise ici, mais procède par simple examen de chacun de 16 cas possibles. Sur la figure 8.7 (page 179), les points de substitution auxiliaires notés A^+ (resp. A^-) sont ceux qui peuvent être dilatés par 1 2 (resp. 2 1) dans ce contexte.

Ceci achève la preuve que les conditions (C_1) et (C_2) sont des conditions nécessaires sur une permutation σ pour que σ soit une permutation en épingles.

Remarque 8.41. Le raisonnement ci-dessus démontre que si σ est une permutation en épingles de racine V dont deux fils ne sont pas des feuilles, alors la permutation qui étiquette V est un quasi-épi β, et les deux fils non feuilles de V dilatent une paire de points de substitution (principal et auxiliaire) de β.

Si $|\beta| \geq 6$, alors cette paire est uniquement déterminée, et les points de substitution principal

et auxiliaire sont distingués au sein de cette paire. Si $|\beta| = 4$ ou 5, il y a quatre paires de points de substitution, mais dans chaque paire, les points de substitution principal et auxiliaire sont distingués (voir figure 8.7).

Parmi les deux fils de V qui sont dilatés, on peut donc distinguer le fils C qui dilate le point de substitution principal et le fils B qui dilate le point de substitution auxiliaire. La démonstration qui précède (associée aux lemmes 8.39 et 8.35) assure que dans une représentation en épingles de σ, toutes les épingles en dehors du fils C sont complètement déterminées.

Ainsi, toute représentation en épingles de σ commence par une représentation en épingles de C, puis lit un point de B, puis lit tous les fils de V qui sont des feuilles, et enfin termine par le point restant dans B.

8.2.5 Preuve de la condition suffisante

On peut maintenant achever la démonstration du théorème 8.25 en montrant que les conditions (C_1) et (C_2) sont suffisantes pour qu'une permutation σ soit une permutation en épingles. On prouve ici par récurrence sur la taille de σ qu'une permutation satisfaisant les conditions (C_1) et (C_2) est une permutation en épingles. On rappelle que $\mathcal{T}_\sigma$ désigne l'arbre de décomposition de σ.

On remarque d'abord que pour $\sigma = 1$, les conditions (C_1) et (C_2) sont vraies par vacuité, et la représentation en épingles à une unique épingle est une représentation en épingles de σ.

On suppose donc que $|\sigma| > 1$, et que pour toute permutation π de taille inférieure à $|\sigma|$, si π satisfait les conditions (C_1) et (C_2) alors π est une permutation en épingles. On distingue deux cas, selon que la racine de $\mathcal{T}_\sigma$ est un nœud linéaire ou un nœud premier.

Lorsque la racine de $\mathcal{T}_\sigma$ est un nœud linéaire, on note $\sigma = \oplus[\sigma^1, \sigma^2, \dots, \sigma^k]$, sa décomposition par substitution, le cas où $\sigma = \ominus[\sigma^1, \sigma^2, \dots, \sigma^k]$ étant symétrique. On suppose évidemment que σ satisfait les conditions (C_1) et (C_2). Comme les arbres de décomposition des $(\sigma^i)_{1\le i\le k}$ sont des sous-arbres de $\mathcal{T}_\sigma$, chacune des permutations $(\sigma^i)_{1\le i\le k}$ satisfait aussi les conditions (C_1) et (C_2). Par hypothèse de récurrence, chaque $(\sigma^i)_{1\le i\le k}$ est donc une permutation en épingles. En outre, la condition (C_1) est valable pour la racine de $\mathcal{T}_\sigma$, de sorte qu'au plus une des permutations $(\sigma^i)_{1\le i\le k}$ n'est pas un épi montant. On note i_0 l'indice tel que σ^{i_0} n'est pas un épi montant, si cet indice existe. Sinon, on choisit une valeur $i_0 \in [1..k]$. Comme σ^{i_0} est une permutation en épingles, elle admet une représentation en épingles p^{i_0}. D'après le lemme 8.19, pour tout $i < i_0$ (resp. $i > i_0$), il existe une représentation en épingles p^i de σ^i (qui est un épi montant) dont l'origine est située en haut à droite (resp. en bas à gauche). On considère alors $p = p^{i_0} p^{i_0-1} \dots p^1 p^{i_0+1} \dots p^k$: p est une représentation en épingles de σ. On peut remarquer que d'autres représentations en épingles p de σ auraient pu être construites de façon similaire : celles de la forme $p = p^{i_0} w$ où w est n'importe quel *shuffle* de $p^{i_0-1} \dots p^1$ et $p^{i_0+1} \dots p^k$.

Lorsque la racine de $\mathcal{T}_\sigma$ est un nœud premier, on note $\sigma = \alpha[\sigma^1, \sigma^2, \dots, \sigma^k]$ la décomposition par substitution de σ. On suppose aussi que σ satisfait les conditions (C_1) et (C_2). Comme dans le cas précédent, par hypothèse de récurrence, les permutations $(\sigma^i)_{1\le i\le k}$ sont toutes des permutations en épingles. On note p^i une représentation en épingles de chaque σ^i. On rappelle que par définition, σ^i dilate le point α_i de la permutation simple α étiquetant la racine de $\mathcal{T}_\sigma$. En appliquant la condition (C_2) à la racine de $\mathcal{T}_\sigma$, on déduit que α est aussi une permutation en épingles, et qu'au plus deux des permutations $\sigma^1, \sigma^2, \dots, \sigma^k$ ne sont pas de taille 1.

Si toutes les permutations $\sigma^1, \sigma^2, \dots, \sigma^k$ sont de taille 1, alors $\sigma = \alpha$, de sorte que σ est une permutation en épingles. Si une unique permutation σ^i est de taille strictement supérieure à 1, la condition (C_2) assure que σ^i dilate un point actif de α. Ainsi, il existe une représentation en épingles p de α où $p_1 = \alpha_i$. Une représentation en épingles de σ est obtenue en remplaçant dans p l'épingle p_1 par la représentation en épingles p^i de σ^i.

Lorsque deux des permutations $\sigma^1, \sigma^2, \dots, \sigma^k$ sont de taille strictement supérieure à 1, alors α est un quasi-épi, et on peut supposer sans perte de généralité que c'est un quasi-épi montant. Parmi les deux fils de α qui ne sont pas des feuilles, l'un (disons σ^i) dilate le point de substitution principal

α_i de α, et l'autre (disons σ^j) est la permutation 1 2 qui dilate le point de substitution auxiliaire α_j de α. On note p représentation en épingles de α dont la première épingle p_1 correspond à son point de substitution principal, et la deuxième p_2 correspond à son point de substitution auxiliaire. Pour obtenir une représentation en épingles de σ, on commence par effacer la première épingle de p et on la remplace par une représentation en épingles p^i de σ^i. On remplace ensuite p_2 par le point de σ^j qui est le plus proche du bloc σ^i. Comme les deux points qui dilatent α_j suivent la direction définie par l'alignement des points de substitution principal et auxiliaire de α (voir figure 8.12), on peut définir cette notion de point de σ^j le plus proche du bloc σ^i (le bloc qui dilate le point de substitution principal de α). On continue ensuite la lecture de tous les points de p et on termine enfin par le point restant de σ^j, qui est alors une épingle séparant le dernier point lu par p (le point extérieur lorsque $|\alpha| \geq 6$) des précédents.

Dans tous les cas, on obtient bien une représentation en épingles de σ, démontrant ainsi que σ est une permutation en épingles. Ceci achève de prouver que les conditions (C_1) et (C_2) sont suffisantes pour qu'une permutation soit une permutation en épingles.

Dans le paragraphe 8.4, on répond à la conjecture 8.9 de R. Brignall, N. Ruškuc et V. Vatter dans [BRV08] : on calcule la série génératrice de la classe des permutations en épingles, prouvant ainsi qu'elle est rationnelle. La démonstration de ce résultat utilise la caractérisation des arbres de décomposition des permutations en épingles du théorème 8.25, et des techniques standards de combinatoire analytique (brièvement décrites dans le chapitre 10 en annexe ; pour plus de détails sur ce thème, on pourra consulter l'ouvrage de référence [FS08]).

Cependant, cela requiert de calculer au départ la série génératrice des permutations en épingles simples. On s'intéresse à cette question dans le paragraphe 8.3.

8.3 Série génératrice des permutations en épingles simples

On introduit ici un peu de terminologie supplémentaire, adaptée au paragraphe qui suit.

Définition 8.42. Une représentation en épingles $p = (p_1, p_2, \ldots, p_n)$ et un mot d'épingles $w = w_1 w_2 \ldots w_n$ sont dits *simples* lorsque la permutation σ qu'ils codent est une permutation simple.

On remarque qu'une représentation en épingles *simple* est toujours *propre* (ce qui n'est pas le cas des mots d'épingles, la deuxième lettre d'un mot d'épingles simple pouvant être un chiffre). Cependant, il existe des représentations en épingles propres (et des mots d'épingles stricts) qui codent des permutations qui ne sont pas simples.

On s'intéresse d'abord (paragraphe 8.3.2) à la caractérisation des représentations en épingles simples, dans le but de les énumérer. Ceci nous permettra de calculer dans un deuxième temps (paragraphe 8.3.3) la série génératrice des permutations en épingles simples. Bien que la correspondance entre représentations en épingles simples et permutations en épingles simples ne soit pas univoque, on peut calculer le nombre de représentations en épingles simples associées à une permutation en épingles simple, et ainsi déduire de la série génératrice des représentations en épingles simples celle des permutations en épingles simples.

Pour les besoins de cette étude, on donne dans le paragraphe 8.3.1 des propriétés importantes des deux premiers points des représentations en épingles propres.

8.3.1 Propriétés des premiers points d'une représentation en épingles d'une permutation en épingles simple

Définition 8.43. On considère une permutation σ donnée par sa représentation graphique. On dit qu'une paire de points $\{x, y\}$ est un cavalier lorsque la distance de la case contenant x (exclue) à la case contenant y (incluse) est exactement 3, sans que x et y soient dans la même ligne ou dans la même colonne (voir figure 8.13).

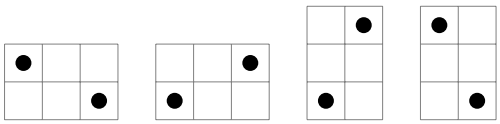


FIG. 8.13 – Les quatre configurations possibles de paires de points formant un cavalier.

Lemme 8.44. *On note $p = (p_1, p_2, \ldots, p_n)$ une représentation en épingles propre d'une permutation σ. Si $|\sigma| > 2$ les deux premières épingles p_1 et p_2 forment un cavalier dans σ.*

Démonstration. Par définition des représentations en épingles propres, et comme $|\sigma| > 2$, p_3 sépare p_2 de p_1, et aucune épingle après p_3 ne pourra séparer p_2 de p_1. Ainsi, p_1 et p_2 sont séparés uniquement par p_3, de sorte à former un cavalier. □

Lemme 8.45. *Soit σ une permutation en épingles simple, et $p = (p_1, p_2, \ldots, p_n)$ une de ses représentations en épingles simples. Si deux points p_i et p_j de σ forment un cavalier, alors i ou j est égal à $1, 2$ ou n.*

Démonstration. On rappelle d'abord que d'après la remarque 8.34, comme σ est une permutation simple, toute épingle p_i ($i \geq 3$) sépare p_{i-1} de $\{p_1, \ldots, p_{i-2}\}$. On considère une épingle p_i. On cherche tous les points p_j, avec $j < i$, tels que $\{p_i, p_j\}$ soit un cavalier dans σ. On veut montrer que pour tout j, $\{i, j\} \cap \{1, 2, n\} \neq \emptyset$. On suppose donc $i \geq 3$ et $i < n$, le résultat étant évident pour $i = 1, 2$ ou n. Sans perte de généralité, on suppose que l'épingle p_i est située au-dessus de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$, et que p_{i-1} se trouve à droite de p_i, comme illustré sur la figure 8.14. Le rectangle en trait gras représente la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$.

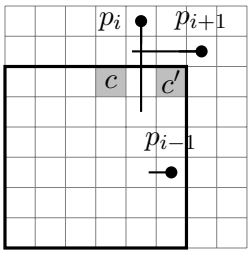


FIG. 8.14 – Représentations en épingles jusqu'à une épingle p_i, pour $i < n$, dans la démonstration du lemme 8.45.

Comme $i < n$, p_{i+1} sépare p_i des épingles précédentes, et comme p_i est de type U, p_{i+1} est soit de type L soit de type R, comme représenté sur la figure 8.14. Par conséquent, p_i ne peut former un cavalier avec une épingle p_j précédente que si p_j se trouve dans une des cases c ou c' indiquées sur la figure.

Il y a une épingle dans la case c' : Seul le point p_{i-1} peut se trouver dans la case c' et dans ce cas, cela implique que p_{i-1}, qui se trouve dans le coin d'une boîte englobante, est soit p_1 soit p_2 (d'après le lemme 8.15). Les possibles cavaliers ainsi mis en évidence sont donc les paires de points $\{p_1, p_2\}$ et $\{p_2, p_3\}$.

Il y a une épingle dans la case c : Dans ce cas, on démontre que la case c contient soit p_1 soit p_2. En effet, en supposant le contraire, l'épingle p_j dans la case c séparerait verticalement deux épingles d'indice inférieur à j, appartenant à la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$. Or le seul point de la permutation à droite de la case c et à l'intérieur de cette boîte englobante est p_{i-1} (la colonne entre c et c' étant occupée par p_i), ce qui apporte une contradiction. Les paires des points pouvant donner lieu à un cavalier dans cette situation sont donc $\{p_1, p_i\}$ et $\{p_2, p_i\}$.

Dans tous les cas, on a bien, $\{i, j\} \cap \{1, 2, n\} \neq \emptyset$. □

Définition 8.46. Un cavalier *actif* dans une permutation en épingles σ est une paire de points $\{x, y\}$ formant un cavalier dans σ, et telle qu'il existe une représentation en épingles de σ dont les deux premiers points sont x et y.

En conséquence du lemme 8.44, le nombre de représentations en épingles d'une permutation simple dépend du nombre de ses cavaliers actifs.

Lemme 8.47. *Dans toute permutation en épingles simple σ, il y a au plus deux cavaliers actifs, sauf pour les quatre permutations suivantes, qui ont quatre cavaliers actifs :* 3 1 4 2, 2 4 1 3, 2 5 3 1 4 *et* 4 1 3 5 2. *Les permutations en épingles simples de taille au plus* 6 *et leur cavaliers actifs sont représentés sur la figure 8.15.*

Pour $n > 6$, toutes les permutations en épingles simples de taille n ont exactement un cavalier actif, sauf douze d'entre elles, qui ont deux cavaliers actifs. Il s'agit des quatre épis de taille n et des huit quasi-épis de taille n.

Démonstration. Les résultats de la figure 8.15 sont simplement obtenus en examinant tous les cas.

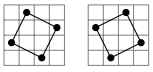
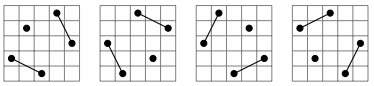
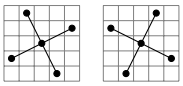
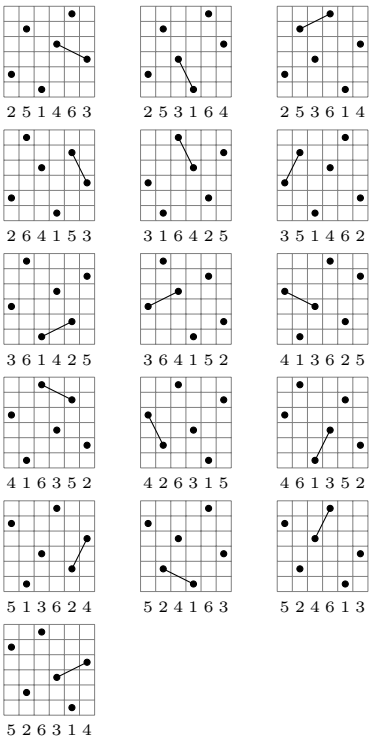
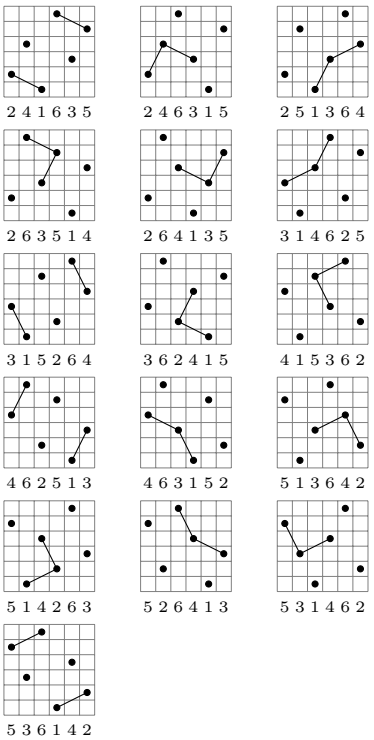
n	1 cavalier actif	2 cavaliers actifs	4 cavaliers actifs
4			 2 4 1 3 3 1 4 2
5		 2 4 1 5 3 3 1 5 2 4 3 5 1 4 2 4 2 5 1 3	 2 5 3 1 4 4 1 3 5 2
6	 2 5 1 4 6 3 2 5 3 1 6 4 2 5 3 6 1 4 2 6 4 1 5 3 3 1 6 4 2 5 3 5 1 4 6 2 3 6 1 4 2 5 3 6 4 1 5 2 4 1 6 3 5 2 4 2 6 3 1 5 4 6 1 3 5 2 5 1 3 6 2 4 5 2 4 1 6 3 5 2 4 6 1 3 5 2 6 3 1 4	 2 4 1 6 3 5 2 4 6 3 1 5 2 5 1 3 6 4 2 6 3 5 1 4 2 6 4 1 3 5 3 1 4 6 2 5 3 1 5 2 6 4 3 6 2 4 1 5 4 1 5 3 6 2 4 6 2 5 1 3 4 6 3 1 5 2 5 1 3 6 4 2 5 1 4 2 6 3 5 2 6 4 1 3 5 3 1 4 6 2 5 3 6 1 4 2	

FIG. 8.15 – Les permutations en épingles simples de taille $n \leq 6$ et leurs cavaliers actifs.

Soit σ une permutation en épingles simple de taille $n > 6$ et soit $p = (p_1, p_2, \ldots, p_n)$ une représentation en épingles de σ. D'après le lemme 8.44, la paire de points $\{p_1, p_2\}$ est un cavalier actif de σ. On veut démontrer que toute permutation en épingles ayant au moins deux cavaliers actifs

est un épi ou un quasi-épi. Il est ensuite facile de vérifier que les épis et quasi-épis ont exactement deux cavaliers actifs (voir un exemple sur la figure 8.16).

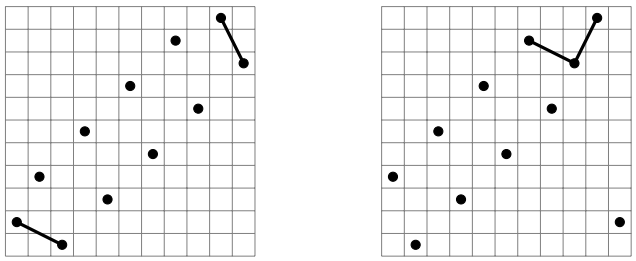


FIG. 8.16 – Un épi et un quasi-épi dont les cavaliers actifs sont marqués.

On suppose donc que σ possède au moins cavaliers actifs, l'un d'eux étant nécessairement $\{p_1, p_2\}$. Le lemme 8.45 assure que le deuxième cavalier actif (choisi parmi ceux qui restent) est de la forme $\{p_1, p_i\}, \{p_2, p_i\}$ ou $\{p_i, p_n\}$, pour un certain indice i.

Le deuxième cavalier actif est $\{p_1, p_i\}$. Sans perte de généralité, on considère p_1 et p_2 sont positionnés l'un par rapport à l'autre comme représenté sur la figure 8.17. Il y a alors 7 cases différentes où peut se trouver le point p_i formant un cavalier avec p_1. Ces cases sont numérotées de 1 à 7 sur la figure 8.17. Les cases 7 et 3 doivent être vides, car elles ont en commun une colonne ou une ligne avec le point p_2. Une épingle dans la case 1 crée un bloc avec l'épingle p_2 : σ étant simple, la case 1 est donc vide. Ainsi, les seules cases qui peuvent être occupées par p_i sont celles numérotées 2, 4, 5 et 6.

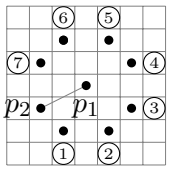


FIG. 8.17 – Cavaliers $\{p_1, p_2\}$ et $\{p_1, p_i\}$.

– L'épingle p_i est dans la case 5. On note r une représentation en épingles de σ commençant par les points $\{p_1, p_i\}$. (Cette représentation en épingles existe car $\{p_1, p_i\} = \{r_1, r_2\}$ est un cavalier actif de σ.) Alors r_3 se trouve sur la ligne entre r_1 et r_2. D'après le lemme 8.14, r_3 est à distance 2 de la boîte englobante de $\{r_1, r_2\}$. Il ne peut pas être à gauche de cette boîte englobante, ou il serait dans la même colonne que p_2. Ainsi, r_3 est à sa droite, comme illustré dans la première configuration de la figure 8.18. Pour la même raison, p_3 est en-dessous de la boîte englobante de $\{p_1, p_2\}$ comme le montre le premier schéma de la figure 8.18.

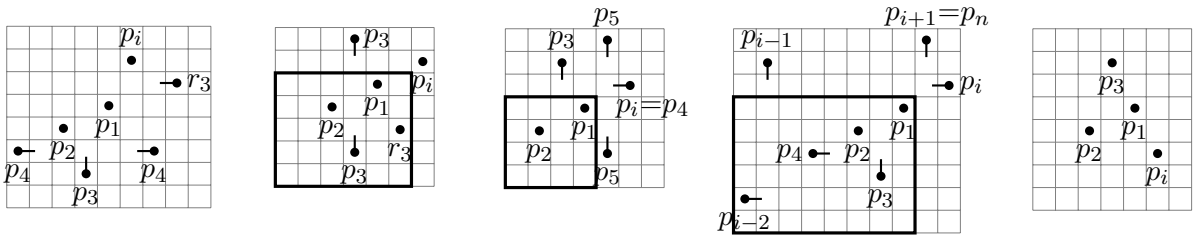


FIG. 8.18 – Les différents cas pour le cavalier actif $\{p_1, p_i\}$.

Il y a alors deux cases possibles pour p_4. Cette épingle se trouve sur la ligne entre p_2 et p_3, à distance 2 de la boîte englobante de $\{p_1, p_2, p_3\}$, soit sur sa gauche, soit sur sa droite (voir sur

la figure 8.18, premier schéma). Si elle est à droite, alors les six points $\{p_1, p_2, p_3, p_4, p_i, r_3\}$ forment une permutation de taille 6, ou un intervalle. Ceci contredit que σ est simple et de taille $n > 6$. Donc p_4 est à gauche de la boîte englobante. La seule manière de poursuivre la représentation en épingles p est alors en alternant les épingles de type L et D, jusqu'à une épingle de type R ou U. Cette épingle ne peut pas être de type U, puisque la ligne au-dessus et à distance 2 de la boîte englobante est occupée par p_i. Si c'est une épingle de type R, elle se trouve dans la colonne entre $p_i = r_2$ et r_3, et doit donc être r_4. Dans ce cas, dans la représentation en épingles r, l'épingle r_4 ne satisfait pas la condition de distance 2 par rapport à la boîte englobante de $\{r_1, r_2, r_3\}$, ce qui contredit le lemme 8.14.

– L'épingle p_i est dans la case 4. Alors $i \geq 4$ et la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$ est en bas à gauche de p_i. Elle peut ou non contenir la colonne située juste à droite de p_1.
On suppose d'abord que cette colonne est dans la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$, comme le montre le deuxième schéma de la figure 8.18. Il existe alors un point dans la boîte englobante et dans cette colonne. Ce point séparant $p_1 = r_1$ de $p_i = r_2$ doit être r_3, et se trouve donc deux lignes en-dessous de p_1. Quant à l'épingle p_3, elle se trouve dans la colonne entre p_1 et p_2, et à distance 2 de la boîte englobante de $\{p_1, p_2\}$. Si elle se trouve au-dessus, l'ensemble $\{p_1, p_2, p_3, p_i, r_3\}$ forme un intervalle dans σ, et si elle se trouve en-dessous, c'est $\{p_1, p_2, p_3, r_3\}$ qui forme un intervalle, contredisant dans les deux cas que σ est une permutation simple de taille supérieure à 6.
On se place maintenant dans le cas où la colonne juste à droite de p_1 n'appartient pas à la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$. Comme précédemment, p_3 se trouve dans la colonne entre p_1 et p_2, et est à distance 2 de la boîte englobante de $\{p_1, p_2\}$, soit au-dessus, soit en-dessous. On suppose d'abord que p_3 est au-dessus, comme dans le troisième schéma de la figure 8.18. Alors $p_i = p_4$. On s'intéresse donc à p_5, qui sépare $p_4 = p_i = r_2$ de $\{p_1, p_2, p_3\}$, donc en particulier sépare r_2 de $p_1 = r_1$. On déduit donc que $p_5 = r_3$ est à distance 2 de la boîte englobante de $\{p_1, p_4\}$. Tout choix pour p_5 crée alors un intervalle de taille 5 dans σ, apportant une contradiction.
Si p_3 est en-dessous de la boîte englobante de $\{p_1, p_2\}$ (quatrième schéma de la figure 8.18), alors à partir de p_3, p est une alternance d'épingles de type L et D, jusqu'à p_{i-2} (ici, $i-2 \geq 4$). Alors p_{i-1} est un épingle de type U ou R que p_i sépare de $\{p_1, \ldots, p_{i-2}\}$. En supposant que p_{i-1} est de type R, elle doit être à distance 2 de la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$, et donc dans la même colonne que p_i ce qui est impossible. Donc p_{i-1} est de type U, et se trouve dans la ligne juste au-dessus de p_i pour respecter la condition de distance 2. p continue ensuite avec p_i qui est une épingle de type R. Alors p_{i+1} sépare en particulier $p_1 = r_1$ de $p_i = r_2$, de sorte que $p_{i+1} = r_3$ se trouve dans la colonne entre p_1 et p_i, et deux lignes au-dessus de p_i (la ligne à distance 2 en-dessous de p_1 est déjà occupée par p_4). Cela signifie que p_{i+1} est à distance 1 de la boîte englobante de $\{p_1, \ldots, p_i\}$, et donc que $i+1 = n$. Dans la représentation en épingles r, on a donc $r_1 = p_1$, $r_2 = p_i$, donc nécessairement $r_3 = p_{i+1}$ et $r_4 = p_{i-1}$. Il y a donc plusieurs points (au moins p_2 et p_3) sur les côtés de la boîte englobante de $\{r_1, r_2, r_3, r_4\}$ apportant une contradiction au lemme 8.10.

– L'épingle p_i est dans la case 2. L'épingle p_3 se trouve dans la colonne entre p_1 et p_2 à distance 2 de la boîte englobante de $\{p_1, p_2\}$. Elle ne peut pas se trouver en-dessous de cette boîte englobante ou sinon elle créerait un intervalle avec p_1, p_2, p_i. Elle se trouve donc au-dessus, comme dans le dernier schéma de la figure 8.18. On peut alors être en présence d'une permutation en épingles faite d'une alternance d'épingles de type L et U, jusqu'à une épingle p_k de type R ou D. Alors p_i sépare cette épingle p_k des précédentes, et on a $p_i = p_{k+1}$. L'ensemble $\{p_1, \ldots, p_{k+1}\}$ forme alors un intervalle, et donc $i = k + 1 = n$. On vérifie alors facilement qu'on se trouve en présence d'une permutation simple σ qui est un quasi-épi, et a donc deux cavaliers actifs.

– L'épingle p_i est dans la case 6. Elle sépare verticalement p_1 de p_2 et donc $i = 3$. Alors, p_4

est soit sur la gauche, soit sur la droite de la boîte englobante de $\{p_1, p_2, p_3\}$. Mais une autre représentation en épingles de σ commence par p_1 and p_3, puisque ces deux points forment un cavalier actif, et p_4 doit donc aussi être à distance 2 de la boîte englobante de $\{p_1, p_3\}$. En choisissant p_4 à gauche, on crée un intervalle $\{p_1, p_2, p_3, p_4\}$, donc p_4 se trouve à droite de cette boîte englobante, en dans la case numérotée 4 sur la figure 8.17. Dans la représentation en épingles commençant par p_1, p_3, p_4, on peut ensuite avoir une alternance d'épingles de type U et R, jusqu'à ce qu'une épingle de type L ou D permette de lire le point p_2 (comme dans le cas où p_i se trouve dans la case 2). Dans ce cas, on obtient encore que σ est un quasi-épi.

Le deuxième cavalier actif est $\{p_2, p_i\}$. Sachant que $(p_2, p_1, p_3, \ldots, p_n)$ est une autre représentation en épingles de σ, ce cas est résolu par le précédent.

Le deuxième cavalier actif est $\{p_i, p_n\}$. On suppose d'abord que $i \geq 4$. On considère alors la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$. Sans perte de généralité, on suppose que p_{i-1} est au-dessus de cette boîte englobante, et que p_i est une épingle de type R, comme le montre la figure 8.19. On

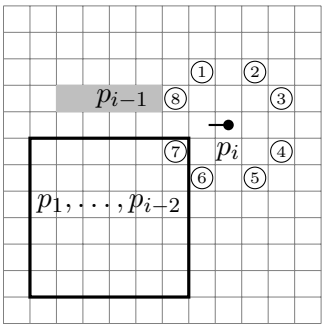


FIG. 8.19 – Cas du cavalier actif $\{p_i, p_n\}$, avec $i \geq 4$.

remarque que comme p_{i-1} est de type U, dans la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$, il y a un point dans chaque ligne, et dans chaque colonne, sauf dans celle de p_{i-1}. Comme p_n et p_i forment un cavalier actif, p_n doit se trouver dans l'une des huit cases numérotées sur la figure 8.19. Mais les cases 3, 4, 5, 6, 8 doivent rester vides, puisque leurs lignes sont déjà occupées par d'autres points. p_n ne peut pas non plus se trouver dans la case 7, puisqu'il doit être à l'extérieur de la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$. Si p_n est dans la case 2, alors la représentation en épingles r, qui commence la lecture de σ par $r_1 = p_i, r_2 = p_n$, continue ensuite par $r_3 = p_{i-1}$ et donc p_{i-1} doit se trouver à distance 2 de la boîte englobante de $\{p_i, p_n\}$ c'est-à-dire dans la colonne la plus à droite de la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$, ce qui est impossible. On en déduit que p_n est dans la case 1. Comme précédemment, $r_3 = p_{i-1}$. Ainsi, r_4 est une épingle de type D, r_5 est de type L, et r est une alternance d'épingles de type D et L. Le choix d'une épingle d'un autre type la placerait sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{i-2}\}$, contredisant le lemme 8.10. On conclut dans ce cas que σ est un épi.

On suppose à présent que $i < 4$. Les cas $i = 1$ et $i = 2$ ont déjà été considérés dans les cas précédents. Si $i = 3$, on peut supposer sans perte de généralité que p_1, p_2 et p_3 sont dans la configuration représentée sur le premier schéma de la figure 8.20.

Alors p_n se trouve dans l'une des sept cases numérotées de 1 à 7 sur la figure 8.20. Les cases 1, 4 et 5 doivent rester vides, puisqu'un autre point se trouve dans leur colonne. Si p_n se trouvait dans une des cases 2 ou 7, on créerait un intervalle de taille 4 dans σ, ce qui est impossible. Ainsi, p_n se trouve en case 3 ou 6. On suppose que p_n est dans la case 6, le cas de la case 3 pouvant être traité de façon similaire. L'épingle p_4 se trouve alors dans la ligne entre p_2 et p_3, à distance 2 de la boîte englobante de $\{p_1, p_2, p_3\}$. Si elle est à droite de cette boîte englobante, alors $p_5 = p_n$, formant un intervalle de taille 5 dans σ, de sorte que p_4 doit être à sa gauche (voir le deuxième schéma de la figure 8.20). Alors l'épingle p_5 se trouve dans la colonne entre p_4 et p_1, à distance 2 de la boîte

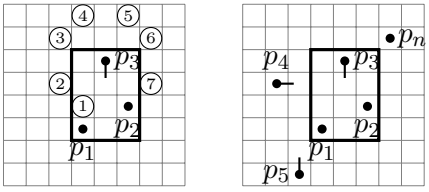


FIG. 8.20 – Cas du cavalier actif $\{p_3, p_n\}$.

englobante de $\{p_1, p_2, p_3, p_4\}$. Si p_5 est au-dessus, alors elle se trouve dans la case juste au-dessus de la case 3, et $p_6 = p_n$, créant un intervalle de taille 6 dans σ. Donc p_5 est en-dessous de cette la boîte englobante considérée, comme le montre le deuxième schéma de la figure 8.20. On considère à présent la représentation en épingles r de σ qui commence par $r_1 = p_3, r_2 = p_n$. Nécessairement, $r_3 = p_2$, et $r_4 = p_4$. Mais alors p_1 et p_5 sont tous deux sur les côtés de la boîte englobante de $\{r_1, r_2, r_3, r_4\}$, apportant une contradiction au lemme 8.10. □

Une conséquence du lemme 8.47 qui sera utile au paragraphe 8.3.3 est la suivante :

Lemme 8.48. *Pour tout $n > 6$, il y a 4 permutations en épingles simples de taille n possédant 4 points actifs, 8 qui en possèdent 3, et toutes les autres ont 2 points actifs. Lorsque $n \leq 6$, on a (voir la figure 8.15) :*

- *taille 4 : 2 permutations avec chacune 4 points actifs,*
- *taille 5 : 2 permutations avec 5 points actifs, et 4 qui en possèdent 4,*
- *taille 6 : 4 permutations avec 4 points actifs, 12 qui en possèdent 3, et 16 qui en possèdent 2.*

Démonstration. D'après les définitions 8.26 et 8.46, et en utilisant le lemme 8.44, on déduit facilement que les points actifs d'une permutation en épingles simple σ sont les points appartenant à un cavalier actif de σ. La figure 8.15 donne alors le résultat pour $n \leq 6$. Pour $n > 6$, il suffit de remarquer que les deux cavaliers actifs d'un quasi-épi ont un point en commun, alors qu'ils sont disjoints dans un épi. Le lemme 8.47 donne alors le résultat. □

On termine ce paragraphe par une remarque établissant un lien entre le nombre de représentations en épingles simples et le nombre de permutations en épingles simples.

Remarque 8.49. Étant donnée une permutation en épingles simple σ dont un cavalier actif est marqué, il y a une unique représentation en épingles p de σ (l'unicité étant à échange de p_1 et p_2 près) qui commence par le cavalier actif marqué. Ceci résulte du fait que chaque épingle p_i de p pour $i \geq 3$ est séparante (ou alors on créerait un intervalle non trivial dans σ qui est simple), et donc que p_i est déterminée comme l'unique épingle sur les côtés de la boîte englobante de $\{p_1, \ldots, p_{i-1}\}$.

Dans le paragraphe 8.3.3, on utilise cette remarque pour déduire de l'énumération des représentations en épingles simples celle des permutations en épingles simples.

8.3.2 Énumération des représentations en épingles simples

Comme on l'a remarqué plus haut (page 189), il existe des représentations en épingles propres qui ne sont pas simples. Dans [BHV08a], le théorème 3.4 (une fois reformulé avec la terminologie employée dans cette thèse) énonce que toute représentation en épingles propre est *presque* simple, au sens où pour toute représentation en épingles propre $p = (p_1, p_2, \ldots, p_n)$, soit p, soit $(p_2, p_3, \ldots, p_n)$, soit $(p_1, p_3, \ldots, p_n)$ est une représentation en épingles simple. On raffinant la preuve de ce théorème, on peut démontrer que *presque toutes* les représentations en épingles propres sont simples, et caractériser celles qui ne le sont pas. Les étapes de la démonstration sont très semblables, mais le résultat obtenu est d'une nature différente.

On a remarqué aussi (remarque 8.3) que pour toute représentation en épingles propre $p = (p_1, p_2, p_3, \ldots, p_n)$, alors $p^\sim = (p_2, p_1, p_3, \ldots, p_n)$ est aussi une représentation en épingles propre. Ces deux objets représentent en fait la même chose : on choisit d'abord une paire de points $\{p_1, p_2\}$, puis une épingle p_3 qui sépare p_1 et p_2, et on continue ainsi en insérant des épingles séparantes à chaque étape. C'est pourquoi, dans toutes les considérations énumératives qui vont suivre, on comptera les deux représentations en épingles p et $p^\sim$ comme **un seul et unique objet**.

Pour obtenir les résultats d'énumération attendus, on utilisera parfois des mots d'épingles simples plutôt que des représentations en épingles simples. Le lemme 8.50 montre que c'est équivalent, à un facteur multiplicatif 4 près. Le théorème 8.53 énumère les représentations en épingles propres qui ne sont pas simples. On obtient ensuite facilement l'énumération des représentations en épingles simples, donnée par le théorème 8.55.

Lemme 8.50. *Pour toute représentation en épingles propre p de taille au moins 3, il y a exactement 4 mots d'épingles stricts qui codent p (sans contrainte d'ordre entre p_1 et p_2). Ces 4 mots d'épingles stricts correspondent aux 4 lectures possibles du cavalier actif $\{p_1, p_2\}$.*

Démonstration. Il y a 9 positions possibles de l'origine (p_0) par rapport à p_1 et p_2, parmi lesquelles 4 seulement donnent lieu à un mot d'épingles strict. Il s'agit des positions $1, 2, 8$ et 9 de la figure 8.21, correspondant respectivement aux préfixes $4R, 3R, 1L$ et $2L$ d'un mot d'épingles strict.

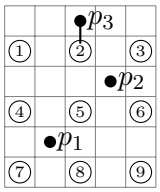


FIG. 8.21 – Parmi les 9 positions possibles du point fictif servant d'origine au codage d'une représentation en épingles propre p par un mot d'épingles, 4 seulement donnent lieu à des mots d'épingles stricts.

Dans les positions 1 et 2 (resp. 8 et 9), la lecture des deux premiers points se fait dans l'ordre p_1 puis p_2 (resp. p_2 puis p_1). Il faut bien remarquer que lorsque l'origine est en 1 (resp. 9), et la lecture de p_2 puis p_1 (resp. p_1 puis p_2) n'est pas admissible, car elle induirait deux épingles de même orientation (ici, verticale) consécutives, ce qui est par définition interdit.

Pour les autres positions possibles de l'origine, et quelque soit l'ordre choisi pour p_1 et p_2 (lorsque les deux ordres possibles sont admissibles), les deux premières lettres du mot sont des chiffres. □

Lemme 8.51. *Pour tout $n \geq 3$ il y a 2^n représentations en épingles propres de taille n.*

Démonstration. On montre qu'il y a 2^{n+2} mots d'épingles stricts de taille n, le lemme 8.50 donnant alors le résultat directement. Il y a 4 possibilités pour la première lettre d'un mot d'épingles strict ($1, 2, 3$ ou 4), 4 encore pour la deuxième lettre (U, D, L ou R), et à partir de la troisième lettre, il n'y a plus que deux choix admissibles, qui dépendent de la lettre précédente dans le mot (seuls U et D peuvent apparaître après L ou R, et inversement). On obtient donc 2^{n+2} mots d'épingles stricts de taille n, ce qui termine la démonstration. □

La démonstration du théorème 8.53 suit les mêmes étapes que celle du théorème 3.4 de [BHV08a]. Le lemme 8.52 est démontré dans [BHV08a] (où il apparaît comme le lemme 3.3) :

Lemme 8.52. *Dans une représentation en épingles propre $p = (p_1, p_2, \ldots, p_n)$, pour tout $i \geq 3$, p_i et p_{i+1} sont séparés soit par p_{i-1} soit par chaque point de l'ensemble $\{p_1, \ldots, p_{i-2}\}$.*

Théorème 8.53. *Pour tout $n \geq 5$, il y a 16 représentations en épingles propres de taille n qui ne sont pas simples. Elles correspondent aux permutations suivantes :*

– *les* 8 *quasi-épis de taille* $n-1$ *dont le point de substitution auxiliaire est dilaté par* 1 2 *ou* 2 1 *(selon que le quasi-épi est montant ou descendant),*
– *les* 8 *permutations obtenues à partir des* 4 *épis de taille* $n-1$ *en ajoutant un élément à leur représentation graphique, dans l'un des deux coins définissant la diagonale qui correspond à la direction de l'épi.*

Pour $n = 4$*, seules* 8 *représentations en épingles propres ne sont pas simples. Elles correspondent aux* 8 *permutations du deuxième point ci-dessus.*

La figure 8.22 montre les deux types de représentations en épingles propres qui ne sont pas simples.

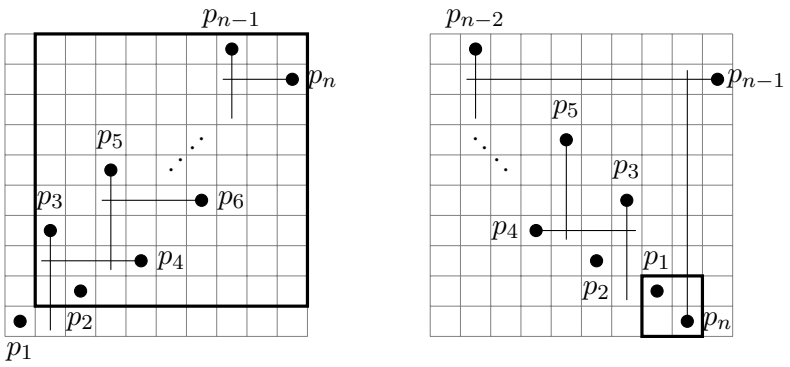


FIG. 8.22 – Les deux types de permutations qui ne sont pas simples mais admettent des représentations en épingles propres.

Démonstration. La démonstration du théorème 8.53 utilise à plusieurs reprises la remarque suivante, dont la preuve, immédiate, est omise.

Remarque 8.54. Si B est un bloc d'une permutation σ, et si x et y sont deux points de B, alors tout point de σ qui sépare x et y appartient aussi au bloc B.

On considère donc une représentation en épingles propre $p = (p_1, \ldots, p_n)$ (avec $n \geq 4$) qui code une permutation σ, et on suppose que σ n'est pas simple. Il existe alors un intervalle non trivial dans σ. On choisit un tel intervalle $M \subseteq \{p_1, \ldots, p_n\}$, qui soit minimal au sens de l'inclusion (M ne contient aucun intervalle, si ce n'est des singletons). M étant de taille au moins 2, on peut choisir deux indices i et j, avec $i < j$ tels que $\{p_i, p_j\} \subseteq M$. On fait ce choix de telle sorte que j soit minimal parmi toutes les valeurs possibles.

Si $i < j < n$, alors $\{p_{j+1}, \ldots, p_n\} \subseteq M$, puisque toutes ces épingles séparent deux point appartenant à M. Avec le lemme 8.52, et comme j est minimal, on déduit que $\{p_1, \ldots, p_{j-2}\} \subseteq M$, sauf si $i = j-1$. Dans ce dernier cas, le lemme 8.52 et la minimalité de j assurent que $i \leq 2$, et on reviendra sur ce cas ensuite. On se concentre d'abord sur la situation où $\{p_1, \ldots, p_{j-2}\} \subseteq M$. Si $j \geq 4$, p_{j-1} sépare deux points de M, et donc appartient aussi à M. Ainsi, $M = \{p_1, p_2, \ldots, p_n\}$, ce qui est impossible. Si $j \leq 2$, On obtient aussi que $M = \{p_1, p_2, \ldots, p_n\}$, c'est-à-dire qu'on est face à la même contradiction. Enfin, si $j = 3$, on peut supposer sans perte de généralité que $i = 2$ et $p_1 \notin M$. Toutes les épingles à partir de p_4 séparent deux points de M donc appartiennent aussi à M. Mais comme $p_1 \notin M$, il ne peut exister aucun indice k tel que p_1 soit sur les côtés de la boîte englobante de $\{p_2, \ldots, p_k\}$. Cette condition force M à représenter un épi, et p_1 à se trouver dans un coin de la boîte englobante de M, juste à côté de p_2 et p_3. Cette situation est représentée sur la première partie de la figure 8.22.

Il reste donc à considérer les cas où $i = j-1$ pour $i = 1$ ou 2. Si $i = 1$, alors $\{p_1, p_2\} \subseteq M$, et on obtient par récurrence que pour tout $k \geq 3, p_k \in M$, puisque c'est une épingle séparant deux

points de M. Ainsi, $M = \{p_1, p_2, \ldots, p_n\}$, ce qui est comme précédemment une contradiction. Si $i = 2$, alors $\{p_2, p_3\} \subseteq M$, et on obtient de la même manière que $\{p_2, p_3, \ldots, p_n\} \subseteq M$. Le point p_1 n'est pas dans M, sinon la minimalité de j serait mise en défaut. Par conséquent, on retrouve la situation illustrée par le premier schéma de la figure 8.22 : M est un épi et p_1 est dans un coin de M, tout proche de p_2 et p_3.

Si au contraire $j = n$, alors par minimalité de j, $M = \{p_i, p_n\}$. Dans le cas $i = n - 1$, le lemme 8.52 apporte une contradiction. Si $3 \leq i \leq n - 2$, alors p_i sépare $\{p_1, \ldots, p_{i-1}\}$, alors que p_n ne peut pas séparer ces points, ce qui est à nouveau une contradiction. Ainsi, $i = 1$ ou 2. Sans perte de généralité, on suppose que $i = 1$, c'est-à-dire que $M = \{p_1, p_n\}$. Il est alors impossible que $\{p_2, p_n\}$ soit un intervalle : p_3 séparant p_1 de p_2 doit aussi séparer p_n de p_2. Par conséquent, on peut supposer que $M = \{p_1, p_n\}$ est le seul intervalle non trivial de σ, ou la question aurait été résolu par un des cas précédents. Cela implique en particulier que la représentation graphique de $\{p_2, \ldots, p_n\}$ est celle d'une permutations simple, et donc que $n \geq 5$. Sans restreindre le propos, on peut faire l'hypothèse que p_1 et p_n sont en ordre décroissant, dans une lecture de gauche à droite, comme le montre le deuxième schéma de la figure 8.22. Alors p_n est une épingle soit de type R soit de type D. On supposera qu'elle est de type D, ce qui n'est pas une restriction, à symétrie près. Cela force toutes les épingles de p_2 à p_{n-2} à être en haut à gauche de p_1, et p_{n-1} à être une épingle de type R. Nécessairement, les épingles de p_3 à p_{n-2} sont de type L et U en alternance, de sorte que σ est un quasi-épi où le point de substitution auxiliaire est dilaté, par 2 1 dans le cas représenté sur la figure 8.22. De manière générale, ce point est dilaté par 1 2 ou 2 1, et la distinction entre ces deux cas se fait par le sens (montant ou descendant) du quasi-épi. □

Théorème 8.55. *Pour tout $n \geq 5$ il y a $2^n - 16$ représentations en épingles simples de taille n, il y en a 8 de taille 4, et aucune de taille strictement inférieure. La série génératrice de l'ensemble des représentations en épingles simples est donc $SiRep(z) = 8z^4 + \frac{32z^5}{1-2z} - \frac{16z^5}{1-z}$.*

Démonstration. Le premier point est une conséquence immédiate du lemme 8.51 et du théorème 8.53. Le second point est obtenu par un calcul élémentaire. □

8.3.3 Énumération des permutations en épingles simples

On rappelle que les représentations en épingles simples sont en bijection avec les permutations en épingles simples ayant un cavalier actif marqué (voir la remarque 8.49). L'énumération donnée par le théorème 8.55 peut donc aussi être vue comme celle des permutations en épingles simples, où chaque permutation est comptée avec une multiplicité égale à son nombre de cavaliers actifs. Ce n'est pas exactement cette série génératrice avec multiplicité qui sera utile pour l'énumération des permutations en épingles au paragraphe 8.4. Cependant, elle autorise à calculer deux séries génératrices qui serviront ce but : la série génératrice des permutations en épingles simples (sans multiplicité), et la série génératrice des permutations en épingles simples avec une multiplicité égale au nombre de leurs points actifs.

Théorème 8.56. *La série génératrice des permutations en épingles simples (sans multiplicité) est $Si(z) = 2z^4 + 6z^5 + 32z^6 + \frac{128z^7}{1-2z} - \frac{28z^7}{1-z}$.*

Théorème 8.57. *La série génératrice des permutations en épingles simples, comptées avec une multiplicité égale à leur nombre de points actifs, est $SiMult(z) = 8z^4 + 26z^5 + 84z^6 + \frac{256z^7}{1-2z} - \frac{40z^7}{1-z}$.*

Démonstrations des théorèmes 8.56 et 8.57. Le théorème 8.55 et le lemme 8.47 assurent que :

- pour la taille $n = 4$, il y a 2 permutations en épingles simples, chacune d'elles ayant 4 cavaliers actifs,
- pour la taille $n = 5$, il y a 2 permutations en épingles simples avec 4 cavaliers actifs, et 4 avec 2 cavaliers actifs,

– pour la taille $n = 6$, il y a 16 permutations en épingles simples avec 2 cavaliers actifs, et 16 avec 1 cavalier actif,
– pour toute taille $n \geq 7$, il y a 12 permutations en épingles simples avec 2 cavaliers actifs et $2^n - 16 - 2 \times 12 = 2^n - 40$ permutations en épingles simples avec 1 cavalier actif.

Le dernier point utilise la remarque 8.49 : les 12 permutations en épingles simples avec 2 cavaliers actifs comptent pour un total de 2×12 représentations en épingles simples.

On obtient donc

$$\begin{aligned} Si(z) &= 2z^4 + (2+4)z^5 + (16+16)z^6 + \sum_{n\geq 7}(12 + 2^n - 40)z^n \\ &= 2z^4 + 6z^5 + 32z^6 + \sum_{n\geq 7}(2^n - 28)z^n \end{aligned}$$

ce qui termine, après quelques calculs sans difficulté, la démonstration du théorème 8.56.

Pour le théorème 8.57, il faut étudier le nombre de points actifs dans toute permutation en épingles simple. Avec le lemme 8.48, on obtient immédiatement que

$$\begin{aligned} SiMult(z) &= (2 \times 4)z^4 + (2 \times 5 + 4 \times 4)z^5 + (4 \times 4 + 12 \times 3 + 16 \times 2)z^6 \\ &\quad + \sum_{n\geq 7}(4 \times 4 + 8 \times 3 + (2^n - 40) \times 2)z^n \\ &= 8z^4 + 26z^5 + 84z^6 + \sum_{n\geq 7}(2^{n+1} - 40)z^n \end{aligned}$$

ce qui achève la démonstration du théorème 8.57. □

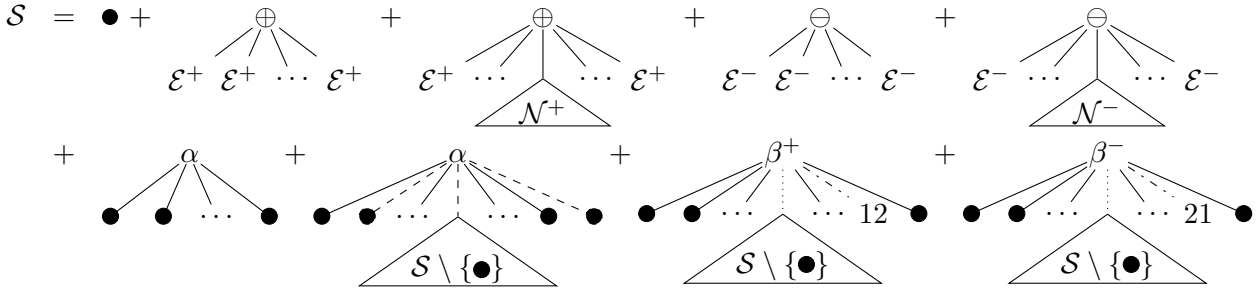
8.4 Série génératrice de la classe des permutations en épingles

Dans ce paragraphe, on calcule la série génératrice de la classe des permutations en épingles. Plus exactement, on calcule la série génératrice des arbres de décomposition associés aux permutations en épingles, ce qui, d'après le théorème 2.44 (page 46), est équivalent.

8.4.1 Une équation définissant les arbres de décomposition des permutations en épingles

On note $\mathcal{S}$ l'ensemble des arbres de décomposition des permutations en épingles. On note aussi $\mathcal{E}^+$ (resp. $\mathcal{E}^-$) l'ensemble des arbres de décomposition des épis montants (resp. descendants), et $\mathcal{N}^+$ (resp. $\mathcal{N}^-$) l'ensemble des arbres de décomposition des permutations en épingles qui ne sont pas des épis montants (resp. descendants) et dont la racine n'est pas un nœud $\oplus$ (resp. $\ominus$). L'ensemble $\mathcal{N}^+$ (resp. $\mathcal{N}^-$) correspond donc aux arbres qui ne représentent pas des épis montants (resp. descendants) mais qui peuvent être fils d'un nœud $\oplus$ (resp. $\ominus$) dans l'arbre de décomposition d'une permutation en épingles.

On adopte dans ce paragraphe les notations génériques α (resp. β^+, resp. β^-) pour représenter une permutation en épingles simple (resp. un quasi-épi montant, resp. un quasi-épi descendant). Une formulation alternative (et plus visuelle) de la caractérisation donnée au théorème 8.25 est donnée par l'équation suivante :



On justifie cette équation en rappelant les conditions sur les arbres de décompositions pour que la permutation qui leur est associée soit une permutation en épingles : une permutation σ est une permutation en épingles si et seulement si son arbre de décomposition $\mathcal{T}_\sigma$ satisfait l'une des conditions ci-dessous.

- $\mathcal{T}_\sigma$ est une feuille.
- La racine de $\mathcal{T}_\sigma$ est un nœud linéaire d'étiquette $\oplus$ et tous ses fils sont des épis montants.
- La racine de $\mathcal{T}_\sigma$ est un nœud linéaire d'étiquette $\oplus$ et tous ses fils sont des épis montants, sauf l'un d'entre eux, qui appartient à $\mathcal{N}^+$.
- Le cas où la racine de $\mathcal{T}_\sigma$ est un nœud linéaire d'étiquette $\ominus$ est identique aux deux points ci-dessus, en remplaçant $\mathcal{E}^+$, $\mathcal{N}^+$ et épis montants par $\mathcal{E}^-$, $\mathcal{N}^-$ et épis descendants.
- La racine de $\mathcal{T}_\sigma$ est un nœud premier étiqueté par une permutation en épingles simple α, et tous ses fils sont des feuilles.
- La racine de $\mathcal{T}_\sigma$ est un nœud premier étiqueté par une permutation en épingles simple α, et qui a exactement un fils non feuille. Ce fils non trivial doit dilater un des points actifs de α (indiqués par _____).
- La racine de $\mathcal{T}_\sigma$ est un nœud premier étiqueté par un quasi-épi montant β^+, et elle a deux fils non réduits à des feuilles : l'un dilate de point de substitution principal de β^+ (noté .....), et l'autre est la permutation 1 2 qui dilate le point de substitution auxiliaire de β^+ (noté _____).
- Le cas où la racine de $\mathcal{T}_\sigma$ est étiquetée par un quasi-épi descendant β^- et a deux fils non feuilles est identique au cas précédent, à ceci près que 1 2 est remplacé par 2 1.

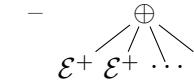
8.4.2 Séries génératrices simples mises en jeu

Dans l'équation décrivant les arbres de décomposition des permutations en épingles donnée plus haut, de nombreuses séries génératrices apparaissent. On peut les expliciter grâce aux techniques de combinatoire analytique de [FS08] rappelées dans le chapitre 10.

– $\mathcal{E}^+$, $\mathcal{E}^-$: Ce cas représente l'ensemble des arbres associés aux épis (montants ou descendants). D'après la définition 8.16, ces deux ensembles ont la énumération, et donc la même série génératrice, notée $E(z)$. Il y a deux épis montants (resp. descendants) de chaque taille n, sauf pour $n = 1, 2$ où il y a un unique épi dans chaque direction. Ainsi,

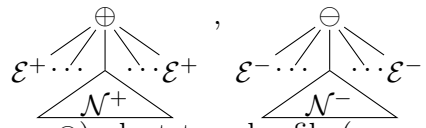
$$E^+(z) = E^-(z) = E(z) = \frac{z + z^3}{1 - z}.$$

On rappelle au passage que $\mathcal{E}^+ \bigcap \mathcal{E}^- = \{\bullet, \mathcal{T}_{2413}, \mathcal{T}_{3142}\}$.

– 

: Les séries génératrices correspondantes, notées $TE^+(z)$ et $TE^-(z)$, sont

$$TE^+(z) = TE^-(z) = TE(z) = \frac{(E(z))^2}{1 - E(z)}.$$

– 

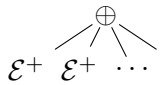
,

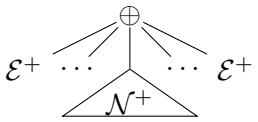
: Ces arbres de décomposition sont ceux ayant une racine $\oplus$ (resp. $\ominus$), dont tous les fils (en nombre supérieur ou égal à 2) sont des épis montants (resp. descendants), sauf un qui appartient à $\mathcal{N}^+$ (resp. $\mathcal{N}^-$). En termes de séries génératrices, cela revient à considérer une séquence d'épis montants (resp. descendants), dont un élément est pointé et remplacé par une permutation de $\mathcal{N}^+$ (resp. $\mathcal{N}^-$). On note $TEN(z)$ la série génératrice des séquences d'épis montants (resp. descendants) où l'un des épis a été remplacé par une permutation de $\mathcal{N}^+$ (resp. $\mathcal{N}^-$), et $N(z)$ la série génératrice des arbres de décomposition de $\mathcal{N}^+$ (resp. $\mathcal{N}^-$). On obtient alors

$$TEN^+(z) = TEN^-(z) = TEN(z) = \frac{2E(z) - E^2(z)}{(1 - E(z))^2} N(z).$$

– $\mathcal{N}^+$, $\mathcal{N}^-$: L'ensemble $\mathcal{N}^+$ (resp. $\mathcal{N}^-$) est l'ensemble des arbres de décomposition qui ne correspondent pas à des épis montants (resp. descendants) et dont les racines ne sont pas étiquetées par $\oplus$ (resp. $\ominus$). On considère à présent seulement le cas de $\mathcal{N}^+$, l'étude de $\mathcal{N}^-$ étant tout à fait similaire. Comme tout épi de taille au moins 4 est une permutation simple (voir lemme 8.17(i)), tout élément de taille au moins 4 dans $\mathcal{E}^+$ est de la forme 

pour une permutation en épingles simple α. D'après la définition 8.18, les permutations de taille au plus 3 dans $\mathcal{E}^+$ sont $1, 21, 231$ et 312, et les arbres de décomposition correspondants ont leur racine étiquetée par $\ominus$, sauf 1 dont l'arbre de décomposition est réduit à $\bullet$. Ainsi, l'intersection de $\mathcal{E}^+$ avec l'ensemble des arbres de racine $\oplus$ est vide. On a par conséquent

$\mathcal{N}^+ = \mathcal{S} - \mathcal{E}^+ -$ 

$-$ 

Du point de vue des séries génératrices, cela donne

$$\begin{aligned} N(z) &= N^-(z) = N^+(z) = S(z) - (E(z) + TE(z) + TEN(z)) \\ &= \frac{(z^3 + 2z - 1)(z^3 + S(z)z^3 + 2S(z)z + z - S(z))}{1 - 2z + z^2} \end{aligned}$$

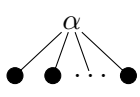
– β^+, β^- : On note $QE(z)$ la série génératrice des quasi-épis, comptés avec une multiplicité égale à leur nombre de paires de points de substitution. D'après la définition 8.20, si $n \geq 6$, il y a quatre épis montants (resp. descendants) de taille n, et pour tout $n < 6$ (et si bien sûr $n \geq 4$), il y a seulement deux telles permutations, mais qui ont multiplicité 2. Ainsi,

$$QE^+(z) = QE^-(z) = QE(z) = \frac{4z^4}{1 - z} \tag{8.1}$$

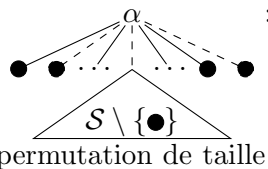
On remarque aussi que $\{\beta^+\} \bigcap \{\beta^-\} = \emptyset$ si l'on considère, comme pour le calcul de la série génératrice, les quasi-épis avec leurs points de substitution principal et auxiliaire marqués.

8.4.3 Série génératrice de la classe

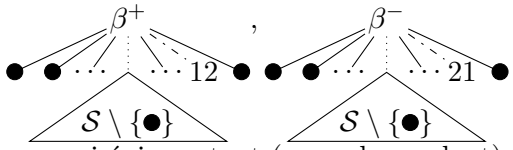
Avant de calculer la série génératrice $S(z)$ de la classe des permutations en épingles, d'autres termes de l'équation de la page 200 doivent être explicités.

– 

: Ces termes sont énumérés par la série $Si(z)$ définie dans le théorème 8.56.

– 

: La racine est un nœud premier et un de ses points actifs est dilaté par une permutation de taille au moins 2. Le théorème 8.57 donne la série génératrice $SiMult(z)$ des permutations en épingles simples comptées avec une multiplicité égale à leur nombre de points actifs. La série génératrice pour les arbres de cette forme est donc $SiMult(z)(\frac{S(z)-z}{z})$.

– 

: Ces arbres de décomposition ont leur racine étiquetée par un quasi-épi montant (resp. descendant), dont les points de substitution principal et auxiliaire sont fixés (énumérés par $QE(z)$, voir l'équation 8.1), et sont tels que

– le point de substitution principal est dilaté par une permutation non feuille, ce qui revient à remplacer la feuille correspondante sous la racine par un arbre de $S \setminus \{\bullet\}$,
– la feuille correspondant au point de substitution auxiliaire est remplacée par 12 (resp. 21).

Dans les séries génératrices, ces deux conditions correspondent respectivement à des multiplications par $\frac{S(z)-z}{z}$ et par z. On obtient donc que les séries génératrices des termes ayant la forme ci-dessus sont $QE^+(z)(S(z)-z)$ et $QE^-(z)(S(z)-z)$.

On est maintenant en mesure de transcrire l'équation donnant $\mathcal{S}$ en termes d'arbres (celle de la page 200) en une équation satisfaite par la série génératrice $S(z)$ des permutations en épingles, et on obtient :

$$\begin{aligned} S(z) &= z + \frac{E^+(z)^2}{1-E^+(z)} + \frac{2E^+(z)-E^+(z)^2}{(1-E^+(z))^2}N^+(z) + \frac{E^-(z)^2}{1-E^-(z)} + \frac{2E^-(z)-E^-(z)^2}{(1-E^-(z))^2}N^-(z) \\ &\quad + Si(z) + SiMult(z)\Big(\frac{S(z)-z}{z}\Big) + QE^+(z)\Big(z\frac{S(z)-z}{z}\Big) + QE^-(z)\Big(z\frac{S(z)-z}{z}\Big) \end{aligned}$$

La résolution de cette équation aboutit au résultat suivant :

Théorème 8.58. *La classe des permutations en épingles a une série génératrice rationnelle, qui est*

$$S(z) = z\frac{8z^6 - 20z^5 - 4z^4 + 12z^3 - 9z^2 + 6z - 1}{8z^8 - 20z^7 + 8z^6 + 12z^5 - 14z^4 + 26z^3 - 19z^2 + 8z - 1}$$

Un développement de Taylor de S donne le nombre de permutations en épingles de taille au plus 10 (d'autres termes pouvant être obtenus avec un logiciel de calcul formel) :

$$S(z) = z + 2z^2 + 6z^3 + 24z^4 + 120z^5 + 664z^6 + 3596z^7 + 19004z^8 + 99596z^9 + 521420z^{10} + \mathcal{O}(z^{11})$$

Remarquons que les huit premiers termes étaient déjà donnés dans [BRV08].

8.5 Discussion sur la base de la classe des permutations en épingles

Jusqu'ici, tout ce chapitre a été consacré à la preuve de la conjecture 8.9 de R. Brignall, N. Ruškuc et V. Vatter dans [BRV08], énonçant que la classe des permutations en épingles a une série génératrice rationnelle. Le théorème 8.58 répond positivement à cette conjecture. Toujours dans [BRV08], une autre question à propos de la classe des permutations en épingles reste en suspens, et concerne la base de cette classe.

On note $\mathcal{B}$ la base de la classe des permutations en épingles. D'après le théorème 1.25, $\mathcal{B}$ est l'ensemble des permutations qui n'admettent pas de représentations en épingles et qui sont minimales pour ce critère, au sens de la relation de motif $\preceq$.

R. Brignall, N. Ruškuc et V. Vatter considèrent dans [BRV08] qu'il n'est pas évident que la classe des permutations en épingles ait une base finie. On démontre ici que cette base $\mathcal{B}$ est en fait infinie, en décrivant une antichaîne infinie $(\sigma_n)_{n\geq 8}$ (c'est-à-dire que les permutations σ_n sont deux à deux incomparables pour $\preceq$) dans la base de la classe des permutations en épingles. On peut remarquer que (σ_n) peut être étendue par $\sigma_6 = 3\,6\,1\,5\,2\,4$ et $\sigma_7 = 3\,7\,4\,6\,1\,5\,2$, mais on ne peut prolonger cette antichaîne par aucune permutation de taille 5, celles-ci étant toutes des permutations en épingles (voir [BRV08]).

L'étude des antichaînes infinies de permutations est un domaine de recherche très actif depuis quelques années, comme on peut s'en rendre compte dans [AMR02, BS00b, Bri07] par exemple. Dans [Bri07], des antichaînes infinies de permutations sont obtenues en ajoutant des épingles autour d'un motif de petite taille. C'est cette technique que l'on utilise ici.

Les permutations $(\sigma_n)_{n\geq 8}$ sont construites par insertion d'épingles séparatrices autour de la permutation $\pi = 1\,5\,2\,4\,3$, dont la représentation graphique est donnée sur la figure 8.23, et qui possède une propriété particulière :

Lemme 8.59. *On considère la permutation $\pi = 1\,5\,2\,4\,3$ et on note x l'élément de π le plus à droite dans sa représentation graphique, qui correspond à la valeur 3. Il n'y a aucune représentation en épingles de π qui finisse par le point x. Cependant, tout motif de π obtenu par suppression d'un élément $y \neq x$ dans π possède une représentation en épingles se terminant en x.*

Démonstration. On note B la boîte englobante de tous les éléments de π sauf x. L'élément x sépare B en deux sous-ensembles contenant chacun deux points, de telle sorte que x ne peut satisfaire ni la condition d'indépendance, ni la condition de séparation par rapport à B. Cela démontre que π n'a pas de représentation en épingles se terminant en x.

Le deuxième point de l'énoncé du lemme 8.59 se prouve facilement par une étude exhaustive de tous les cas. □

On peut aussi remarquer que π est une permutation en épingles, comme toutes les permutations de taille au plus 5.

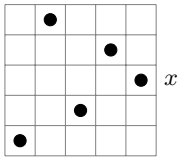


FIG. 8.23 – La permutation $1\,5\,2\,4\,3$, qui est le point de départ de la construction de chacune des permutations σ_n, dont l'ensemble forme une antichaîne infinie dans la base de la classe des permutations en épingles.

On définit alors les permutations $(\sigma_n)_{n\geq 8}$ à partir de la permutation π de la manière suivante :

Définition 8.60. Si $n = 2k+1, (k \geq 4)$, alors σ_n est la permutations obtenue à partir de π par insertion d'épingles séparantes, notées $s_6, s_7, \ldots, s_n$ selon le schéma $(UR)^{k-3}DR$. Si $n = 2k, (k \geq 4)$, alors σ_n est la permutation obtenue à partir de π par insertion d'épingles séparantes $s_6, s_7, \ldots, s_n$ selon le schéma $(UR)^{k-4}ULU$. Dans les deux cas, la première épingle s_6 sépare le point x des quatre autres points de π, et chaque épingle $s_k, (k \geq 7)$ sépare la dernière épingle insérée des autres points.

Remarque 8.61. L'indice n dans la définition 8.60 correspond à la taille de la permutation σ_n, et chaque permutation σ_n contient une unique occurrence de π.

La définition 8.60 est illustrée sur quelques exemples en figure 8.24.

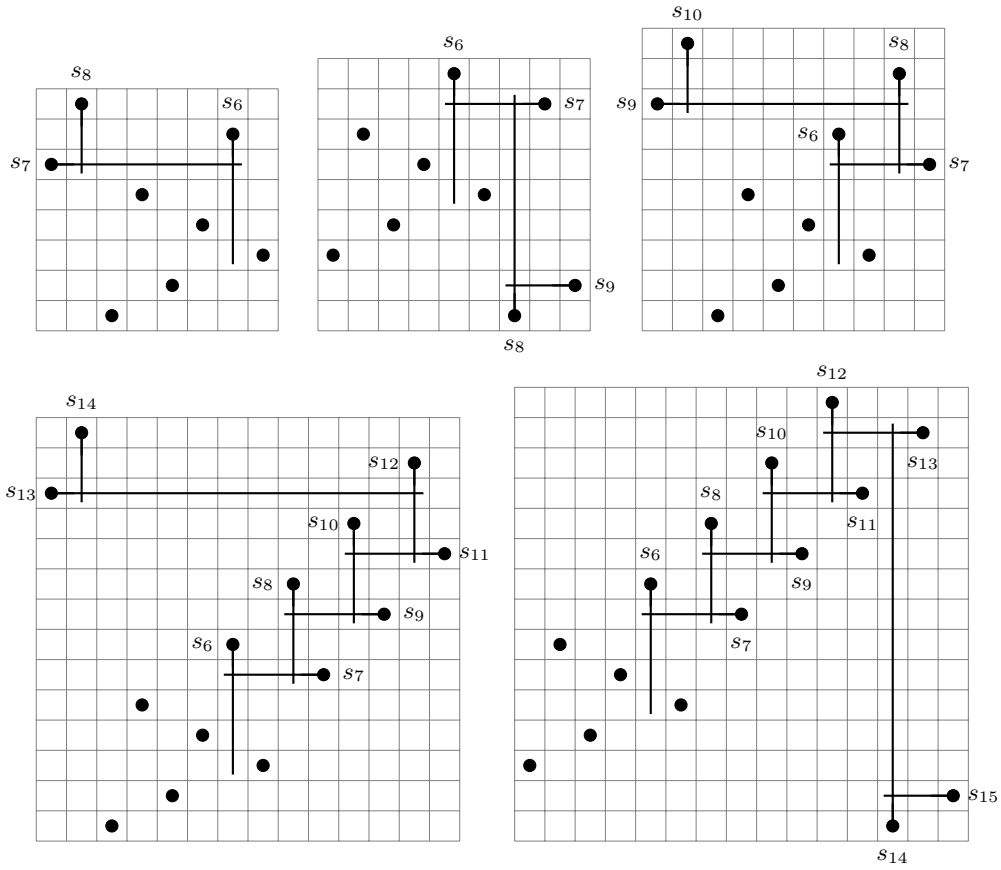


FIG. 8.24 – Les permutations σ_n pour $n = 8, 9, 10, 14$ et 15.

Proposition 8.62. *Pour tout n, la permutation σ_n n'a aucune représentation en épingles, mais chaque permutation obtenue par suppression d'un élément dans σ_n est une permutation en épingles.*

Démonstration. On démontre d'abord par l'absurde que σ_n n'a aucune représentation en épingles. On suppose donc qu'il existe une représentation en épingles p de σ_n. On considère l'unique occurrence de π dans σ_n, et on note $1_\pi, 2_\pi, \ldots, 5_\pi$ les éléments de σ_n qui correspondent respectivement aux éléments $1, 2, \ldots, 5$ dans π. Avec les notations du lemme 8.59, $x = 3_\pi$.

D'après le lemme 8.44, $\{p_1, p_2\}$ est un cavalier. Ainsi, il y a peu de paires de points de σ_n qui peuvent correspondre à $\{p_1, p_2\}$. Plus précisément, il y a cinq cavaliers dans σ_n, qui sont les paires de points $\{1_\pi, 2_\pi\}$, $\{2_\pi, 4_\pi\}$, $\{3_\pi, 4_\pi\}$, $\{4_\pi, 5_\pi\}$, et $\{s_{n-1}, s_n\}$. Pour chacun de ces cas, on dérive une contradiction.

Le raisonnement pour obtenir cette contradiction est le même dans tous les cas. Après p_1 et p_2, les épingles suivantes dans la représentation en épingles p sont déterminées par le lemme 8.10. Mais dans chacun des cinq cas à étudier, il arrive un moment où il y a plusieurs points sur les côtés la boîte englobante des point déjà lus par p. Le lemme 8.10 assure alors que la construction de p ne peut pas se poursuivre, démontrant ainsi que σ_n n'a pas de représentation en épingles.

On démontre à présent que la suppression de tout élément dans σ_n aboutit à une permutation qui possède une représentation en épingles. On distingue cinq cas, selon le point qui est supprimé : le point $x = 3_\pi$ de π, un autre point de π, la dernière épingle insérée autour de π (c'est-à-dire s_n), ou l'avant-dernière (c'est-à-dire s_{n-1}), ou enfin toute autre épingle s_k. On note σ' la permutation obtenue après la suppression dans σ_n.

Si $x = 3_\pi$ est supprimé de σ_n, alors une représentation en épingles de σ' est construite ainsi : on commence par une représentation en épingles de $\pi \setminus \{x\}$ (qui existe trivialement), puis on lit s_6 (qui est alors une épingle indépendante), et on continue avec les épingles $s_7, \ldots, s_n$ dans cet ordre (qui sont toutes des épingles séparantes).

Si c'est un point $y \neq x$ de π qui est supprimé de σ_n, alors d'après le lemme 8.59, il existe une représentation en épingles de $\pi \setminus \{y\}$ qui termine par x, qui peut être prolongée en une représentation en épingles de σ' par les épingles séparantes $(s_j)_{6 \leq j \leq n}$.

Si une épingle s_k, $k \geq 6$, est supprimée de σ_n, le début d'une représentation en épingles de σ' est construit de la manière suivante : on commence par une représentation en épingles p de $\pi \setminus \{1_\pi\}$ se terminant en $x = 3_\pi$ (dont l'existence est assurée par le lemme 8.59), et on continue avec les épingles séparantes s_j, en s'arrêtant à s_{k-1} (lorsque $k = 6$, on ne prend donc aucune épingle séparante s_j). La fin de la représentation en épingles de σ' varie selon que $k = n$, $k = n - 1$ ou $k \in [6..(n-2)]$.

Si $k = n$, 1_π joue le rôle de la dernière épingle, qui est séparante, et on obtient la représentation en épingles désirée. Si $k = n - 1$, après s_{n-2} viennent les épingles s_n, puis 1_π, toutes deux indépendantes. Dans tous les autres cas, après s_{k-1}, on termine la représentation en épingles de σ' en construction par les épingles 1_π puis s_{k+1} (indépendantes) et enfin $s_{k+2}, \ldots, s_n$ (séparantes).

Il est ainsi démontré que toute permutation obtenue par suppression d'un élément dans σ_n est une permutation en épingles. □

Conséquence 8.63. *La suite de permutations (σ_n) est une antichaîne (pour la relation d'ordre $\preceq$), et pour tout n, σ_n appartient à la base $\mathcal{B}$ des motifs exclus qui caractérise la classe des permutations en épingles.*

Démonstration. On démontre le premier point par l'absurde : on suppose que (σ_n) n'est pas une antichaîne. Il existe donc deux entiers k et n, avec $k < n$ tels que σ_k soit motif de σ_n. Comme $k < n$, cela implique qu'il existe une permutation σ' de taille $n - 1$, obtenue par suppression d'un élément dans σ_n telle que $\sigma_k \preceq \sigma' \prec \sigma_n$. D'après la proposition 8.62, σ' est une permutation en épingles. Le lemme 8.7 assure alors que σ_k est aussi une permutation en épingles, contredisant la conséquence 8.63.

La proposition 8.62 implique que tout motif strict de σ_n est une permutation en épingles, puisque l'ensemble des permutations en épingles forme une classe de permutations (d'après le lemme 8.7). Le deuxième point est donc simplement obtenu par définition de la base d'une classe de permutations. □

On peut alors conclure que :

Théorème 8.64. *La classe des permutations en épingles a une base infinie.*

La littérature ne présente que quelques rares exemples de classes de permutations ayant à la fois une base infinie et une série génératrice rationnelle. On en trouve un exemple dans [AAA$^+$07] : les classes $\mathcal{T}_k$ contenant les permutations obtenues après k *transposition switches* en série, pour $k \geq 5$. On peut remarquer que dans [AAA$^+$07] le caractère rationnel des séries génératrices est obtenu par des techniques de théorie des automates, ce qui peut être mis en parallèle avec la preuve donnée ici du théorème 8.58, où le langage des mots d'épingles joue un rôle fondamental.

Chapitre 9

Algorithme polynomial décidant si une classe contient un nombre fini de permutations simples

Sommaire

9.1 Mots d'épingles et première procédure de décision 208
9.1.1 Quelques propriétés des mots d'épingles 208
9.1.2 Les étapes de la procédure de décision de Brignall, Ruškuc et Vatter . 211
9.2 Quels mots d'épingles pour quelles permutations en épingles 214
9.2.1 Permutations en épingles de base 215
9.2.2 Permutations en épingles de racine linéaire construites récursivement . 220
9.2.3 Permutations en épingles de racine première construites récursivement 227
9.3 Facteurs de mots d'épingles pour identifier les motifs dans les permutations 232
9.4 Algorithme polynomial testant si une classe contient un nombre fini de simples 236
9.4.1 Les étapes de l'algorithme 236
9.4.2 Construction de l'automate pour les cas de base 238
9.4.3 Construction de l'automate pour les cas récursifs de racine linéaire . . 240
9.4.4 Construction de l'automate pour les cas récursifs de racine première . . 241
9.4.5 Récapitulatif de complexité 243
9.4.6 Marquage des états 243
9.4.7 Assemblage de l'automate 244
9.4.8 Cas particulier des classes fermées par produit de substitution 246

Dans ce chapitre, on s'intéresse en détails à la procédure de R. Brignall, N. Ruškuc et V. Vatter dans [BRV08] permettant de décider si une classe de permutations donnée par sa base finie de motifs exclus contient un nombre fini de permutations simples. On rappelle que cette étude est motivée par le théorème de M. H. Albert et M. D. Atkinson dans [AA05] affirmant que si une classe $\mathcal{C}$ contient un nombre fini de permutations simples, alors sa série génératrice est algébrique, et que la preuve de ce résultat donne en outre une méthode de calcul de la série génératrice de $\mathcal{C}$ étant données les permutations simples appartenant à $\mathcal{C}$.

Le théorème de M. H. Albert et M. D. Atkinson s'applique à toutes les classes de permutations, décrites de quelque manière que ce soit, et dont la base n'est pas *a priori* finie.[g] Il est important de remarquer que les travaux de [BRV08], aussi bien que l'amélioration qui en est proposée dans ce chapitre, ne s'appliquent qu'au cas des classes de permutations décrites par leur base de motifs exclus, et dans le cas où cette base est finie.

9.1 Mots d'épingles et première procédure de décision

La procédure de décision de [BRV08] utilise les représentations en épingles des permutations, à travers leur codage par des mots d'épingles. Avant de présenter les principales étapes de cette procédure, on donne certaines propriétés des mots d'épingles qui en justifieront la correction.

9.1.1 Quelques propriétés des mots d'épingles

On rappelle et on précise la définition des mots d'épingles, déjà donnée page 176.

Définition 9.1. Soit $p = (p_1, p_2, \dots, p_n)$ une représentation en épingles d'une permutation en épingles σ.
Un *mot d'épingles* $w = w_1 w_2 \dots w_n$ associé à p est un mot sur l'alphabet $\{1, 2, 3, 4, L, R, U, D\}$ tel que pour tout $k \geq 2$, l'épingle p_{k+1} est codée par la lettre w_{k+1} selon les règles suivantes :

- si p_{k+1} sépare p_k de l'ensemble $\{p_1, p_2, \dots, p_{k-1}\}$, alors elle se trouve sur l'un des côtés de la boîte englobante de $\{p_1, p_2, \dots, p_k\}$, et w_{k+1} est L, R, U ou D, selon la position de p_{k+1}, respectivement à gauche, à droite, au-dessus ou en-dessous de cette boîte englobante,
- si p_{k+1} satisfait les conditions d'extériorité et d'indépendance, et par là même se trouve dans un des coins (ou quadrants), numérotés selon le schéma $\frac{2|1}{3|4}$, par rapport à la boîte englobante de $\{p_1, p_2, \dots, p_k\}$, alors le numéro du quadrant est la lettre codant p_{k+1} dans le mot d'épingles.

Pour le codage de p_1 et p_2, on choisit un point fictif p_0 dans le plan, puis on code p_1 par la lettre (ici, un chiffre entre 1 et 4) correspondant à la position relative de p_1 par rapport à p_0, et enfin, on code p_2 comme les points p_k pour $k \geq 3$, selon sa position par rapport à la boîte englobante de $\{p_0, p_1\}$.

On rappelle que p_0 doit être choisi de telle sorte qu'aucun point de la permutation σ, sauf p_1, ne soit à l'intérieur de la boîte englobante de $\{p_0, p_1\}$, et que plusieurs tels choix de p_0 sont possibles, conduisant ainsi à avoir plusieurs mots d'épingles codant une même représentation en épingles.

Remarque 9.2. Pour toute représentation en épingles p donnée[h], il y a au plus huit mots d'épingles qui la codent. Ces huit mots d'épingles correspondent aux huit placements possibles de p_0 par rapport à p_1 et p_2 qui sont illustrés sur la figure 9.1.

Parmi ces huit préfixes de mots d'épingles, certains sont interdits lorsque p_3 est une épingle séparante : en effet si p_3 est une épingle de type U ou D (resp. L ou R) alors les préfixes $2U$ et $3U$ (resp. $3R$ et $4R$) ne sont pas admissibles dans un mot d'épingles codant p.

[g] On notera cependant que sous l'hypothèse qu'une classe $\mathcal{C}$ contient un nombre fini de permutations simples, le théorème de [AA05] affirme aussi que la base de $\mathcal{C}$ est finie.

[h] Contrairement au chapitre 8, l'ordre entre p_1 et p_2 est ici fixé à l'avance.

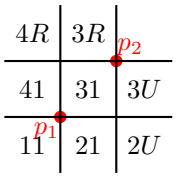


FIG. 9.1 – Les placements possibles de p_0 par rapport à p_1 et p_2 qui autorisent le codage par un mot d'épingles d'une représentation en épingles $p = (p_1, p_2, \ldots, p_n)$. À chaque place possible de p_0 sont indiquées les deux premières lettres du mot d'épingles correspondant codant p.

Définition 9.3. On introduit une notation supplémentaire pour distinguer les deux types de lettres des mots d'épingles. On appellera *chiffres* les lettres $1, 2, 3$ et 4, et *directions* les lettres L, R, U et D.

Une première propriété importante des mots d'épingles est la suivante :

Proposition 9.4. *Dans tout mot d'épingles, une lettre U ou D (resp. R ou L) ne peut être suivie que par une lettre de l'ensemble $\{1, 2, 3, 4, R, L\}$ (resp. $\{1, 2, 3, 4, U, D\}$).*

Démonstration. Par définition, dans toute représentation en épingles, il est impossible d'avoir deux épingles séparantes consécutives qui aient la même orientation (verticale ou horizontale). La proposition 9.4 est le pendant de cette remarque dans le codage par des mots d'épingles. □

Cette propriété est en fait une caractérisation des mots d'épingles :

Proposition 9.5. *Tout mot w sur l'alphabet $\{1, 2, 3, 4, L, R, U, D\}$ qui commence par un chiffre et ne contient pas de facteur dans $\{LL, RR, LR, RL, UU, DD, UD, DU\}$ est un mot d'épingles, c'est-à-dire qu'il code une représentation en épingles d'une permutation en épingles.*

Démonstration. Par hypothèse, w ne contient pas deux directions successives qui aient la même orientation (horizontale ou verticale). Il est alors évident de construire une représentation en épingles p qui soit codée par w, en insérant autour d'une origine fictive p_0 des épingles p_i qui sont
- indépendantes et dans le quadrant indiqué par w_i si w_i est un chiffre,
- séparantes (c'est-à-dire séparant p_{i-1} de $\{p_0, p_1, \ldots, p_{i-2}\}$) et dans la direction indiquée par w_i sinon.

□

Ce seront essentiellement les mots d'épingles codant les représentations en épingles *propres* (où toutes les épingles p_i pour $i \geq 3$ sont séparantes, voir définition 8.4 page 173) qui vont nous intéresser dans la suite. C'est pourquoi on précise :

Définition 9.6. Un mot d'épingles $w = w_1 \ldots w_n$ est un *mot d'épingles strict* lorsque w_1 est un chiffre et toutes les lettres suivantes de w sont des directions.

On notera $\mathcal{SP}$ le langage des mots d'épingles stricts.

Un mot d'épingles $w = w_1 \ldots w_n$ est un *mot d'épingles quasi-strict* lorsque w_1 et w_2 sont des chiffres et toutes les lettres suivantes de w sont des directions.

On avait déjà remarqué (page 176) que les mots d'épingles stricts codent des représentations en épingles propres, mais que tous les mots d'épingles codant une représentation en épingles propre ne sont pas stricts. On précise cette remarque :

Proposition 9.7. *Tout mot d'épingles codant une représentation en épingles propre est soit un mot d'épingles strict soit un mot d'épingles quasi-strict.*

Toute représentation en épingle propre admet un codage par un mot d'épingles strict.

Démonstration. Dans une représentation en épingles propre $p = (p_1, p_2, \ldots, p_n)$, toutes les épingles à partir de p_3 sont des épingles séparantes, et sont donc codées par une lettre qui est une direction dans un mot d'épingles associé à p. La figure 9.1 assure qu'on peut choisir l'origine p_0 d'une représentation en épingles de sorte que le mot d'épingles correspondant soit strict. □

Remarque 9.8. Une conséquence de la proposition 9.7 est en particulier que tout mot d'épingles codant une permutation simple est un mot d'épingles strict ou quasi-strict.

On termine ces rappels sur les mots d'épingles en définissant l'ordre partiel $\trianglelefteq$ de [BRV08] sur les mots d'épingles, qui transporte dans ce contexte la relation de motif $\preceq$ sur les permutations.

Définition 9.9. Soit u un mot d'épingles. On définit un *facteur suivant un chiffre* dans u comme étant un facteur de u qui commence par un chiffre, et dont toutes les autres lettres sont des directions. Un facteur suivant un chiffre est dit *fort* s'il ne peut pas être prolongé à droite. Les facteurs forts suivant un chiffre portent le nom de *strongly numeral-led factors* en anglais.

Définition 9.10. Soient u et w deux mots d'épingles. On décompose u en ses facteurs forts suivant un chiffre, sous la forme $u = u^{(1)} \ldots u^{(j)}$. On note alors $u \trianglelefteq w$ s'il existe une décomposition de w en une séquence de facteurs $w = v^{(1)}w^{(1)} \ldots v^{(j)}w^{(j)}v^{(j+1)}$ telle que pour tout $i \in [1..j]$
- si $w^{(i)}$ commence par un chiffre, alors $w^{(i)} = u^{(i)}$, et
- si $w^{(i)}$ commence par une direction, alors $v^{(i)}$ est non vide, la première lettre de $w^{(i)}$ correspond à un point qui se trouve dans le quadrant indiqué par la première lettre de $u^{(i)}$, et à partir de leur deuxième lettre, $u^{(i)}$ et $w^{(i)}$ sont deux mots identiques.

Exemple 9.11. Le mot $u = 1RUL3U42D$ vérifie $u \trianglelefteq w$ pour $w = 1RUL4RDLULU4RULDL$. La décomposition de w en $w = v^{(1)}w^{(1)} \ldots v^{(j)}w^{(j)}v^{(j+1)}$ est donnée par

$$w = \underbrace{1RUL}_{w^{(1)}}\underbrace{4RD}_{v^{(2)}}\underbrace{LU}_{w^{(2)}}\underbrace{LU}_{v^{(3)}}\underbrace{4}_{w^{(3)}}\underbrace{RU}_{v^{(4)}}\underbrace{LD}_{w^{(4)}}\underbrace{L}_{v^{(5)}},$$

$v^{(1)}$ étant vide. Les représentations en épingles correspondantes sont données sur la figure 9.2.

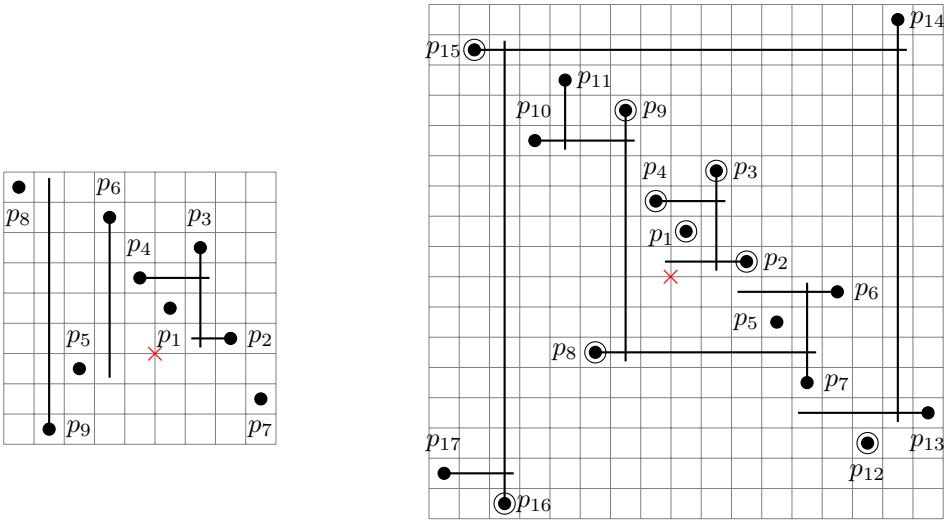


FIG. 9.2 – Les représentations en épingles associées aux mots d'épingles u et w de l'exemple 9.11. Les points origines p_0 sont marqués d'une croix rouge. Dans la représentation en épingles de droite, correspondant au mot w, les points entourés sont ceux apparaissant dans un facteur $w^{(i)}$. Par exemple, l'épingle p_8 codant la première lettre de $w^{(2)}$ correspond à un point dans le quadrant 3 par rapport aux épingles p_1 à p_4 qui codent $w^{(1)}$. On remarque que ces points entourés forment une occurrence de la permutation codée par u.

Cet ordre sur les mots d'épingles est fortement lié à la relation de motif $\preceq$ sur les permutations (qui est une relation d'ordre) :

Lemme 9.12 ([BRV08])**.** *Si le mot d'épingles w correspond à la permutation σ et si $\pi \preceq \sigma$, alors il existe un mot d'épingles u correspondant à π tel que $u \trianglelefteq w$. Inversement, si $u \trianglelefteq w$, alors la permutation codée par le mot d'épingles u est motif de la permutation correspondant à w.*

Conséquence 9.13. *Si π et σ sont deux permutations, et si w est un mot d'épingles qui code σ, alors $\pi \preceq \sigma$ si et seulement s'il existe un mot d'épingles u codant π tel que $u \trianglelefteq w$.*

Une spécialisation de cette conséquence au cas des permutations en épingles propres sera proposée par le lemme 9.22 page 213. Dans le lemme 9.22, on explicite le fait qu'une permutation codée par un mot d'épingles est une permutation en épingles. De la même façon, on aurait pu proposer une variante de la conséquence 9.13 explicitant le fait que σ et π sont des permutations en épingles.

9.1.2 Les étapes de la procédure de décision de Brignall, Ruškuc et Vatter

On présente ici les principales étapes de la procédure de décision de [BRV08] pour tester qu'une classe de permutations donnée par sa base finie contient un nombre fini de permutations simples.

Réduction à un sous-ensemble de permutations presque simples

La procédure décrite dans [BRV08] repose sur un lemme démontré au préalable dans [BHV08a] par R. Brignall, S. Huczynska et V. Vatter. Pour énoncer ce lemme, il faut définir trois types particuliers de permutations.

Définition 9.14. Une *permutation alternante* est une permutation telle que tous les éléments de valeur impaire se trouvent à gauche de tous les éléments de valeur paire, ou toute permutation symétrique (pour une symétrie du carré, voir page 19) d'une permutation vérifiant cette propriété.

Une *permutation parallèle* est une permutation alternante où
- soit chacun des deux ensembles d'éléments qui la définissent forme une séquence croissante,
- soit chacun de ces deux ensembles forme une séquence décroissante.

Une *permutation en chevron* est une permutation alternante où les deux ensembles d'éléments qui la définissent forment deux séquences d'entiers, l'une croissante et l'autre décroissante.

Une *permutation en épingles propre* est une permutation en épingles qui admet une représentation en épingles propre.

Les permutations parallèles, en chevrons, et en épingles propres ne sont pas nécessairement des permutations simples, mais ce sont des permutations "presque simples", au sens de la proposition suivante :

Proposition 9.15. *Toute permutation en épingles propre de taille $n \geq 5$ contient une permutation simple de taille n ou $n-1$.*

Toute permutation parallèle de taille $n \geq 5$ contient une permutation simple de taille n, $n-1$ ou $n-2$.

Toute permutation en chevron de taille $n \geq 3$ peut être transformée en une permutation simple de taille $n+1$ par insertion d'un seul point, selon deux types d'insertion différents. Les permutations obtenues par ces deux insertions sont appelées permutations simples en chevron de type 1 *et* de type 2 *(voir figure 9.3).*

Réciproquement, les permutations parallèles, les permutations simples en chevron et les permutations en épingles propres décrivent, dans une certaine mesure, l'ensemble des permutations simples :

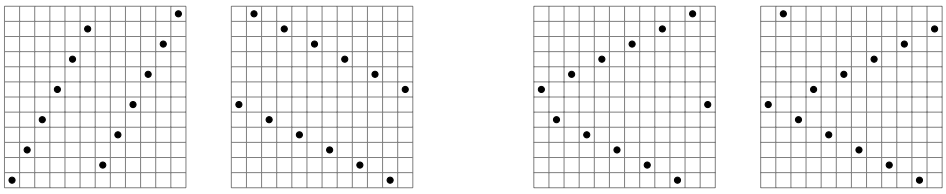


FIG. 9.3 – De gauche à droite : deux permutations parallèles, une permutation en chevron simple de type 1 et une permutation en chevron simple de type 2.

Proposition 9.16. *Il existe une fonction f telle que pour tout k, toute permutation simple de taille supérieure ou égale à $f(k)$ contient*
- *soit une permutation parallèle de taille k,*
- *soit une permutation simple en chevron (de type 1 ou 2) de taille k,*
- *soit une permutation en épingles propre de taille k.*

La fonction f de la proposition 9.16 est triplement exponentielle en k. On pourra consulter [BHV08a] pour plus de détails sur f et pour les preuves des propositions 9.15 et 9.16.

Ces deux propositions ont pour conséquence immédiate le lemme suivant, qui sert de point de départ à la procédure de décision de [BRV08] :

Lemme 9.17. *Une classe de permutations $\mathcal{C}$ contient un nombre infini de permutations simples si et seulement si elle contient*
- *soit un nombre infini de permutations parallèles,*
- *soit un nombre infini de permutations simples en chevron de type 1,*
- *soit un nombre infini de permutations simples en chevron de type 2,*
- *soit un nombre infini de permutations en épingles propres.*

Pour décider si une classe contient un nombre fini de permutations simples, il suffit donc de savoir décider si elle contient un nombre fini de permutations dans chacune des quatre catégories données au lemme 9.17.

Procédures de décision simples

On considère une classe $\mathcal{C} = \mathcal{S}(\mathcal{B})$ décrite par sa base finie motifs exclus $\mathcal{B}$. Pour tester si $\mathcal{C}$ contient un nombre fini de permutations dans chacune des trois premières catégories du lemme 9.17, [BRV08] offre les propositions suivantes, toutes démontrées en utilisant le principe de *juxtaposition* de classes de permutations [Atk99].

Proposition 9.18. *Une classe de permutations $\mathcal{C} = \mathcal{S}(\mathcal{B})$ contient un nombre infini de permutations parallèles si et seulement si $\mathcal{B}$ contient un élément de chaque classe symétrique de la classe $\mathcal{S}(1\,2\,3, 2\,4\,1\,3, 3\,4\,1\,2)$.*

Proposition 9.19. *Une classe de permutations $\mathcal{C} = \mathcal{S}(\mathcal{B})$ contient un nombre infini de permutations simples en chevron de type 1 si et seulement si $\mathcal{B}$ contient un élément de chaque classe symétrique de la classe $\mathcal{S}(1\,2\,4\,3, 1\,3\,2\,4, 1\,4\,2\,3, 1\,4\,3\,2, 2\,4\,3\,1, 3\,1\,2\,4, 4\,1\,2\,3, 4\,1\,3\,2, 4\,2\,3\,1, 4\,3\,1\,2)$.*

Proposition 9.20. *Une classe de permutations $\mathcal{C} = \mathcal{S}(\mathcal{B})$ contient un nombre infini de permutations simples en chevron de type 2 si et seulement si $\mathcal{B}$ contient un élément de chaque classe symétrique de la classe $\mathcal{S}(2\,1\,3\,4, 2\,1\,4\,3, 3\,1\,2\,4, 3\,1\,4\,2, 3\,2\,4\,1, 3\,4\,1\,2, 4\,1\,2\,3, 4\,1\,3\,2, 4\,2\,3\,1, 4\,3\,1\,2)$.*

Pour tester si une classe $\mathcal{C} = \mathcal{S}(\mathcal{B})$ contient un nombre fini de permutations simples en chevron ou de permutations parallèles, il suffit donc de procéder à des tests d'occurrence de motifs de

taille au plus 4 dans les permutations $\beta_1, \beta_2, \ldots, \beta_k$ qui composent la base $\mathcal{B}$ de $\mathcal{C}$, supposée finie. D'après un résultat de [AAAH01], proposé comme le théorème 3.5 page 71, la recherche d'une occurrence d'un motif de taille au plus 4 dans une permutation de taille n peut être effectuée en temps $\mathcal{O}(n \log n)$. Ainsi, en appliquant cette méthode, on obtient :

Conséquence 9.21. *Il existe un algorithme testant si une classe $\mathcal{C} = \mathcal{S}(\mathcal{B})$ donnée par sa base finie $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_k\}$ contient un nombre fini de permutations parallèles (resp. de permutations simples en chevron de type* 1*, resp. de permutations simples en chevron de type* 2*), dont la complexité est $\mathcal{O}(n \log n)$ si $n = \sum |\beta_i|$, ou $\mathcal{O}(k \cdot n \log n)$ si $n = \max |\beta_i|$.*

Procédure de décision pour les permutations en épingles propres

La procédure de décision qui teste le quatrième et dernier point du lemme 9.17 est plus complexe. Elle utilise le codage des représentations en épingles par des mots d'épingles, et des techniques de théorie des automates. Dans tout ce paragraphe, on note $\mathcal{C}$ une classe de permutations, et $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_k\}$ sa base de motifs exclus, supposée finie.

Lemme 9.22. *Une permutation en épingles propre σ contient en tant que motif une permutation $\beta \in \mathcal{B}$ si et seulement si β est une permutation en épingles et il existe un mot d'épingles u codant β et un mot d'épingles* strict *w codant σ tels que $u \trianglelefteq w$.*

Démonstration. Ce lemme est directement déduit de la conséquence 9.13, sachant que les permutations en épingles forment une classe de permutations, et que toute représentation en épingles propre peut être codée par un mot d'épingles strict (proposition 9.7). □

On peut donc considérer l'ensemble $\mathcal{E}$ des mots d'épingles stricts w tels que $w \trianglerighteq u$ pour un mot d'épingles u codant un motif $\beta \in \mathcal{B}$. Cet ensemble $\mathcal{E}$ correspond exactement aux permutations en épingles propres qui contiennent un motif $\beta \in \mathcal{B}$, c'est-à-dire qui n'appartiennent pas à $\mathcal{C} = \mathcal{S}(\mathcal{B})$.

On peut remarquer que seuls les motifs de $\mathcal{B}$ qui sont des permutations en épingles sont utiles dans la définition de $\mathcal{E}$. Ceci s'explique par le fait que les permutations en épingles forment une classe de permutations : ainsi, tout motif de $\mathcal{B}$ qui n'est pas une permutation en épingles n'empêche aucune permutation en épingles d'appartenir à $\mathcal{C}$.

Proposition 9.23. *On rappelle que $\mathcal{SP}$ désigne le langage de tous les mots d'épingles stricts.*

Le langage $\mathcal{SP} \setminus \mathcal{E}$ est le langage des mots d'épingles stricts qui codent des permutations appartenant à $\mathcal{C}$. $\mathcal{C}$ contient un nombre infini de permutations en épingles propres si et seulement si $\mathcal{SP} \setminus \mathcal{E}$ est infini.

Démonstration. C'est une conséquence directe du fait que tout mot d'épingles strict code une permutation en épingles propre – une permutation en épingles propre étant associée à un nombre fini de mots d'épingles stricts (au plus 32 d'après le lemme 8.47 et la figure 9.1), et que réciproquement toute permutation en épingles propre admet un codage par un mot d'épingles strict. □

La procédure proposée par [BRV08] consiste à construire un automate reconnaissant le langage $\mathcal{SP} \setminus \mathcal{E}$. On peut alors décider par des techniques standards de théorie des automates si $\mathcal{SP} \setminus \mathcal{E}$ est ou non fini.

Pour construire cet automate, [BRV08] donne un transducteur qui, sur tout mot d'épingles u en entrée, produit un automate $\mathcal{A}_u$ acceptant l'ensemble des mots d'épingles stricts w qui contiennent u au sens de $\trianglelefteq$. L'ensemble des mots d'épingles u qui codent un motif β de la base $\mathcal{B}$ de $\mathcal{C}$ étant fini, on peut construire pour chacun de ces mots l'automate $\mathcal{A}_u$ correspondant.

Les langages reconnus par des automates (c'est-à-dire les langages rationnels) sur un alphabet donné (ici, $\{1, 2, 3, 4, L, R, U, D\}$) sont une classe de langages fermée pour l'union, la complémentation et l'intersection. Des opérations explicites sur les automates correspondent à chacune de ces opérations sur les langages. On peut donc construire successivement :

- un automate $\mathcal{A}_\mathcal{L}$ qui reconnait le langage $\mathcal{L}$ des mots d'épingles stricts $\{w : \exists u \exists \beta \in \mathcal{B}$ tels que u est un mot d'épingles codant β et $u \trianglelefteq w\}$,
- un automate $\mathcal{A}^c_\mathcal{L}$ qui reconnait le langage $\mathcal{L}^c$ complémentaire de $\mathcal{L}$ sur l'alphabet $\{1, 2, 3, 4, L, R, U, D\}$,
- et un automate $\mathcal{A}$ qui reconnait l'intersection de $\mathcal{L}^c$ et de $\mathcal{SP}$.

Il suffit alors de tester si le langage reconnu par $\mathcal{A}$ est infini ou non. [BRV08] ne donne pas de méthode explicite pour le faire, mais une telle méthode classique consiste à chercher dans $\mathcal{A}$ un cycle qui soit accessible (atteignable depuis un état initial) et co-accessible (depuis lequel on peut rejoindre un état final).

La méthode décrite par [BRV08] est effective, mais les considérations de complexité ne sont pas abordées. Après une analyse, on peut décrire les tailles des automates construits, la taille d'un automate désignant son nombre d'états. On désigne par n la taille maximale d'un motif $\beta \in \mathcal{B}$ et par k le nombre de motifs dans $\mathcal{B}$.

- Pour tout mot d'épingles u codant un motif $\beta \in \mathcal{B}$, $\mathcal{A}_u$ est de taille $\mathcal{O}(|u|) = \mathcal{O}(n)$, puisque $\mathcal{A}_u$ est construit comme produit d'un transducteur donné à 9 états avec un automate reconnaissant seulement u, de taille linéaire en $|u|$ *a priori*.
- L'automate $\mathcal{A}_\mathcal{L}$ correspond à l'union des automates $\mathcal{A}_u$, pour tous les mots d'épingles u codant un motif $\beta \in \mathcal{B}$. Plusieurs constructions sont possibles pour l'union d'automates : par juxtaposition, ou par produit[i]. On note N le nombre de mots d'épingles u qui codent un motif $\beta \in \mathcal{B}$. Dans la première construction d'union (par juxtaposition), $\mathcal{A}_\mathcal{L}$ a taille $\mathcal{O}(n \cdot N)$. Dans la seconde (par produit), la taille de $\mathcal{A}_\mathcal{L}$ est $\mathcal{O}(n^N)$. On conservera la première ici.
- Pour pouvoir complémenter, il faut passer par une étape de déterminisation, qui induit une explosion exponentielle du nombre d'états. Ainsi, $\mathcal{A}^c_\mathcal{L}$ est de taille $\mathcal{O}(2^{(n \cdot N)})$.
- L'intersection avec le langage $\mathcal{SP}$ des mots d'épingles stricts consiste simplement à faire le produit de $\mathcal{A}^c_\mathcal{L}$ avec un automate à 4 états, de sorte que la taille de $\mathcal{A}$ est aussi de l'ordre de $\mathcal{O}(2^{(n \cdot N)})$.

La construction de chacun de ces automates demande un temps qui est proportionnel à sa taille.

Il n'y a pas de maîtrise *a priori* sur le nombre N de mots d'épingles u à considérer. On peut remarquer que la permutation identité $12 \dots n$ admet de l'ordre de 2^n mots d'épingles (tous les mots de taille n sur l'alphabet $\{1, 3\}$). Ainsi, N peut être de l'ordre de $k \cdot 2^n$ (voire plus), conduisant à un automate $\mathcal{A}$ de taille $\mathcal{O}(2^{(n \cdot k \cdot 2^n)})$.

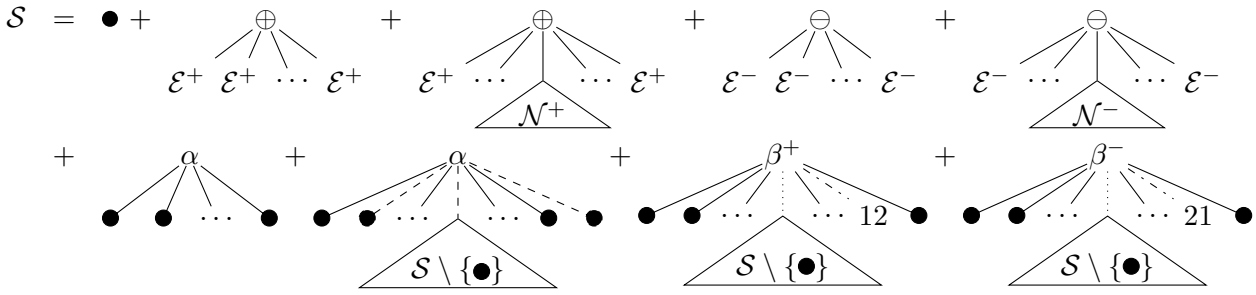
En outre, non seulement la valeur de N n'est pas connue, mais l'identification des motifs $\beta \in \mathcal{B}$ qui sont des permutations en épingles, et le calcul des mots d'épingles u qui leur sont associés n'est pas traité dans [BRV08].

Le travail présenté dans ce chapitre a la vocation de rendre la procédure de [BRV08] polynomiale (en $\mathcal{O}(n^{3k})$), depuis la détermination des permutations qui sont en épingles dans la base $\mathcal{B}$, jusqu'à la recherche d'un cycle dans un automate permettant de décider si $\mathcal{C}$ contient ou non un nombre fini de permutations en épingles propres. Avec le lemme 9.17 et la conséquence 9.21, cela fournira un algorithme polynomial pour décider si une classe donnée par sa base finie contient un nombre fini de permutations simples, et dans ce cas a une série génératrice algébrique.

9.2 Quels mots d'épingles pour quelles permutations en épingles

Ce paragraphe s'intéresse à décrire, pour toute permutation en épingles σ, l'ensemble des mots d'épingles associés à σ, que l'on notera $P(\sigma)$. Pour ce faire, on travaille sur les arbres de décomposition des permutations en épingles plutôt que sur les permutations en épingles elles-même. On rappelle la caractérisation de l'ensemble $\mathcal{S}$ des arbres de décomposition des permutations en épingles donnée page 200 :

[i] La construction par produit conserve le déterminisme des automates, c'est pourquoi on la préfèrera dans la suite. Ici, les automates $\mathcal{A}_u$ n'étant pas déterministes au départ, on n'a aucune raison de choisir la construction produit.



Cette équation fait naturellement apparaître trois cas à envisager pour décrire les langages des mots d'épingles associés à une permutation en épingles σ : les cas non récursifs dans cette équation, les cas récursifs où la racine est linéaire, et les cas récursifs où la racine est première.

9.2.1 Permutations en épingles de base

Permutation de taille 1

Il est évident que la permutation de taille 1, dont l'arbre de décomposition est réduit à une feuille, admet exactement quatre mots d'épingles, à savoir les mots 1, 2, 3 et 4.

Permutations simples

Les permutations en épingles de racine première qui ne sont pas décrite par un cas récursif de l'équation de la page 215 sont les permutations en épingles simples.

On considère une permutation en épingles simple σ. Comme on l'a déjà remarqué (voir juste après la définition 8.42), une représentation en épingles p de σ est nécessairement propre. Le lemme 8.44 assure alors que les deux premiers points de p forment un cavalier dans σ, qui est par définition actif (voir la définition 8.46). D'après le lemme 8.47, il y a au plus quatre cavaliers actifs dans σ. Un cavalier actif étant fixé, la remarque 8.49 justifie que la suite de la représentation en épingles p est déterminée. Sachant que pour chaque cavalier actif, les deux ordres possibles pour lire les deux points formant le cavalier sont admissibles, on obtient un maximum de huit représentations en épingles correspondant à σ. Pour chacune de ces représentations en épingles, la remarque 9.2 assure qu'il y a huit codages possibles par un mot d'épingles. On conclut donc qu'à toute permutation en épingles simple σ correspondent au plus 64 mots d'épingles.

Pour que cette construction soit effective, il nous faut décrire un moyen algorithmique de construire cet ensemble des mots d'épingles associés à σ. On notera n la taille de σ.

On recherche d'abord tous les cavaliers présents dans σ, ce qui peut être fait en temps $\mathcal{O}(n)$, chaque point de σ étant impliqué dans au plus huit cavaliers. Le nombre des cavaliers ainsi identifiés est aussi en $\mathcal{O}(n)$.

Pour chacun de ces cavaliers $\{p_1, p_2\}$, on construit une suite $p = (p_1, p_2, p_3, \ldots)$ où chaque p_i pour $i \geq 3$ sépare p_{i-1} de $\{p_1, \ldots, p_{i-2}\}$, aussi longue que possible (c'est-à-dire qu'on s'arrête lorsqu'aucun point ne sépare p_{i-1} de $\{p_1, \ldots, p_{i-2}\}$). Pour être un peu plus rapide, on peut en plus arrêter la construction de la suite p dès que plusieurs possibilités pour p_i se présentent. Le temps de calcul est $\mathcal{O}(n)$ pour chaque cavalier.

Comme toutes les représentations en épingles de σ sont propres, elles correspondent exactement aux suites p ainsi construites dont la construction va à son terme, c'est-à-dire qui utilisent les n points de σ.

On obtient ainsi toutes les représentations en épingles de σ, en temps $\mathcal{O}(n^2)$. Pour le passage des représentations en épingles aux mots d'épingles codant σ, on utilise simplement la remarque 9.2

pour les deux premières lettres, et les suivantes sont obtenues selon la définition 9.1 en parcourant p, donc en temps $\mathcal{O}(n)$.

On peut résumer ce qui précède par la proposition suivante :

Proposition 9.24. *Une permutation en épingles simple σ de taille n est codée par au plus 64 mots d'épingles, et le calcul de ces mots d'épingles peut être fait en temps $\mathcal{O}(n^2)$.*

Remarque 9.25. Il est important pour la suite de remarquer que cette procédure n'utilise pas le fait que σ est une permutation en épingles, mais seulement que σ est une permutation simple. Ainsi, étant donnée une permutation simple σ de taille n, on dispose d'un algorithme en temps $\mathcal{O}(n^2)$ qui teste si σ est ou non une permutation en épingles, et qui calcule le cas échéant les mots d'épingles de σ.

Permutations de racine linéaire

On traite seulement le cas des permutations de racine $\oplus$, le cas de la racine $\ominus$ étant symétrique. On cherche donc à décrire l'ensemble des mots d'épingles associés à une permutation en épingles σ qui se décompose sous la forme $\sigma = \oplus[\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(q)}]$ où tous les $\alpha^{(i)}$ sont des épis montants. La notation ξ étant adoptée dans toute la suite pour noter des épis, on écrira $\sigma = \oplus[\xi_1, \xi_2, \dots, \xi_q]$.

On a vu au chapitre précédent qu'il existe toujours une représentation en épingles de σ qui lit chacun des ξ_i en une fois (voir le lemme 8.31). Cependant, on veut obtenir ici la totalité des mots d'épingles codant des représentations en épingles de σ, et il existe aussi des représentations en épingles de σ qui lisent certains ξ_i en deux fois. Le lemme 9.26 précise dans quel cadre.

Lemme 9.26. *Soit $p = (p_1, p_2, \dots, p_n)$ une représentation en épingles de σ.*

Le seul ξ_i qui peut être lu en deux fois par p est celui auquel appartient le premier point p_1 de p, noté ξ_{i_0}.

Si p lit ξ_{i_0} en deux fois, alors le deuxième ξ_i lu par p est soit ξ_{i_0-1} soit ξ_{i_0+1}, et est réduit à une feuille, notée x. En outre, l'ensemble de points $E = \xi_{i_0} \cup \{x\}$ est lu en une fois par p, et les autres ξ_i sont aussi lus chacun en une fois par p, après E.

Démonstration. C'est une conséquence des lemmes 9.40 et 9.42 prouvés dans le cas général d'une permutation en épingles de racine $\oplus$, où les fils de cette racine ne sont pas forcément tous des épis.

Lorsqu'un épi ξ_i est lu en deux fois, les lectures en deux fois de ce bloc sont plus précisément identifiées comme cas particuliers de la figure 9.8 où le bloc S représenté est de taille 1 ou de taille 2 (seulement dans les cas $U1$,$U2$,$D1$ et $D2$). □

Conséquence 9.27. *Toute représentation en épingles p de σ commence par lire en une fois un ensemble de points correspondant à deux fils consécutifs de la racine (par exemple ξ_j et ξ_{j+1}), puis p lit les fils ξ_i restants, chacun en une fois.*

L'ordre de lecture entre les ξ_i pour $i < j$ est nécessairement ξ_{j-1} puis $\xi_{j-2}, \dots$ et enfin ξ_1.

L'ordre de lecture entre les ξ_i pour $i > j+1$ est nécessairement ξ_{j+2} puis $\xi_{j+3}, \dots$ et enfin ξ_q.

Démonstration. Le premier point est évident si tous les ξ_i sont lus en une fois. Le lemme 9.26 prouve que c'est aussi le cas si p lit un ξ_i en deux fois.

Le deuxième point est assuré par le fait que toute épingle p_i de p doit satisfaire la condition d'extériorité par rapport à $\{p_1, \dots, p_{i-1}\}$. □

La conséquence 9.27 implique que la restriction de p sur tout ξ_i pour $i < j$ (resp. pour $i > j+1$) est une représentation en épingles de ξ_i dont l'origine est située dans le quadrant 1 (resp. 3) par rapport à l'ensemble des points de ξ_i. C'est pourquoi on pose la définition suivante :

Définition 9.28. Pour tout épi montant ξ, on note $f^{(\ell)}(\xi)$ l'ensemble des mots d'épingles codant une représentation en épingles de ξ dont l'origine se trouve dans le quadrant $\ell = 1$ ou 3 par rapport à l'ensemble des points de ξ.

On a vu au chapitre 8 qu'un épi de taille au moins 5 possède exactement 2 cavaliers actifs, un à chacune des extrémités de la diagonale définissant la direction de cet épi (voir le lemme 8.47). Dans le cas d'un épi montant, les deux points d'un cavalier actif forment un motif 21. Ce cavalier est alors soit *horizontal* (de type H), c'est-à-dire de la forme  , soit *vertical* (de type V), c'est-à-dire de la forme  . On dira qu'un épi montant est de type (A, B), pour $A, B \in \{H, V\}$ lorsque son cavalier actif dans l'angle en bas à gauche est de type A, et celui dans l'angle en haut à droite est de type B. On remarque qu'un épi montant de taille paire est de type (H, H) ou (V, V), et qu'un épi montant de taille impaire est de type (H, V) ou (V, H).

Lemme 9.29. *Soit ξ un épi montant, de taille $n \geq 5$. Si n est pair, on pose $n = 2p + 2$, et alors*

$$f^{(1)}(\xi) = \begin{cases} 3L(DL)^p \text{ si } \xi \text{ est de type } (H,H) \\ 3D(LD)^p \text{ si } \xi \text{ est de type } (V,V) \end{cases} \quad \text{et } f^{(3)}(\xi) = \begin{cases} 1R(UR)^p \text{ si } \xi \text{ est de type } (H,H) \\ 1U(RU)^p \text{ si } \xi \text{ est de type } (V,V). \end{cases}$$

Si n est impair, on pose $n = 2p + 1$, et alors

$$f^{(1)}(\xi) = \begin{cases} 3(DL)^p \text{ si } \xi \text{ est de type } (H,V) \\ 3(LD)^p \text{ si } \xi \text{ est de type } (V,H) \end{cases} \quad \text{et } f^{(3)}(\xi) = \begin{cases} 1(RU)^p \text{ si } \xi \text{ est de type } (H,V) \\ 1(UR)^p \text{ si } \xi \text{ est de type } (V,H). \end{cases}$$

Pour les épis montants de taille inférieure à 5, les valeurs de $f^{(1)}$ et $f^{(3)}$ sont données par :

$f^{(1)}(1) = 3$	$f^{(3)}(1) = 1$	$f^{(1)}(21) = \{3D, 3L\}$	$f^{(3)}(21) = \{1R, 1U\}$
$f^{(1)}(312) = 3LD$	$f^{(3)}(312) = 1UR$	$f^{(1)}(231) = 3DL$	$f^{(3)}(231) = 1RU$
$f^{(1)}(2413) = 3LDL$	$f^{(3)}(2413) = 3RUR$	$f^{(1)}(3142) = 3DLD$	$f^{(3)}(3142) = 1URU$

Démonstration. La preuve procède simplement par vérification exhaustive de tous les cas. La figure 9.4 donne pour aider à cette vérification tous les épis montants de taille au plus 4, et deux exemples d'épis montants de taille 8 et 9, dont les cavaliers actifs sont marqués. □

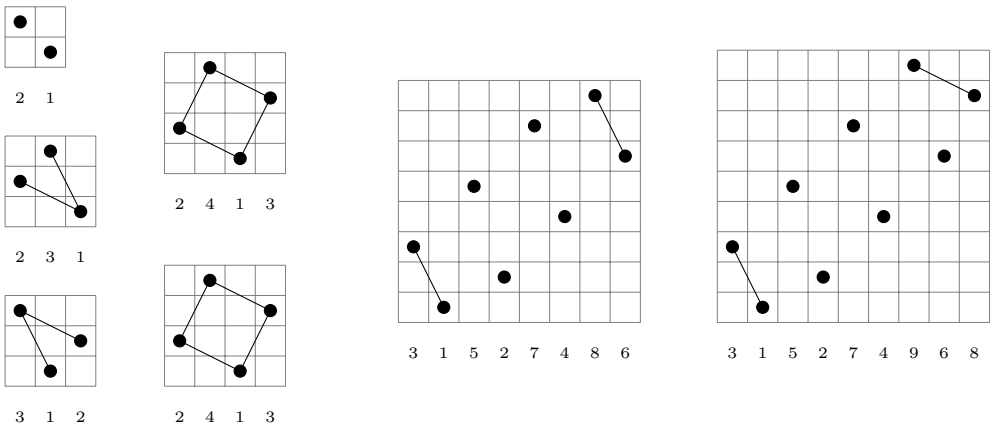


FIG. 9.4 – Les épis montants de taille au plus 4, et deux exemples d'épis montants de taille 8 (de type (V, V)) et 9 (de type (V, H)), dont les cavaliers actifs sont marqués.

Remarque 9.30. Pour les épis ξ descendants, on définirait de même $f^{(2)}(\xi)$ et $f^{(4)}(\xi)$. Un lemme semblable au lemme 9.29 donnerait une expression explicite de $f^{(2)}(\xi)$ et $f^{(4)}(\xi)$.

Pour exprimer les lectures possibles de l'ensemble des deux premiers fils lus, on pose d'abord :

Définition 9.31. Pour tout épi montant ξ, on note $f^{int}(\xi)$ l'ensemble des mots d'épingles codant une représentation en épingles de ξ, sans contrainte sur la position de l'origine (l'origine est interne à ξ).

Lemme 9.32. *On pose* $\mathcal{H} = \{44, 34, 3D, 14, 4U, 24, 42, 32, 2D, 1U, 12, 22\}$ *et* $\mathcal{V} = \{44, 34, 4L, 3L, 14, 24, 42, 32, 1R, 12, 2R, 22\}$. *Soit* ξ *un épi montant, de taille* $n \geq 5$.

Si n *est pair, on pose* $n = 2p+2$, *et alors* $f^{int}(\xi) = \begin{cases} \mathcal{V}(UR)^p \cup \mathcal{V}(DL)^p & \text{si } \xi \text{ est de type } (H,H) \\ \mathcal{H}(RU)^p \cup \mathcal{H}(LD)^p & \text{si } \xi \text{ est de type } (V,V). \end{cases}$

Sinon, on pose $n = 2p+1$, *et alors* $f^{int}(\xi) = \begin{cases} \mathcal{V}U(RU)^{p-1} \cup \mathcal{H}L(DL)^{p-1} & \text{si } \xi \text{ est de type } (H,V) \\ \mathcal{H}R(UR)^{p-1} \cup \mathcal{V}D(LD)^{p-1} & \text{si } \xi \text{ est de type } (V,H). \end{cases}$

En posant $\mathcal{H}' = \{4D, 43, 33, 41, 1D, 13, 31, 23, 3U, 11, 21, 2U\}$ *et* $\mathcal{V}' = \{4R, 43, 3R, 33, 41, 13, 31, 23, 11, 21, 1L, 2L\}$, *on a aussi :*

$$\begin{aligned} &f^{int}(1) = \{1,2,3,4\}, \\ &f^{int}(21) = \mathcal{H} \cup \mathcal{V}, \\ &f^{int}(312) = \mathcal{H}R \cup \mathcal{V}D \cup (\mathcal{H}' \cup \mathcal{V}')2, \\ &f^{int}(231) = \mathcal{H}L \cup \mathcal{V}U \cup (\mathcal{H}' \cup \mathcal{V}')4, \\ &f^{int}(2413) = \mathcal{V}DL \cup \mathcal{V}UR \cup \mathcal{H}'RD \cup \mathcal{H}'LU, \\ &f^{int}(3142) = \mathcal{H}LD \cup \mathcal{H}RU \cup \mathcal{V}'DR \cup \mathcal{V}'UL. \end{aligned}$$

Démonstration. Par un examen de tous les cas possibles, comme sur la figure 9.1, on s'aperçoit facilement que l'ensemble $\mathcal{H}$ (resp. $\mathcal{V}$) correspond aux codages par un mot d'épingles d'une représentation en épingles de 21 qui ne finissent pas par L ou R (resp. U ou D). Ces langages représentent donc les lectures possibles d'un cavalier actif dans un épi montant qui peuvent être suivi par une épingle séparante respectivement horizontale (L ou R) ou verticale (U ou D).

Par une vérification exhaustive, on montre ensuite que la première (resp. la deuxième) partie qui compose chaque langage $f^{int}(\xi)$ décrit les mots d'épingles associées aux représentations en épingles de ξ commençant par le cavalier actif situé dans l'angle en bas à gauche (resp. en haut à droite) de ξ.

Le cas des épis de taille au plus 4 est aussi traité par recherche exhaustive, les ensembles $\mathcal{H}'$ et $\mathcal{V}'$ étant définis de la même façon que $\mathcal{H}$ et $\mathcal{V}$, pour le motif 12 plutôt que 21. □

Définition 9.33. Pour tout couple (ξ_g, ξ_d) d'épis montants, on note $f^{double}(\xi_g, \xi_d)$ l'ensemble des mots d'épingles codant une représentation en épingles de $\oplus[\xi_g, \xi_d]$.

Lemme 9.34. *On définit* $\mathcal{H}'$ *et* $\mathcal{V}'$ *comme dans le lemme 9.32. Pour tout couple* (ξ_g, ξ_d) *d'épis montants, on pose* $f^{d\to g}(\xi_g, \xi_d) = f^{int}(\xi_d)f^{(1)}(\xi_g)$ *et* $f^{g\to d}(\xi_g, \xi_d) = f^{int}(\xi_g)f^{(3)}(\xi_d)$.

En outre, si $\xi_d = 1$ *et* $n = |\xi_g| \geq 4$, *on pose*

$$f^{mix}(\xi_g, 1) = \begin{cases} \mathcal{H}'L(DL)^p & \text{si } \xi_g \text{ est de type } (H,H) \\ \mathcal{V}'D(LD)^p & \text{si } \xi_g \text{ est de type } (V,V) \end{cases} \quad \text{si } n \text{ est pair et } n = 2p+2,$$

$$\text{et } f^{mix}(\xi_g, 1) = \begin{cases} \mathcal{H}'(LD)^p & \text{si } \xi_g \text{ est de type } (V,H) \\ \mathcal{V}'(DL)^p & \text{si } \xi_g \text{ est de type } (H,V) \end{cases} \quad \text{si } n \text{ est impair et } n = 2p+1.$$

Lorsque $|\xi_g| = 2$ *ou* 3, *on pose aussi* $f^{mix}(21, 1) = \mathcal{H}'L \cup \mathcal{V}'D$, $f^{mix}(231, 1) = (\mathcal{H}' \cup \mathcal{V}')1D \cup \mathcal{V}'DL$ *et* $f^{mix}(312, 1) = (\mathcal{H}' \cup \mathcal{V}')1L \cup \mathcal{H}'LD$.

On définit de façon symétrique $f^{mix}(\xi_g, \xi_d)$ *pour* $\xi_g = 1$ *et* $\xi_d \neq 1$.

Dans toute autre situation, c'est-à-dire lorsque soit $\xi_g = \xi_d = 1$, *soit* $\xi_g \neq 1$ *et* $\xi_d \neq 1$, $f^{mix}(\xi_g, \xi_d)$ *est vide.*

Alors $f^{double}(\xi_g, \xi_d) = f^{d\to g}(\xi_g, \xi_d) \cup f^{g\to d}(\xi_g, \xi_d) \cup f^{mix}(\xi_g, \xi_d)$.

Démonstration. Il est évident que $f^{d\to g}(\xi_g, \xi_d)$ (resp. $f^{g\to d}(\xi_g, \xi_d)$) est l'ensemble des mots d'épingles associés à $\oplus[\xi_g, \xi_d]$ qui lisent d'abord ξ_d (resp. ξ_g) entièrement, puis ξ_g (resp. ξ_d) entièrement.

Les situations où il existe des représentations en épingles de $\oplus[\xi_g, \xi_d]$ qui ne procèdent pas par lecture complète de ξ_g et de ξ_d sont identifiées par le lemme 9.26. Par symétrie, on peut se

restreindre au cas où $\xi_d = 1$ et ξ_g est lu en deux fois. Alors nécessairement $\xi_g \neq 1$, et la preuve du lemme 9.42 assure que les lectures en deux fois de ξ_g sont données par les cas suivant de la figure 9.8 :

- $U3+$ pour $S = 1$ lorsque $|\xi_g| \geq 4$ et ξ_g est de type (H, H) ou (V, H),
- $U4+$ pour $S = 1$ lorsque $|\xi_g| \geq 4$ et ξ_g est de type (V, V) ou (H, V),
- $U2$ pour $S = 12$ et $U4$ pour $S = 1$ lorsque $\xi_g = 231$,
- $U1$ pour $S = 12$ et $U3$ pour $S = 1$ lorsque $\xi_g = 312$,
- $U1$ pour $S = 1$ et $U2$ pour $S = 1$ lorsque $\xi_g = 21$.

On vérifie facilement, par exemple sur la figure 9.5, que la définition de f^{mix} traduit dans le cadre des mots d'épingles ces cas particuliers. □

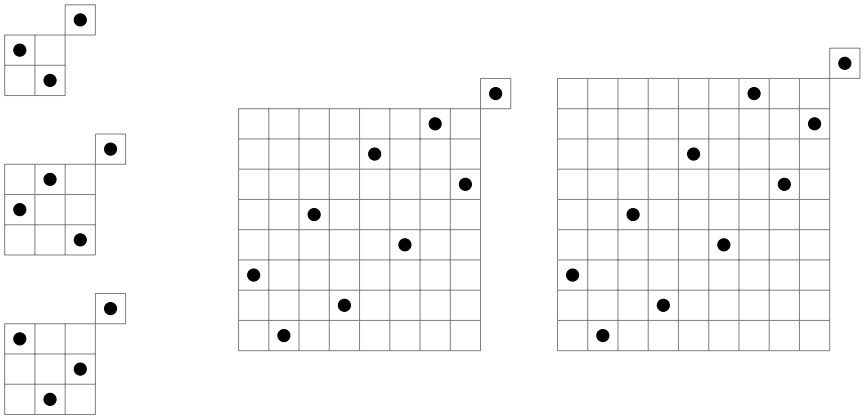


FIG. 9.5 – Les configurations telles que la permutation $\oplus[\xi_g, 1]$ admet des représentations en épingles où ξ_g est lu en deux fois, pour un épi montant ξ_g.

Pour pouvoir conclure sur les ensembles de mots d'épingles codant une permutation $\sigma = \oplus[\xi_1, \xi_2, \ldots, \xi_q]$ où tous les ξ_i sont des épis montants, il reste à définir la notion classique de *produit de mélange* ou *shuffle*.

Définition 9.35. On définit le *produit de mélange* de deux séquences $a = (a_1, a_2, \ldots, a_q)$ et $b = (b_1, b_2, \ldots, b_m)$ par $a ⧢ b = \{c = (c_1, c_2, \ldots, c_{q+m})$ tel que $\{c_i : 1 \leq i \leq q+m\} = \{a_i : 1 \leq i \leq q\} \cup \{b_i : 1 \leq i \leq m\}$ (en tant que multi-ensembles), et la sous-séquence extraite de c contenant les a_i (resp. b_i) est égale à a (resp. b)$\}$.

Proposition 9.36. *L'ensemble des mots d'épingles associés à une permutation $\sigma = \oplus[\xi_1, \xi_2, \ldots, \xi_q]$ où tous les ξ_i sont des épis montants, est*

$$P(\sigma) = \bigcup_{1 \leq j \leq q-1} f^{double}(\xi_j, \xi_{j+1}) \cdot \Big((f^{(1)}(\xi_{j-1}), \ldots, f^{(1)}(\xi_1)) ⧢ (f^{(3)}(\xi_{j+2}), \ldots, f^{(3)}(\xi_q)) \Big).$$

Démonstration. C'est une conséquence directe du lemme 9.26 et de sa conséquence 9.27. □

Les lemmes précédents donnent des expressions explicites de $f^{(1)}(\xi)$, $f^{(3)}(\xi)$ et $f^{double}(\xi_g, \xi_d)$, pour tous les épis montants ξ, ξ_d et ξ_g, et donc une expression explicite de $P(\sigma)$. On verra au paragraphe 9.4 comment cet ensemble de mots d'épingles peut être décrit par un automate calculable en temps polynomial.

On pourrait donner par le même raisonnement des expressions explicites pour $f^{(2)}(\xi)$, $f^{(4)}(\xi)$ et $f^{double}(\xi_g, \xi_d)$, pour tous les épis descendants ξ, ξ_d et ξ_g. On aurait alors :

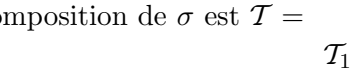
Proposition 9.37. *L'ensemble des mots d'épingles associés à une permutation* $\sigma = \ominus[\xi_1, \xi_2, \ldots, \xi_q]$ *où tous les* ξ_i *sont des épis descendants, est*

$$P(\sigma) = \bigcup_{1 \leq j \leq q-1} f^{double}(\xi_j, \xi_{j+1}) \cdot \Big((f^{(4)}(\xi_{j-1}), \ldots, f^{(4)}(\xi_1)) \sqcup\!\sqcup (f^{(2)}(\xi_{j+2}), \ldots, f^{(2)}(\xi_q)) \Big).$$

9.2.2 Permutations en épingles de racine linéaire construites récursivement

On s'intéresse maintenant aux cas récursifs de l'équation de la page 215, dans le cas où la racine de σ est linéaire. On supposera que σ se décompose sous la forme $\sigma = \oplus[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(q)}]$ où tous les $\alpha^{(i)}$ sauf un sont des épis montants. Par des arguments de symétrie, on peut faire cette hypothèse sans perte de généralité.

On commence par démontrer des propriétés générales sur les permutations en épingles de racine $\oplus$, s'écrivant donc sous la forme $\sigma = \oplus[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(q)}]$. Ces propriétés sont valables dans le cas où tous les $\alpha^{(i)}$ sont des épis montants aussi bien que dans le cas où tous sauf un sont des épis montants.

On considère donc une permutation en épingles $\sigma = \oplus[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(q)}]$ de racine $\oplus$. On note $\mathcal{T}_i$ l'arbre de décomposition de la permutation $\alpha^{(i)}$, pour tout $i \in [1..q]$. Ainsi, l'arbre de décomposition de σ est $\mathcal{T} =$ 

. Les $\mathcal{T}_i$ sont par définition des blocs (c'est-à-dire des intervalles) de σ, mais σ peut contenir d'autres blocs plus petits. Par commodité, et sauf mention explicite du contraire, un bloc désignera un des arbres $\mathcal{T}_i$ dans la suite.

On considère une représentation en épingles $p = (p_1, \ldots, p_n)$ de σ et on note i_0 l'indice du bloc $\mathcal{T}_i$ auquel appartient p_1.

Lemme 9.38. *Pour toute paire d'entiers* $i, j \in [1..q]$ *telle que* $i < j < i_0$ *ou* $i_0 < j < i$, $\mathcal{T}_j$ *est lu entièrement avant que* p *ne commence la lecture de* $\mathcal{T}_i$.

Démonstration. On pose $\ell = \min\{\ell', p_{\ell'} \in \mathcal{T}_i\}$, et on note $\mathcal{B}$ la boîte englobante de $\{p_1, \ldots, p_\ell\}$. Comme $p_1 \in \mathcal{T}_{i_0}$, $\mathcal{T}_j \subset \mathcal{B}$, (voir la figure 9.6) est donc $\mathcal{T}_j$ est lu entièrement par p avant p_ℓ. □

Ce lemme donne l'ordre dans lequel les blocs sont lus. On analyse maintenant comment se comporte p à l'intérieur de chaque bloc $\mathcal{T}_i$, et on caractérise ceux qui peuvent être lus en plusieurs fois[j] par p. On montre en outre que si un bloc est lu en plusieurs fois, alors son arbre de décomposition a une forme bien spécifique, explicitée par le lemme 9.42.

Lemme 9.39. *Soit* $p = (p_1, p_2, \ldots, p_n)$ *une représentation en épingles correspondant à* σ. *Alors pour tout* $k \in \{1 \ldots n\}$, *lorsque* p *a lu les épingles* $(p_1, p_2, \ldots, p_k)$, *il y a au plus un bloc dont la lecture est commencée mais pas encore terminée.*

Démonstration. Le raisonnement pour la preuve du lemme 9.39 est illustré sur la figure 9.6. On suppose que les épingles $\{p_1, \ldots, p_k\}$ ont déjà été lues. D'après le lemme 9.38, il existe au plus un bloc $\mathcal{T}_i$ avec $i < i_0$ et au plus un bloc $\mathcal{T}_m$ avec $m > i_0$ dont la lecture est commencée mais pas terminée. On peut remarquer qu'alors $i = \min\{\ell \mid \exists h \in \{1, \ldots, k\}, p_h \in T_\ell\}$. On peut exprimer la valeur de m par une expression analogue, en remplaçant le minimum par un maximum. Si la lecture de $\mathcal{T}_i$ n'est pas terminée, alors comme $\mathcal{T}_i$ est par définition $\oplus$-indécomposable, il existe nécessairement une épingle p_q dans la zone indiquée en ▨. Une telle épingle est alors sur les côtés de la boîte englobante de $\{p_1, \ldots, p_k\}$. Le même raisonnement pour le bloc $\mathcal{T}_m$ indique la présence d'une épingle séparante dans la zone ▧. D'après le lemme 8.10 du chapitre 8, il y a au plus un point sur les côtés de la boîte englobante, de sorte qu'il y a au plus un bloc dont les k premières épingles de p ont commencé la lecture sans l'avoir encore achevé. □

[j] La définition d'un bloc lu en plusieurs fois a été donnée page 181.

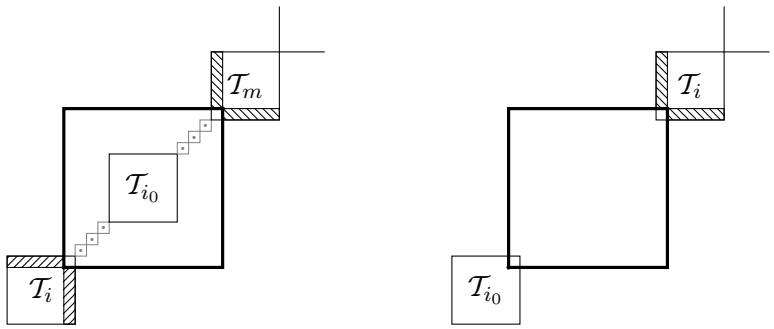


FIG. 9.6 – À gauche, illustration de la démonstration du lemme 9.39. Le rectangle en traits gras représente la boîte englobante de $\{p_1, \ldots, p_k\}$. À droite, illustration de la démonstration du lemme 9.40. Le rectangle en traits gras représente la boîte englobante de $\{p_1, \ldots, p_{k-1}\}$.

Lemme 9.40. *Pour toute représentation en épingles $p = (p_1, p_2, \ldots, p_n)$ de σ, chaque bloc $\mathcal{T}_i$ est lu en une fois par p, sauf peut-être le bloc $\mathcal{T}_{i_0}$.*

Démonstration. On considère un bloc $\mathcal{T}_i$ avec $i \neq i_0$ dont on suppose qu'il est lu en plusieurs fois par p. On considère la première épingle à l'extérieur de $\mathcal{T}_i$ après que p a commencé la lecture de $\mathcal{T}_i$. On note p_k cette épingle. Comme $p_1 \in \mathcal{T}_{i_0}$ est à l'extérieur de $\mathcal{T}_i$, la boîte englobante de $\{p_1, p_2, \ldots, p_{k-1}\}$ intersecte $\mathcal{T}_i$ sans la contenir complètement (voir la figure 9.6). Comme $\mathcal{T}_i$ est $\oplus$-indécomposable, il doit exister une épingle dans la zone ▨. Comme cette épingle est sur les côtés de la boîte englobante de $\{p_1, p_2, \ldots, p_{k-1}\}$, le lemme 8.10 du chapitre 8 assure que ce point est p_k. Ainsi, $p_k \in \mathcal{T}_i$, ce qui apporte une contradiction à la définition de p_k. □

Conséquence 9.41. *Si $\mathcal{T}_{i_0}$ est lu en plusieurs fois, par exemple par les épingles $p_1, \ldots p_\ell$ puis $p_k, \ldots$ avec $k > \ell$, alors chacune des épingles $(p_j)_{\ell<j<k}$ forme à elle seule un intervalle fort maximal de σ (c'est-à-dire correspond à un bloc $\mathcal{T}_i$).*

Démonstration. En effet, si on suppose le contraire pour une telle épingle p_j, alors la lecture de p_j laisserait deux blocs $\mathcal{T}_i$ avec une lecture inachevée : le bloc auquel p_j appartient, et $\mathcal{T}_{i_0}$. Ceci contredit le lemme 9.39. □

Lemme 9.42. *Les seules permutations en épingles σ de racine $\oplus$ pour lesquelles un bloc $\mathcal{T}_{i_0}$ peut être lu en plusieurs fois sont celles dont les arbres de décomposition sont représentés sur la figure 9.9. La figure 9.8 donne aussi une vision schématique de la représentation graphique de ces permutations.*

En outre, pour chacune de ces permutations, si $\mathcal{T}_{i_0}$ est lu en plusieurs fois, alors $\mathcal{T}_{i_0}$ est lu en deux fois, la première partie de $\mathcal{T}_{i_0}$ qui est lue est S, la seconde est composée de l'ensemble des points restants dans $\mathcal{T}_{i_0}$ (qui sont des feuilles), et ces deux parties de la lecture de $\mathcal{T}_{i_0}$ sont séparées seulement par la lecture de la feuille x.

Démonstration. Soit $p = (p_1, p_2, \ldots, p_n)$ une représentation en épingles de σ où $\mathcal{T}_{i_0}$ est lu en plusieurs fois. On note $p_1, \ldots, p_\ell$ les épingles correspondant à la première partie de la lecture de $\mathcal{T}_{i_0}$. Alors les épingles $p_{\ell+1}, \ldots, p_{m-1}$ appartiennent à d'autres blocs $\mathcal{T}_i$, jusqu'à $p_m \in \mathcal{T}_{i_0}$.

Comme $\sigma = \oplus[\mathcal{T}_1, \ldots, \mathcal{T}_q]$, toutes les épingles p_i pour $i \in \{\ell+1, \ell+2, \ldots, m-1\}$, qui n'appartiennent pas à $\mathcal{T}_{i_0}$, se trouvent dans une des zones hachurées ▨ de la figure 9.7. Si ces deux zones contenaient chacune au moins un point p_i pour $i \in \{\ell+1, \ell+2, \ldots, m-1\}$, alors $\mathcal{T}_{i_0}$ serait inclus dans la boîte englobante de $\{p_1, \ldots, p_{m-1}\}$, interdisant à l'épingle p_m de satisfaire la condition d'extériorité, et apportant donc une contradiction à sa définition. Ainsi, toutes les épingles p_i pour $i \in \{\ell+1, \ell+2, \ldots, m-1\}$ sont dans la même zone hachurée, par exemple celle en haut à droite.

Si p_m respecte la condition d'indépendance, alors p_m appartenant à $\mathcal{T}_{i_0}$, il doit être dans le quadrant 3 par rapport à la boîte englobante de $\{p_1, \ldots, p_l\}$, de même que tous les points restants dans

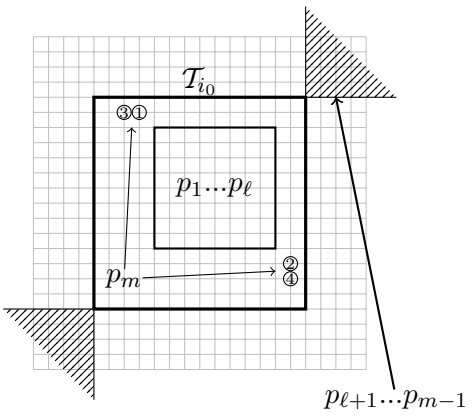


FIG. 9.7 – Le bloc $\mathcal{T}_{i_0}$ est lu en deux fois.

$\mathcal{T}_{i_0}$. Ceci conduit à un bloc $\mathcal{T}_{i_0}$ qui est $\oplus$-décomposable, ce qui est impossible dans la décomposition par substitution de σ en $\oplus[\mathcal{T}_1, \dots, \mathcal{T}_q]$. Ainsi, p_m est une épingle séparante.

Si $m - 1 - \ell \geq 2$, c'est-à-dire s'il y a au moins deux points p_i avec $i \in \{\ell + 1, \ell + 2, \dots, m - 1\}$ dans la zone hachurée, alors on s'aperçoit que $p_m \notin \mathcal{T}_{i_0}$. En effet, dans ce cas p_m sépare deux points appartenant à $\cup_{i>i_0} \mathcal{T}_i$, et cet ensemble est un intervalle de σ : la remarque 8.54 (page 197) assure alors que $p_m \in \cup_{i>i_0} \mathcal{T}_i$, ce qui interdit d'avoir $p_m \in \mathcal{T}_{i_0}$. On conclut donc que $m - 1 - \ell = 1$, c'est-à-dire qu'il y a un unique point de la zone hachurée qui est lu entre $p_1, \dots, p_\ell$ et p_m. Ce point est noté x.

D'autre part, la conséquence 9.41 assure que x forme à lui seul un intervalle fort maximal de σ. Comme p_m sépare x de $\{p_1, \dots, p_l\}$, c'est le seul point sur les côtés de la boîte englobante de $\{p_1, \dots, p_l, x\}$. Pour satisfaire cette contrainte, il est nécessaire que x se trouve dans l'angle en bas à gauche de la zone hachurée.

On rappelle que la distance entre une épingle séparante et la boîte englobante des points précédents est au plus 2 (voir page 176). La position de x laisse donc quatre positions possibles pour p_m, numérotées de ① à ④ sur la figure 9.7.

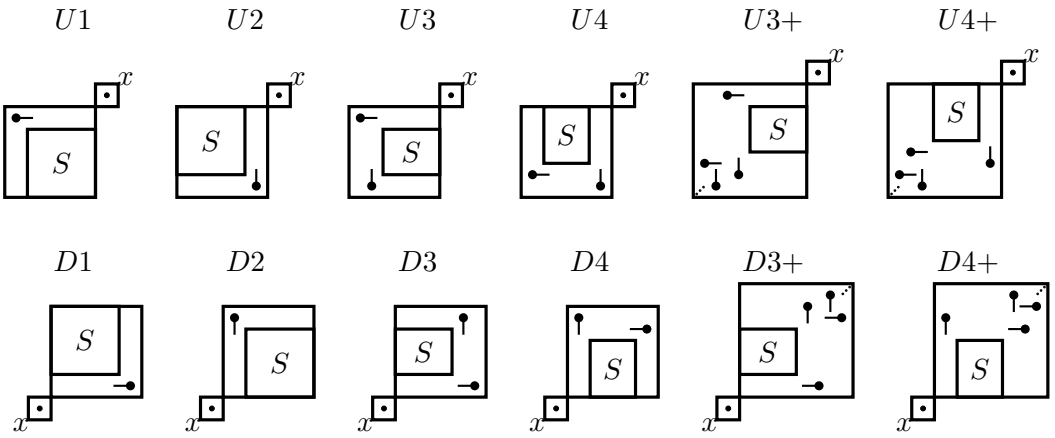


FIG. 9.8 – Les douze formes possibles d'un bloc $\mathcal{T}_{i_0}$ dans une permutation en épingles σ de racine $\oplus$ conduisant à la possibilité d'une lecture en deux fois de $\mathcal{T}_{i_0}$. Sur chaque figure, les lectures en deux fois de $\mathcal{T}_{i_0}$ ont toutes pour première partie S, et pour deuxième partie les points restant dans le bloc, avec seulement la lecture de x qui s'intercale entre les deux. Il faut remarquer que si la condition $U4$ (resp. $U3, D4, D3$) est satisfaite, alors la condition $U2$ (resp. $U1, D2, D1$) l'est aussi, pour un bloc S augmenté d'un point.

Si p_m est en position ① (cas $U1$ de la figure 9.8) ou ② (cas $U2$), alors les épingles $\{p_1, p_2, \ldots, p_l, p_m\}$ forment un intervalle, et décrivent donc l'intégralité de $\mathcal{T}_{i_0}$ (puisque $\mathcal{T}_{i_0}$ est $\oplus$-indécomposable). Sinon, p_m est en position ③ (cas $U3$ et $U3+$) ou ④ (cas $U4$ et $U4+$). Par symétrie, on peut supposer que p_m est en position ④. Alors p_{m+1} est une épingle séparante de type L, qui a à nouveau deux positions possibles : à distance 1 (ce qui termine le bloc $\mathcal{T}_{i_0}$ et correspond au cas $U4$) ou à distance 2 de la boîte englobante des points précédents. Ce procédé peut être répété en alternant des épingles séparantes de type L et D, jusqu'à ce qu'une épingle, par exemple p_{m+k} soit à distance 1 de la boîte englobante des points précédents, et termine le bloc $\mathcal{T}_{i_0}$.

En examinant le cas symétrique où c'est la zone hachurée en bas à gauche qui contient les points p_i pour $i \in \{\ell+1, \ell+2, \ldots, m-1\}$, on obtient les cas $D1$ à $D4+$ de la figure 9.8.

Ceci démontre que $\mathcal{T}_{i_0}$ est lu en exactement deux fois, la première partie de la lecture étant réduite à $\{p_1, \ldots, p_\ell\}$ et la seconde à $\{p_m, \ldots, p_{m+k}\}$, et qu'en outre seul x est lu entre ces deux parties de $\mathcal{T}_{i_0}$. Le passage aux arbres de décomposition est évident pour chaque cas apparaissant dans la preuve. □

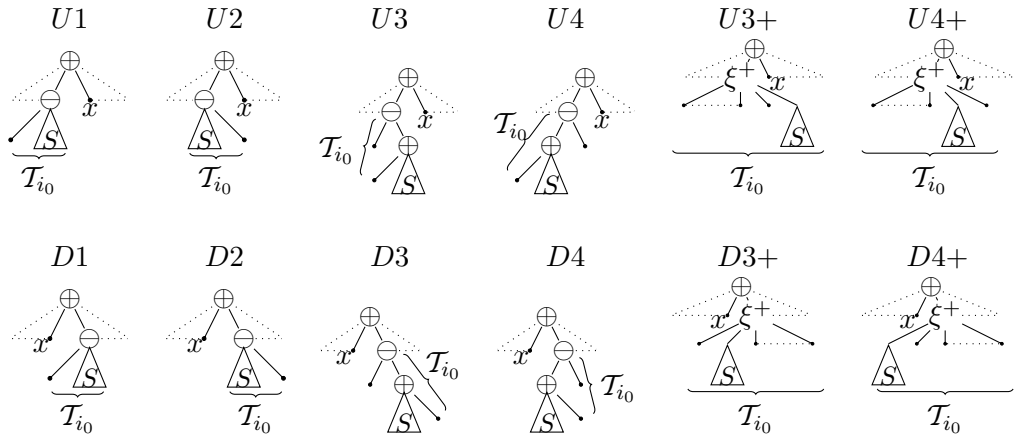


FIG. 9.9 – Les douze formes possibles de l'arbre de décomposition $\mathcal{T}$ d'une permutation en épingles σ de racine $\oplus$ pour lesquelles il existe un bloc $\mathcal{T}_{i_0}$ et une représentation en épingles de σ qui lit $\mathcal{T}_{i_0}$ en deux fois. Le symbole ξ^+ représente n'importe quel épi montant de taille au moins 4 (qui est donc une permutation simple).

On peut remarquer que la preuve du lemme 9.42 démontre aussi que l'ordre de lecture des points correspondant aux feuilles de $\mathcal{T}_{i_0} \setminus S$ est uniquement déterminé. Ceci justifie la remarque suivante :

Remarque 9.43. Dans chaque cas du lemme 9.42, si $\mathcal{T}_{i_0}$ est lu en deux fois, alors la première partie de la lecture de $\mathcal{T}_{i_0}$ (c'est-à-dire S) étant fixée, l'ordre de lecture des points restants de $\mathcal{T}_{i_0}$ est déterminé.

La partie S qui autorise la lecture de $\mathcal{T}_{i_0}$ en deux fois n'est cependant pas déterminée, puisque certaines permutations satisfont plusieurs des conditions $U1$ à $D4+$ de la figure 9.8. Les combinaisons possibles de ces conditions entre elles sont données, à symétrie près, par les deux premières colonnes de la figure 9.10. Ces combinaisons sont considérées comme mutuellement exclusives : par exemple, si une permutation satisfait la condition $2H2$, on dira qu'elle ne satisfait pas $1H2$, pourtant *a priori* incluse dans $2H2$.

Une vérification exhaustive démontre que toutes les combinaisons possibles apparaissent bien sur la figure 9.10, à symétrie près. La troisième colonne montre les arbres de décomposition correspondants. Les automates de la quatrième colonne seront commentés dans le paragraphe 9.4.

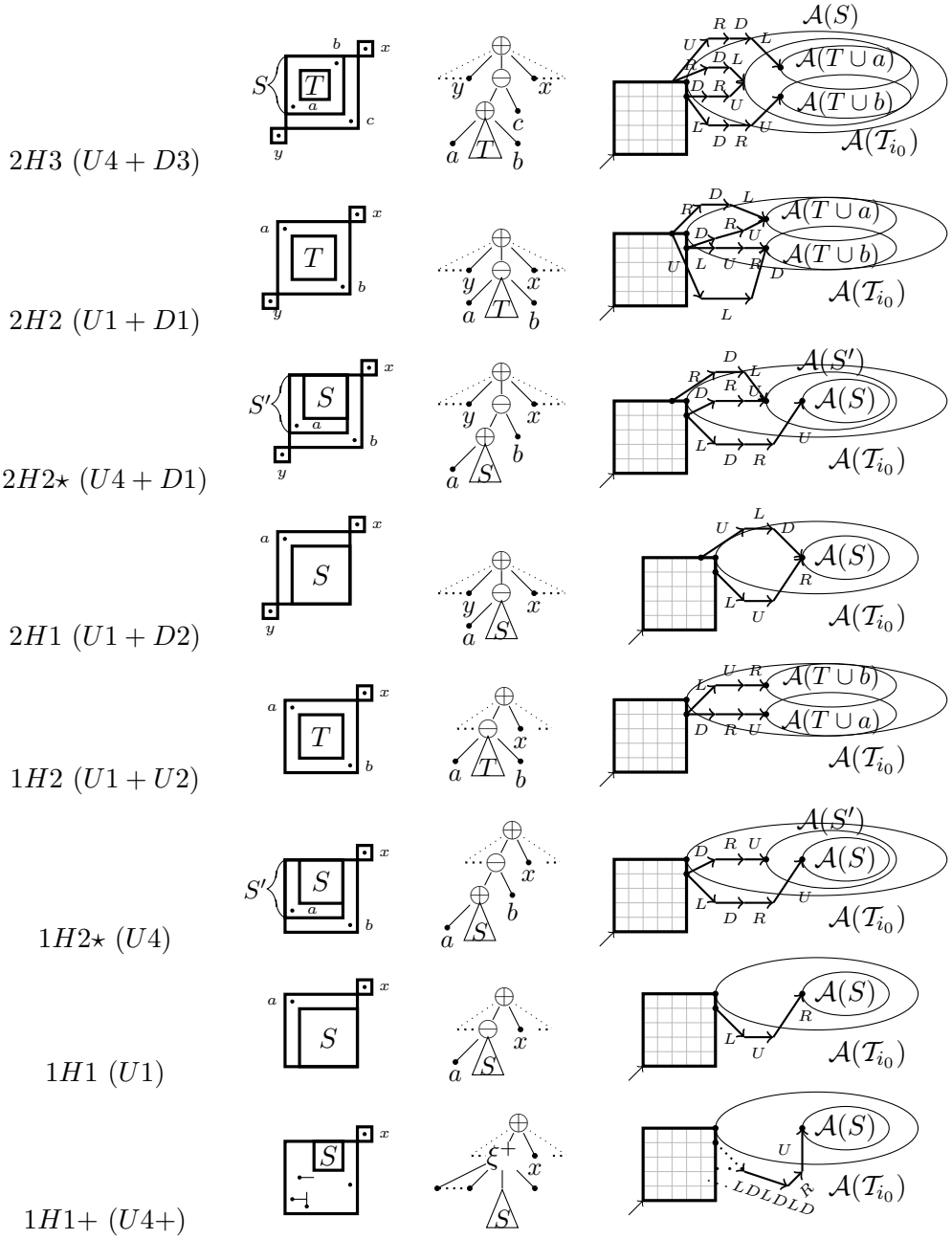
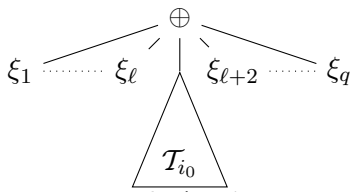


FIG. 9.10 – Les huit formes possibles (à symétrie près) d'un bloc $\mathcal{T}_{i_0}$ dans une permutation en épingles σ qui peut être lu en deux fois par une ou par plusieurs représentations en épingles de σ. Pour que ces conditions soient mutuellement exclusives, on dira qu'une permutation σ satisfait une des conditions iHj seulement lorsqu'aucune des autres conditions au-dessus de iHj sur la figure n'est satisfaite par σ (à symétrie près).

Avec les expressions de $f^{(1)}$ et $f^{(3)}$ donnant les mots d'épingles de tout épi montant ξ qui codent une représentation en épingles de ξ dont l'origine se trouve dans le quadrant 1 ou 3 (voir page 217), on dispose de tous les outils pour décrire l'ensemble $P(\sigma)$ des mots d'épingles de σ, pour toute permutation en épingles $\sigma = \oplus[\alpha^{(1)}, \ldots, \alpha^{(q)}]$ où tous les $\alpha^{(i)}$ sauf un sont des épis montants.

Théorème 9.44.

Soit $\sigma =$  *une permutation en épingles* $\oplus$*-décomposable dont un unique bloc fort maximal* $\mathcal{T}_{i_0}$ *n'est pas un épi montant. L'ensemble* $P(\sigma)$ *des mots d'épingles qui codent une représentation en épingles de* σ *est donné par les points suivants.*

On pose $P_0 = P(\mathcal{T}_{i_0}) \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big)$.

– *Si* σ *ne satisfait aucune des conditions de la figure 9.10 (à symétrie près), alors* $P(\sigma) = P_0$.

– *Si* σ *satisfait la condition* $(1H1)$ *alors* $P(\sigma) = P_0 \cup P_1$, *avec*

$$P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big).$$

– *Si* σ *satisfait la condition* $(1H1+)$ *alors* $P(\sigma) = P_0 \cup P_1$, *avec*

$$P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot w}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big),$$

où w *est l'unique mot qui code l'unique lecture par une représentation en épingles des feuilles restantes dans* $\mathcal{T}_{i_0}$.

– *Si* σ *satisfait la condition* $(1H2)$ *alors* $P(\sigma) = P_0 \cup P_1 \cup P_2$, *avec*

$\star$ $P_1 = \underbrace{P(T \cup a) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star$ $P_2 = \underbrace{P(T \cup b) \cdot \underbrace{1}_{x} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big).$

– *Si* σ *satisfait la condition* $(1H2\star)$ *alors* $P(\sigma) = P_0 \cup P_1 \cup P_2$, *avec*

$\star$ $P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star$ $P_2 = \underbrace{P(S') \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big).$

– *Si* σ *satisfait la condition* $(2H1)$ *alors* $P(\sigma) = P_0 \cup P_1 \cup P_2$, *avec*

$\star$ $P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star$ $P_2 = \underbrace{P(S) \cdot \underbrace{3}_{y} \cdot \underbrace{U}_{a}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big).$

– *Si* σ *satisfait la condition* $(2H2\star)$ *alors* $P(\sigma) = P_0 \cup P_1 \cup P_2 \cup P_3$, *avec*

$\star$ $P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_2 = \underbrace{P(S') \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_3 = \underbrace{P(S') \cdot \underbrace{3}_{y} \cdot \underbrace{R}_{b}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big).$

– *Si σ satisfait la condition (2H2) alors $P(\sigma) = P_0 \cup P_1 \cup P_2 \cup P_3 \cup P_4$, avec*

$\star\ P_1 = \underbrace{P(T \cup a) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{b}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_2 = \underbrace{P(T \cup b) \cdot \underbrace{1}_{x} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_3 = \underbrace{P(T \cup a) \cdot \underbrace{3}_{y} \cdot \underbrace{R}_{b}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_4 = \underbrace{P(T \cup b) \cdot \underbrace{3}_{y} \cdot \underbrace{U}_{a}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big).$

– *Si σ satisfait la condition (2H3) alors $P(\sigma) = P_0 \cup P_1 \cup P_2 \cup P_3 \cup P_4$, avec*

$\star\ P_1 = \underbrace{P(S) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{c}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_2 = \underbrace{P(T \cup b) \cdot \underbrace{1}_{x} \cdot \underbrace{D}_{c} \cdot \underbrace{L}_{a}}_{x \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_\ell), f^{(1)}(\xi_{\ell-1}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+3}), f^{(3)}(\xi_{\ell+4}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_3 = \underbrace{P(T \cup a) \cdot \underbrace{3}_{y} \cdot \underbrace{R}_{c} \cdot \underbrace{U}_{b}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big)$

et

$\star\ P_4 = \underbrace{P(S) \cdot \underbrace{3}_{y} \cdot \underbrace{R}_{c}}_{y \bigcup \mathcal{T}_{i_0}} \cdot \big(f^{(1)}(\xi_{\ell-1}), f^{(1)}(\xi_{\ell-2}), \dots, f^{(1)}(\xi_1)\big) \sqcup\!\sqcup \big(f^{(3)}(\xi_{\ell+2}), f^{(3)}(\xi_{\ell+3}), \dots, f^{(3)}(\xi_q)\big).$

Démonstration. Il est facile de voir que dans chaque cas du théorème 9.44 les ensembles de mots d'épingles proposés correspondent à des codages de représentations en épingles de σ. Réciproquement, et pour chaque cas du théorème, on démontre que tout mot d'épingles codant une représentation en épingles de σ appartient à l'ensemble donné.

On considère une représentation en épingles p de σ. D'après la remarque 8.33, $\mathcal{T}_{i_0}$ est nécessairement le premier bloc lu par p.

Si σ ne satisfait aucune des conditions $U1$ à $D4+$ de la figure 9.8, alors les lemmes 9.42 et 9.40 assurent que p lit entièrement $\mathcal{T}_{i_0}$, puis les blocs $\mathcal{T}_i$ restants dans un ordre qui va en s'éloignant de $\mathcal{T}_{i_0}$. Comme dans la proposition 9.36, les ordres dans lesquels les blocs $(\mathcal{T}_i)_{i \neq i_0}$ peuvent être lus correspondent à un produit de mélange. On a bien alors $P(\sigma) \subseteq P_0$, et donc $P(\sigma) = P_0$.

Si le bloc $\mathcal{T}_{i_0}$ de σ satisfait certaines des conditions de la figure 9.8, les représentations en épingles de σ qui lisent $\mathcal{T}_{i_0}$ en une fois appartiennent de la même façon à P_0, mais ce ne sont pas les seules représentations en épingles de σ. En effet, le lemme 9.40 assure qu'il existe des représentations en épingles de σ qui lisent $\mathcal{T}_{i_0}$ en deux fois. D'après le lemme 9.40, une telle représentation en épingles correspondant à une des conditions de la figure 9.8 procède ainsi : lecture de S, puis du point x, puis des feuilles restantes dans $\mathcal{T}_{i_0}$, dans un ordre déterminé par S (remarque 9.43).

Les différentes configurations de la figure 9.10 représentent, à symétrie près, les blocs $\mathcal{T}_{i_0}$ qui satisfont certaines des conditions de la figure 9.8. Pour chacune de ces configurations, on a donc un ensemble de mots d'épingles codant σ par condition de la figure 9.8 satisfaite, où S est déterminé. Par exemple, la condition $2H3$ correspond aux blocs $\mathcal{T}_{i_0}$ satisfaisant les conditions

- $U4$ pour $S = T \cup b$,
- $D3$ pour $S = T \cup a$,
- $U2$ pour $S = T \cup a \cup b$,
- $D1$ pour $S = T \cup a \cup b$.

D'autre part, comme $\mathcal{T}_{i_0}$ n'est pas un épi montant, si σ satisfait une condition de la figure 9.10, alors les ensembles T et S définis dans ces différentes conditions sont tels que $|T| \geq 1$ et $|S| \geq 2$. Ainsi, non seulement toute représentation en épingles de S peut être prolongée de manière unique en une représentation en épingles de σ, mais le fait que $\mathcal{T}_{i_0}$ ne soit pas un épi implique en outre que tous les mots d'épingles qui codent ces représentations en épingles ont un même suffixe en commun, qui correspond à la lecture de x et de toutes les feuilles $\mathcal{T}_{i_0} \setminus S$.

Dans le cas des conditions $2H3, 2H2$ et $1H2$, cette remarque est aussi valable lorsque $T \cup a$ et $T \cup b$ jouent le rôle de S, de même pour S' dans les conditions $2H2\star$ et $1H2\star$.

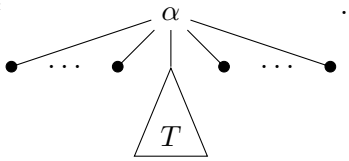
C'est ainsi que sont définis les ensembles P_1, P_2, P_3 et P_4 : les mots d'épingles possibles d'un bloc S (qui peut être égal à S, $T \cup a$, $T \cup b$ ou S', selon les configurations de la figure 9.10), puis la suite du mot d'épingles codant la lecture de x (ou y) et des feuilles restantes dans $\mathcal{T}_{i_0}$, qui est commune à toutes les lectures possibles de S. □

Dans le cas des permutations en épingles σ qui sont $\ominus$-décomposable, on a un analogue du théorème 9.44, où les épis montants sont remplacés par des épis descendants, et les fonctions $f^{(1)}$ et $f^{(3)}$ par $f^{(4)}$ et $f^{(2)}$ respectivement.

La description de ces ensembles de mots d'épingles $P(\sigma)$ codant une permutation en épingles σ de racine linéaire permettra de construire un automate reconnaissant $P(\sigma)$, et on verra au paragraphe 9.4 que cette construction peut être faite en temps polynomial.

9.2.3 Permutations en épingles de racine première construites récursivement

Il reste à examiner les cas récursifs de l'équation de la page 215, dans le cas où la racine de σ est première, étiquetée par une permutation simple α.

On considère d'abord le cas où σ se décompose sous la forme $\sigma = \alpha[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(q)}]$ où tous les $\alpha^{(i)}$ sauf un sont des feuilles. On note T l'unique $\alpha^{(i)}$ qui n'est pas une feuille. Au niveau des arbres de décomposition, $\sigma =$ 

.

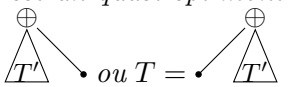
On donne d'abord une condition nécessaire satisfaite par les représentations en épingles de σ.

Lemme 9.45. *Soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ. Alors p satisfait l'une des conditions suivantes :*

(1) *T est lu en une fois par p. Dans ce cas, on a nécessairement $p_1 \in T$.*

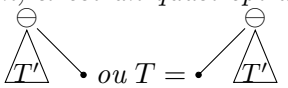
(2) *T est lu en deux fois par p. Dans ce cas, la deuxième (et dernière) partie de la lecture de T est réduite à p_n, α est un quasi-épi, et T dilate un point auxiliaire de α. En outre, σ satisfait les conditions qui suivent.*

– *Si α est un quasi-épi montant, alors*



, en fonction de la position du point de substitution auxiliaire de α. La feuille représentée explicitement est p_n et le cas où p_n est la feuille la plus à gauche (resp. la plus à droite) dans T correspond au cas où le point de substitution auxiliaire de α est à distance 1 de la case en bas à gauche (resp. en haut à droite) sur la représentation graphique de α.

– *Sinon, α est un quasi-épi descendant et*



, en fonction de la position du point de substitution auxiliaire de α. La feuille représentée explicitement est p_n et le cas où p_n est la feuille la plus à gauche (resp. la plus à droite) dans T correspond au cas où le point de substitution auxiliaire de α est à distance 1 de la case en haut à gauche (resp. en bas à droite) sur la représentation graphique de α.

Enfin, si $p_1 \notin T$ alors $T = \{p_2, p_n\}$, et la représentation en épingles p qui satisfait ces conditions est unique.

Démonstration. La conséquence 8.38(i) impose que T est lu en une fois ou en deux fois par p.

(1) On suppose que T est lu en une fois par p. Si $p_1 \notin T$, alors le lemme 8.39 impose que $|T| = 2$ et que T est lu en deux fois par p, contradiction. Donc $p_1 \in T$.

(2) On suppose maintenant que T est lu en deux fois par p. Le lemme 8.37 impose que la deuxième partie de la lecture de T est réduite à p_n.

$\star$ Si $p_1 \notin T$, alors le lemme 8.39 associé au lemme 8.35 assure que T est le deuxième fils de α lu par p, et donc que $p_2 \in T$ (tous les fils de α sauf T étant des feuilles). Le lemme 8.39 assure aussi que $|T| = 2$, et ainsi, $T = \{p_2, p_n\}$.

À symétrie près, on peut toujours supposer que les points p_1 et p_2 forment un motif 12 dans σ. Comme $\{p_2, p_n\}$ est un bloc dans σ, p_n se trouve dans l'une des quatre positions indiquées sur la figure 9.11. La position ③ est impossible, car p_n serait alors à l'intérieur de la boîte englobante B de $\{p_1, p_2\}$. Les positions ④ et ② se trouvent sur les côtés de B, ainsi un point se trouvant dans une de ces cases est nécessairement p_3 (voir le lemme 8.10). Comme $n > 3$ (puisque α est simple), cela implique que p_n occupe la position ①, et donc que $T = 12$.

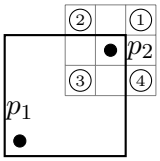


FIG. 9.11 – Positions possibles de p_n dans la preuve du lemme 9.45.

Comme α est une permutation simple, p_3 est une épingle séparante. Si p_3 était de type U ou R, alors p_n serait sur les côtés de la boîte englobante de $\{p_1, p_2, p_3\}$, et on aurait $p_n = p_4$. Comme α est simple, et donc de taille au moins 4, ce dernier point apporte une contradiction. Donc p_3 est de type L ou D. Par symétrie, on peut supposer sans perte de généralité que p_3 est de type D, comme représenté sur la figure 9.12, figure de gauche. Toutes les épingles p_i pour $i \in [3..(n-2)]$ sont séparantes (puisqu'α est simple), et par le même argument que précédemment, sont de type L ou D, nécessairement en alternance. L'épingle p_{n-1} est elle aussi séparante. Pour que p_n sépare p_{n-1} de $\{p_1, p_2, \ldots, p_{n-2}\}$, il est nécessaire que p_{n-1} soit de type U ou R, et ce type est déterminé par la parité de n. Ceci démontre que :

– α est un quasi-épi,
– T dilate un point auxiliaire de α (uniquement déterminé si et seulement si $|\alpha| \geq 6$),
– $T = 12$ ou 21 selon que α est un quasi-épi montant ou descendant,
– la représentation en épingles p de σ telle que $T = \{p_2, p_n\}$ est uniquement déterminée.

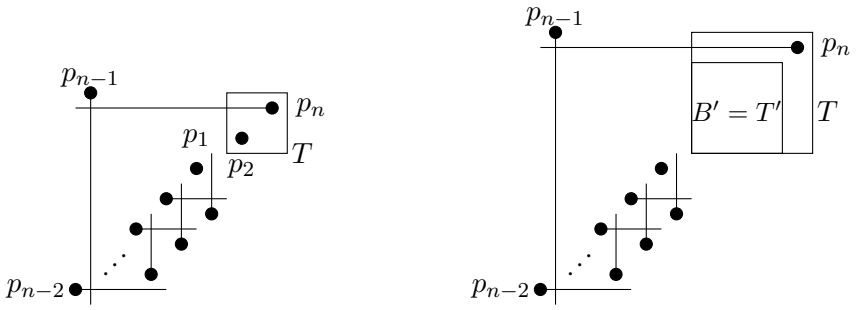


FIG. 9.12 – La forme des permutations σ ayant une racine première dont un seul fils T n'est pas une feuille, et telles qu'il existe une représentation en épingles de σ qui lit T en deux fois.

$\star$ Si maintenant $p_1 \in T$, T étant toujours lu en deux fois par p, alors il existe un indice $i \neq n-1$ tel que $T = \{p_1, \ldots, p_i\} \cup \{p_n\}$.

Si $i \geq 2$, on note B' la boîte englobante de $\{p_1, \ldots, p_i\}$, qui contient donc au moins deux points. Comme $p_n \neq p_{i+1}$, p_n ne peut pas se trouver sur les côtés de la boîte englobante de B'. Par des arguments de symétrie, on peut donc supposer que p_n se trouve le quadrant 1 par rapport à B', comme illustré sur la figure 9.12, figure de droite. Si $i = 1$, alors $T = \{p_1, p_n\}$ contient seulement 2 points, et on peut aussi supposer sans perte de généralité que p_n est dans le quadrant 1 par rapport à p_1.

Comme $T = \{p_1, \ldots, p_i\} \cup \{p_n\}$ est un bloc (de taille au moins 2), aucun point ne peut se trouver sur ses côtés, donc p_{i+1} est une épingle indépendante. En outre, p_{i+1} ne peut pas se trouver dans le quadrant 1 par rapport à T, sinon p_n ne pourrait pas satisfaire la condition d'extériorité. Si p_{i+1} se trouvait dans le quadrant 2 ou 4 par rapport à T, alors p_n serait sur les côtés de la boîte englobante de $\{p_1, \ldots p_{i+1}\}$, et on aurait donc $p_{i+2} = p_n$, contredisant que α est simple. Ainsi, p_{i+1} se trouve dans le quadrant 3 par rapport à T. De la même façon, pour tout $j \in [(i+2)..(n-2)]$, l'épingle p_j se trouve dans le quadrant 3 par rapport à T, et p_j est séparante puisque α est simple. Les épingles p_{i+2} à p_{n-2} sont donc une alternance d'épingles de type L et D, jusqu'à p_{n-1} qui est de type R ou U, pour que p_n puisse séparer $\{p_1, \ldots, p_{n-2}\}$ de p_{n-1}. On conclut donc que α est un quasi-épi, dont T dilate un point de substitution auxiliaire (déterminé uniquement dès que $|\alpha| \geq 6$). En outre, T est $\oplus$-décomposable lorsque α est un quasi-épi montant, et est $\ominus$-décomposable lorsque α est un quasi-épi descendant. □

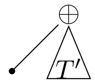
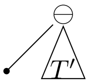
Pour décrire l'ensemble des mots d'épingles associés à σ, il est commode d'introduire la définition suivante :

Définition 9.46. Pour toute permutation en épingles simple α, dont un point actif x est marqué, on considère l'ensemble des mots d'épingles quasi-stricts (c'est-à-dire commençant par deux chiffres) de α qui codent des représentations en épingles $p = (p_1, p_2, \ldots, p_n)$ de α telles que $p_1 = x$. On note alors $Q_x(\alpha)$ l'ensemble des mots d'épingles stricts obtenus à partir de ces mots d'épingles en effaçant leur première lettre.

Les mots d'épingles qui codent une permutation en épingles σ dont l'arbre de décomposition a une racine première avec un unique fils non feuille T sont alors caractérisés comme suit.

Théorème 9.47. *Soit σ une permutation en épingles dont l'arbre de décomposition a une racine première étiquetée par une permutation simple α, avec un unique fils non feuille T. On note x le point de α dilaté par T.*

On définit la condition ($\mathcal{C}$) de la manière suivante :

($\mathcal{C}$) $\begin{cases} \alpha \textit{ est un quasi-épi montant (resp. descendant),} \\ T \textit{ dilate un point de substitution auxiliaire de } \alpha, \\ T \textit{ est de la forme } \end{cases}$  *(resp.*  *) avec la feuille explicitée du côté déterminé par la position du point auxiliaire dilaté de α, comme dans le lemme 9.45.*

Les points suivants sont alors vérifiés.

- *Si ($\mathcal{C}$) n'est pas satisfaite, alors $P(\sigma) = P(T) \cdot Q_x(\alpha)$.*
- *Si ($\mathcal{C}$) est satisfaite, on distingue deux sous-cas en fonction du nombre de feuilles $|T|$ de T.*

  (*a*) *Si $|T| \geq 3$, on pose $\begin{cases} P_1 = P(T) \cdot Q_x(\alpha) \\ P_2 = P(T') \cdot w_\sigma \end{cases}$, où w_σ est l'unique mot qui code l'unique lecture par une représentation en épingles lisant T en deux fois des feuilles restantes dans σ après la lecture de T'. Alors $P(\sigma) = P_1 \cup P_2$.*

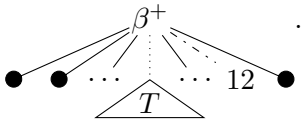
  (*b*) *Si $|T| = 2$, on pose $P_1 = P(T) \cdot Q_x(\alpha)$, et on définit $P_{\{1,n\}}(\sigma)$ (resp. $P_{\{2,n\}}(\sigma)$) comme l'ensemble des mots d'épingles qui codent l'unique représentation en épingles p de σ telle que $T = \{p_1, p_n\}$ (resp. $T = \{p_2, p_n\}$). Alors $P(\sigma) = P_1 \cup P_{\{1,n\}} \cup P_{\{2,n\}}$.*

Démonstration. On suppose d'abord que $\mathcal{C}$ n'est pas satisfaite.
Soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ. D'après le lemme 9.45, T est lu en une fois par p, et $p_1 \in T$. On note $i \geq 2$ l'indice tel que $T = \{p_1, \ldots, p_i\}$. L'épingle p_{i+1} est nécessairement une épingle indépendante, puisqu'une séparante appartiendrait à T qui est un bloc de σ. Alors $q = (p_i, p_{i+1}, \ldots, p_n)$ est une représentation en épingles de α où p_{i+1} doit être une épingle indépendante (pour une origine fictive q_0 de q positionnée à la place de p_1). Il faut donc considérer seulement les mots d'épingles quasi-stricts codant q. Ainsi, pour tout mot d'épingles w codant p, $w \in P(T) \cdot Q_x(\alpha)$.
Réciproquement, si $w \in P(T) \cdot Q_x(\alpha)$, alors il est clair que w est un mot d'épingles qui code σ.

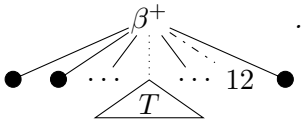
On suppose à présent que $\mathcal{C}$ est satisfaite.
Soit $p = (p_1, \ldots, p_n)$ une représentation en épingles de σ.
⋆ Si T est lu en une fois par p, alors comme ci-dessus on a $w \in P(T) \cdot Q_x(\alpha)$ pour tout mot d'épingles w codant p.
⋆ Sinon, T est lu en deux fois par p, selon les conditions du lemme 9.45.
(*a*) Si $|T| \geq 3$, alors le lemme 9.45 assure que $p_1 \in T$, et qu'il existe un indice $i \geq 2$ tel que $T = \{p_1, \ldots, p_i, p_n\}$ et $T' = \{p_1, \ldots, p_i\}$. Les épingles p_{i+1} à p_{n-1} correspondent à la lecture des feuilles restantes sous α. Comme $|T'| \geq 2$, l'épingle p_{i+1} est indépendante, puisque n'appartenant pas au bloc T. La lettre codant p_{i+1} est donc déterminée indépendamment du contenu de T'. De même, d'après la preuve du lemme 9.45, toutes les épingles p_{i+2} à p_n sont séparantes, et ne dépendent pas du contenu de T'. Le mot w qui code $(p_{i+1}, \ldots, p_n)$ est donc unique et indépendant de p. On le note w_σ. On a alors, pour tout mot d'épingles w qui code une représentation en épingles p de σ, $w \in P(T') \cdot w_\sigma$.
(*b*) Si $|T| = 2$, alors d'après le lemme 9.45, soit $T = \{p_1, p_n\}$, soit $T = \{p_2, p_n\}$. Si $T = \{p_2, p_n\}$, p est déterminée de façon unique. Si $T = \{p_1, p_n\}$, alors on considère $p' = (p_2, p_1, p_3, \ldots, p_n)$, qui est une autre représentation en épingles de σ, pour laquelle $T = \{p'_2, p'_n\}$. Ainsi, p' est uniquement déterminée, et donc p aussi.
Ceci termine de démontrer l'inclusion de $P(\sigma)$ dans les ensembles proposés. L'inclusion réciproque est obtenue par une vérification immédiate. □

On peut remarquer que lorsque la condition $\mathcal{C}$ est satisfaite, $Q_x(\alpha)$ contient un unique mot.

Il reste enfin à considérer le cas où σ se décompose sous la forme $\sigma = \beta[\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(q)}]$ où β est un quasi-épi montant (resp. descendant) et où tous les $\alpha^{(i)}$ sauf deux sont des feuilles, l'un étant

la permutation 12 (resp. 21) qui dilate un point de substitution auxiliaire de β, et l'autre (noté T) dilatant le point de substitution principal correspondant de β.[k] Par symétrie, on se restreindra au cas où β est un quasi-épi montant, et on résumera cette situation par $\sigma =$ 

.

Théorème 9.48. *Soit* $\sigma =$ 

. Alors $P(\sigma) = P(T) \cdot w_{\beta^+}$ *où* w_{β^+} *est un mot unique déterminé par* β^+*, et le marquage de ses points de substitution principal et auxiliaire (correspondant respectivement aux points dilatés par* T *et* 12*) lorsque* $|\beta^+| \leq 5$*.*

Démonstration. On notera dans toute la suite B le fils de β^+ dilaté par 12.

Soit p une représentation en épingles de σ. Si $|T| \geq 3$, alors la conséquence 8.40 et le lemme 8.39 imposent que p lise d'abord T, puis un point de B, puis toutes les autres feuilles sous α, et enfin le deuxième point de B. Il existe donc un indice $i \geq 3$ tel que $T = \{p_1, \ldots, p_i\}$ et $B = \{p_{i+1}, p_n\}$. La remarque 8.41 assure que la séquence de points $(p_{i+1}, \ldots, p_n)$ est uniquement déterminée dans σ. En outre, comme $i \geq 2$, le suffixe de mot d'épingles qui code $(p_{i+1}, \ldots, p_n)$ est aussi déterminé de manière unique, en fonction seulement de β^+, et du marquage des points de substitution principal est auxiliaire de β^+ lorsque $|\beta^+| \leq 5$. On note w_{β^+} ce suffixe de mot d'épingles. Pour tout mot d'épingles w codant p, on a alors $w \in P(T).w_{\beta^+}$.

Si $|T| = 2$, alors d'après le lemme 8.39 et la conséquence 8.40, on a soit $T = \{p_1, p_2\}$ et $B = \{p_3, p_n\}$, soit $B = \{p_1, p_2\}$ et $T = \{p_3, p_n\}$. Dans le premier cas, on conclut comme dans le cas où $|T| \geq 3$ que tout mot d'épingles w codant p vérifie $w \in P(T).w_{\beta^+}$. On montre que le deuxième cas est en fait impossible. En raisonnant par l'absurde, on suppose que $B = \{p_1, p_2\}$ et $T = \{p_3, p_n\}$. La remarque 8.41 assure que T dilate un point de substitution auxiliaire de β^+ et B le point de substitution principal correspondant de β^+. Les points de substitution principal et auxiliaire étant uniquement déterminés dans un quasi-épi de taille au moins 6, on obtient immédiatement une contradiction dans ce cas. Lorsque $|\beta^+| \leq 5$, la remarque 8.41 met aussi en évidence que parmi les deux points de β^+ dilatés, on peut distinguer celui qui est point de substitution principal de celui qui est point de substitution auxiliaire, apportant la même contradiction.

Ceci démontre que $P(\sigma) \subseteq P(T).w_{\beta^+}$. Comme dans les théorèmes précédents, l'inclusion réciproque est une évidence. □

On dispose à présent, pour toute permutation en épingles σ, d'une description de l'ensemble des mots d'épingles associés à σ. Cette description est récursive, en ce sens qu'elle peut faire intervenir l'ensemble des mots d'épingles codant un motif de σ précisément déterminé. Reprenant l'idée d'une amélioration de la complexité de la procédure de [BRV08], on voudrait à présent proposer, pour toute permutation en épingles σ, une construction en temps polynomial d'un automate qui accepte les mots d'épingles stricts codant les permutations en épingles propres π telles que $\sigma \preceq \pi$. Pour ce faire, on revient sur l'ordre $\trianglelefteq$ qui transporte dans les mots d'épingles la relation de motif $\preceq$ sur les permutations.

[k] Ces points de substitution sont uniquement déterminés si $|\beta| \geq 6$. Dans les quais-épis de taille 5, le point de substitution principal est déterminé, mais il y a quatre points auxiliaires possibles, deux conduisant à des quasi-épis montants et deux à des quasi-épis descendants. Pour $|\beta| = 4$, il y a quatre paires de points de substitution possibles, deux donnant un quasi-épi montant et deux un quasi-épi descendant. Voir la figure 8.7 page 179.

9.3 Facteurs de mots d'épingles pour identifier les motifs dans les permutations

Le lemme 9.12 met en évidence le lien entre la relation de motif $\preceq$ sur les permutations, et la relation d'ordre $\trianglelefteq$ sur les mots d'épingles. La relation $u \trianglelefteq w$ sur les mots d'épingles est à peu de choses près une relation de *facteurs par morceaux*, les morceaux étant déterminés par les facteurs forts suivant un chiffre de u : chaque facteur fort suivant un chiffre $u^{(j)}$ de u est égal au facteur $w^{(j)}$ de w dans la définition 9.10, à l'exception de la première lettre de $w^{(j)}$. On définit ici un codage des mots d'épingles, de sorte à transporter la relation $\trianglelefteq$ sur les mots d'épingles en une véritable relation de facteurs par morceaux dans ce codage.

On restreint toute cette étude au cas où w est un mot d'épingles *strict*. Ceci s'explique par le lemme 9.22, le mot w correspondant dans les applications qui nous intéressent (à savoir, la conception d'un algorithme polynomial testant si une classe $\mathcal{C}$ donnée par sa base finie contient un nombre fini de permutations en épingles propres) à un codage d'une permutation en épingles propre.

Définition 9.49. Soit M l'ensemble des mots de longueur supérieure ou égale à 3 sur l'alphabet L, R, U, D tels que R, L est suivi par U, D et vice versa, et soit $\mathcal{SP}_{\geq 2}$ l'ensemble des mots d'épingles stricts de longueur au moins 2. On définit une bijection ϕ de $\mathcal{SP}_{\geq 2}$ dans M par :

Pour $u \in \mathcal{SP}_{\geq 2}$ tel que $u = u'.u''$ avec $|u'| = 2$ on pose $\phi(u) = \varphi(u').u''$ où φ est défini par :

$1R \mapsto RUR$	$2R \mapsto LUR$	$3R \mapsto LDR$	$4R \mapsto RDR$
$1L \mapsto RUL$	$2L \mapsto LUL$	$3L \mapsto LDL$	$4L \mapsto RDL$
$1U \mapsto URU$	$2U \mapsto ULU$	$3U \mapsto DLU$	$4U \mapsto DRU$
$1D \mapsto URD$	$2D \mapsto ULD$	$3D \mapsto DLD$	$4D \mapsto DRD$

Remarque 9.50.
- Pour tout $n \geq 2$, l'application ϕ est une bijection de l'ensemble $\mathcal{SP}_n$ des mots de $\mathcal{SP}$ de taille n dans l'ensemble M_{n+1} des mots de M de taille $n+1$.
- $u_i = \phi(u)_{i+1}$ pour tout $i \geq 2$ et pour tout $u \in \mathcal{SP}_{\geq 2}$,
- On s'aperçoit sur le tableau ci-dessus que les deux premières lettres de $\phi(u)$ suffisent à déterminer la première lettre de u (qui est un chiffre). Donc pour un mot u de $\{LU, LD, RU, RD, UL, UR, DL, DR\}$ on peut par abus de notation définir $\phi^{-1}(u)$.

Conformément à la définition de $\phi^{-1}(u)$ lorsque $|u| = 2$, on peut étendre ϕ aux mots d'épingles stricts de taille 1, mais ϕ n'est alors plus une fonction de $\mathcal{SP}$ dans M. En effet, on associe deux mots à chaque mot d'épingles strict de taille 1 : on pose $\phi(1) = \{UR, RU\}, \phi(2) = \{UL, LU\}, \phi(3) = \{DL, LD\}$ et $\phi(4) = \{DR, RD\}$.

La bijection ϕ consiste à remplacer le chiffre d'un mot d'épingles strict par deux lettres. Dans ce codage, les lettres de $\phi(u)$ suffisent à déterminer dans quel quadrant se situe chaque point d'une représentation en épingles correspondant au mot d'épingles strict u.

Lemme 9.51. *Soit w un mot d'épingles strict et p une représentation en épingles correspondant à w. Alors pour tout $i \geq 2$, w_{i-1} et w_i déterminent le quadrant dans lequel p_i se situe par rapport à $\{p_0, \ldots, p_{i-2}\}$:*
- *si $i \geq 3$, p_i est situé dans le quadrant $\phi^{-1}(w_{i-1}w_i)$.*
- *si $i = 2$, p_i est situé dans le quadrant $\phi^{-1}(BC)$ où $\phi(w_1w_2) = ABC$.*

Démonstration. La preuve se fait par examen de toutes les paires de lettres consécutives possibles. On en donne deux exemples.

Si $i \geq 3$, alors w_{i-1} et w_i sont des directions. Par exemple si $w_{i-1} = L$ et $w_i = U$, alors p_i est situé dans le quadrant 2 et $\phi^{-1}(LU) = 2$.

Si $i = 2$, w_{i-1} est un chiffre et w_i est une direction. Par exemple si $w_{i-1} = 1$ et $w_i = L$, alors p_i est situé dans le quadrant 2, $\phi(1L) = RUL$ et $\phi^{-1}(UL) = 2$. □

Par définition, les facteurs forts suivant un chiffre de tout mot d'épingles u sont des mots d'épingles stricts. On étudie donc d'abord, pour un mot d'épingles strict u, comment la relation $u \trianglelefteq w$ se transporte sur $\phi(u)$ et $\phi(w)$.

On envisage indépendamment le cas où $|u| = 1$.

Lemme 9.52. *On pose $u = 1$ (resp $2, 3, 4$). Pour tout mot d'épingles strict w, $u \trianglelefteq w$ si et seulement si $\phi(w)$ possède un facteur dans $\phi(u) = \{UR, RU\}$ (resp. $\{UL, LU\}$, $\{DL, LD\}$, $\{DR, RD\}$).*

Démonstration. On démontre le lemme 9.52 pour $u = 1$, les autres cas étant traités de la même manière. On se donne un mot d'épingles strict w.

On suppose que $1 \trianglelefteq w$. On considère la décomposition de w en $w = v^{(1)}w^{(1)}v^{(2)}$ donnée par la définition 9.10, et on note $i = |v^{(1)}|$.

Si $i = 0$, alors la première lettre de w est 1, et $\phi(w)$ commence par UR ou RU.

Si $i \geq 2$, alors w_i et w_{i+1} sont deux directions, et le point correspondant à w_{i+1} dans la permutation admettant w pour mot d'épingles se situe dans le quadrant $\phi^{-1}(w_iw_{i+1})$. D'après la définition 9.10 le quadrant correspondant à w_{i+1}, qui est la première lettre de $w^{(1)}$, est le même que celui correspondant à u_1, c'est-à-dire que c'est ici le quadrant 1. Ainsi, $\phi^{-1}(w_iw_{i+1}) = 1$, et donc $w_iw_{i+1} \in \{UR, RU\}$. Enfin, comme $i \geq 2$, w_iw_{i+1} est un facteur de $\phi(w)$, concluant la preuve dans ce cas.

Enfin si $i = 1$, alors la première lettre de $w^{(1)}$ est w_2, qui doit comme avant correspondre à un point situé dans le quadrant 1. En posant $\phi(w_1w_2) = ABC$, le lemme 9.51 assure que $\phi^{-1}(BC) = 1$, c'est-à-dire que $BC \in \{UR, RU\}$. Comme BC est par définition un facteur de $\phi(w)$ (correspondant à ses deuxième et troisième lettres), on a le résultat annoncé.

Réciproquement, on suppose maintenant que $\phi(w)$ possède un facteur UR ou RU. Sans perte de généralité, on suppose qu'on est dans le premier cas. Il existe donc deux mots v et v' tels que $\phi(w) = vURv'$.

Si $|v| = 0$, alors la définition 9.49 indique que w commence par 1, et donc $1 \trianglelefteq w$.

Si $|v| \geq 2$, alors $\phi(w) = vURv'$, et pour la décomposition $w = v^{(1)}w^{(1)}v^{(2)}$ avec $v^{(1)} = \phi^{-1}(v)U, w^{(1)} = R$ et $v^{(2)} = v'$, on vérifie que $1 \trianglelefteq w$.

Si $|v| = 1$, alors comme $\phi(w) \in M$, on a $\phi(w) = RURv'$ ou $\phi(w) = LURv'$, c'est-à-dire que $w = 1Rv'$ ou $w = 2Rv'$. Dans le premier cas, on note $v^{(1)} = 1, w^{(1)} = R$ et $v^{(2)} = v'$, et dans le deuxième cas $v^{(1)} = 2, w^{(1)} = R$ et $v^{(2)} = v'$. Chacune de ces décompositions fait apparaître que $1 \trianglelefteq w$. □

Théorème 9.53. *Pour tous mots d'épingles stricts u et w, avec $|u| \geq 2$, $u \trianglelefteq w$ si et seulement si $\phi(u)$ est un facteur de $\phi(w)$.*

Démonstration. Si $u \trianglelefteq w$, comme u est un mot d'épingles strict, la décomposition de u en facteurs forts est $u = u^{(1)}$, donc w peut être coupé en une suite de facteurs $w = v^{(1)}w^{(1)}v^{(2)}$ comme dans la définition 9.10.

Si $v^{(1)}$ est vide, alors $w^{(1)}$ commence par un chiffre, $w^{(1)} = u^{(1)}$ et u est un préfixe de w donc $\phi(u)$ est un préfixe de $\phi(w)$.

Sinon $w^{(1)}$ commence par une direction (car w est strict) donc la première lettre de $w^{(1)}$ correspond à un point situé dans le quadrant spécifié par u_1 (la première lettre de $u^{(1)}$), et toutes les autres lettres (qui sont des directions) de $u^{(1)}$ et $w^{(1)}$ sont les mêmes : $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$, en notant $i = |v^{(1)}|$.

La seule différence entre u et $\phi(u)$ étant que u_1 a été remplacé par deux lettres, on peut écrire $\phi(u) = \phi(u)_1\phi(u)_2u_2 \dots u_{|u|}$.

Si $i \geq 2$ alors d'après le lemme 9.51, le point correspondant à w_{i+1} est situé dans le quadrant $\phi^{-1}(w_i w_{i+1})$, et donc $u_1 = \phi^{-1}(w_i w_{i+1})$. On a alors $\phi(u) = w_i w_{i+1} \dots w_{i+|u|}$.

Si $i = 1$ alors d'après le lemme 9.51, $u_1 = \phi^{-1}(BC)$ où $\phi(w_1 w_2) = ABC$ donc $\phi(u) = BCu_2 \dots u_{|u|}$ et $\phi(w) = ABCw_3 \dots w_{|w|}$.

Dans tout ces cas, on a bien démontré que $\phi(u)$ est un facteur de $\phi(w)$.

Inversement si $\phi(u)$ est un facteur de $\phi(w)$, alors $\phi(w) = v.\phi(u).v'$. Si v est vide alors $\phi(u)$ est un préfixe de $\phi(w)$ donc u est un préfixe de w d'où $u \trianglelefteq w$.

Si $|v| = i \geq 2$ alors $w = \phi^{-1}(v).\phi(u).v'$. Donc $\phi(u) = w_i \dots w_{i+|\phi(u)|-1} = w_i \dots w_{i+|u|}$ d'où $u_2 \dots u_{|u|} = w_{i+2} \dots w_{i+|u|}$ et $u_1 = \phi^{-1}(w_i w_{i+1})$. Donc u_1 est le quadrant dans lequel se situe w_{i+1} et $u \trianglelefteq w$.

Si $|v| = 1$ alors $\phi(w_1 w_2) = v\,\phi(u)_1 \phi(u)_2$ donc d'après le lemme 9.51, w_2 est situé dans le quadrant $\phi^{-1}(\phi(u)_1 \phi(u)_2) = u_1$. Or $w_3 \dots w_{|u|+1} = \phi(w)_4 \dots \phi(w)_{|u|+2} = \phi(u)_3 \dots \phi(u)_{|u|+1} = u_2 \dots u_{|u|}$, donc $u \trianglelefteq w$. □

Lorsque le mot d'épingles u n'est pas strict, le lemme 9.52 et le théorème 9.53 peuvent être étendus avec la notion suivante, qui formalise l'idée de facteur par morceaux expliquée au début de ce paragraphe.

Définition 9.54. Pour tout mot d'épingles u, dont la décomposition en facteurs forts suivant un chiffre est $u = u^{(1)}u^{(2)} \dots u^{(j)}$, on définit $\mathcal{L}(u) = \Sigma^* \cdot \phi(u^{(1)}) \cdot \Sigma^* \cdot \phi(u^{(2)}) \cdot \Sigma^* \dots \Sigma^* \cdot \phi(u^{(j)}) \cdot \Sigma^*$, où $\Sigma = \{L, R, U, D\}$.

Théorème 9.55. *Soit σ une permutation en épingles propre, et w un mot d'épingles strict de σ. Soit β une permutation. Alors $\beta \preceq \sigma$ si et seulement s'il existe un mot d'épingles u de β tel que $\phi(w) \in \mathcal{L}(u)$.*

Démonstration. On suppose que $\beta \preceq \sigma$. Alors β est une permutation en épingles, et d'après la conséquence 9.13, il existe un mot d'épingles u codant β tel que $u \trianglelefteq w$. On décompose u en facteurs forts suivant un chiffre, sous la forme $u = u^{(1)}u^{(2)} \dots u^{(j)}$. Alors w se décompose en $w = v^{(1)}w^{(1)}v^{(2)}w^{(2)} \dots v^{(j)}w^{(j)}v^{(j+1)}$ de sorte que pour tout $i \in [1..j]$, $u^{(i)} \trianglelefteq v^{(i)}w^{(i)}$. Il faut remarquer que cette notation induit une extension (immédiate) de $\trianglelefteq$ aux mots de M, c'est-à-dire ne contenant pas de chiffres. En étendant aussi ϕ aux mots de M (par l'identité de M), le théorème 9.53 et le lemme 9.52 assurent alors que pour tout $i \in [1..j]$, $\phi(v^{(i)}w^{(i)})$ contient un facteur égal à (ou appartenant à) $\phi(u^{(i)})$. Enfin, comme w est un mot d'épingles strict, $\phi(w) = \phi(v^{(1)}w^{(1)})\phi(v^{(2)}w^{(2)}) \dots \phi(v^{(j)}w^{(j)})v^{(j+1)}$, et ainsi les facteurs $\phi(u^{(1)}) \dots \phi(u^{(j)})$ apparaissent dans $\phi(w)$, dans cet ordre de gauche à droite, et en étant disjoints. En d'autres termes, $\phi(w) \in \mathcal{L}(u)$.

Réciproquement, s'il existe un mot d'épingles u codant β tel que $\phi(w) \in \mathcal{L}(u)$, on peut décomposer $\phi(w)$ en $v^{(1)}\phi(u^{(1)})v^{(2)}\phi(u^{(2)}) \dots v^{(j)}\ \phi(u^{(j)})v^{(j+1)}$. Là encore, ce sont le théorème 9.53 et le lemme 9.52 (sens réciproque) dans leur version étendue aux mots de M, associés à la conséquence 9.13, qui permettent de conclure que $\beta \preceq \sigma$. □

Pour décider si une classe $\mathcal{C}$ contient un nombre fini de permutations en épingles propres (et donc décider si elle contient un nombre fini de permutations simples) on utilisera le théorème 9.55 pour les permutations β qui sont des motifs exclus caractérisant la classe $\mathcal{C}$.

Dans le cas particulier des classes $\mathcal{C}$ fermées par produit de substitution, on peut décrire un algorithme décidant si $\mathcal{C}$ contient un nombre fini de permutations simples, qui soit plus efficace que dans le cadre général. Il faut pour cela analyser plus finement qu'au théorème 9.55, pour tout mot d'épingles u codant une permutation β de la base de $\mathcal{C}$, comment s'exprime la relation $\beta \preceq \sigma$ en fonction des mots d'épingles stricts de σ, pour les permutations en épingles propres σ.

On rappelle que les classes fermées par produit de substitution sont caractérisées au théorème 2.64 comme celles dont tous les motifs exclus sont des permutations simples, et que les mots d'épingles qui codent une permutation simple sont tous stricts ou quasi-stricts (remarque 9.8). Le

théorème 9.53 s'intéresse au cas où u est un mot d'épingles strict codant β. On traite maintenant le cas d'un mot d'épingles u quasi-strict.

Lemme 9.56. *Soit u un mot d'épingles quasi-strict correspondant à une permutation β et w un mot d'épingles strict correspondant à une permutation σ. Si $u \trianglelefteq w$ alors $\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$ qui commence à une position $j \geq 3$.*

Démonstration. La décomposition de u en facteurs forts suivant un chiffre est $u = u^{(1)}u^{(2)}$ avec $|u^{(1)}| = 1$. Puisque $u \trianglelefteq w$, w peut être coupé en une suite de facteurs $w = v^{(1)}w^{(1)}v^{(2)}w^{(2)}v^{(3)}$. De plus $|w^{(1)}| = |u^{(1)}| = 1$. Comme w est un mot d'épingles strict, $w^{(2)}$ ne contient donc pas de chiffre. Ainsi, $v^{(2)}$ est non vide, la première lettre w_i de $w^{(2)}$ correspond à un point situé dans le quadrant spécifié par la première lettre u_2 de $u^{(2)}$, et toutes les autres lettres de $u^{(2)}$ et $w^{(2)}$ sont les mêmes. Donc $u_2 = \phi^{-1}(w_{i-1}w_i)$ par le lemme 9.51 et $w = v^{(1)}w^{(1)}v\phi(u^{(2)})v^{(3)}$ où v est le préfixe de $v^{(2)}$ de longueur $|v^{(2)}| - 1$. Alors $\phi(u^{(2)})$ est un facteur de w qui ne contient pas de chiffre donc $\phi(u^{(2)})$ est un facteur de $\phi(w)$. En outre, comme $|w^{(1)}| = 1$, $\phi(u^{(2)})$ a une occurrence dans w qui commence à une position $j' \geq 2$, donc $\phi(u^{(2)})$ a une occurrence dans $\phi(w)$ qui commence à une position $j \geq 3$. □

Lemme 9.57. *Soit u un mot d'épingles quasi-strict correspondant à une permutation β et w un mot d'épingles strict correspondant à une permutation σ.*

Si $|u| \geq 3$ et si $\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$ qui commence à une position $j \geq 3$ alors β est un motif de σ.

Si $|u| = 2$ et si $\phi(w)$ possède un facteur appartenant à $\phi(u_2)$ qui commence à une position $j \geq 3$ alors β est un motif de σ.

Démonstration. On note $u = u^{(1)}u^{(2)}$ la décomposition en facteurs forts suivant un chiffre de u. Comme u est un mot d'épingles quasi-strict, on a $u^{(1)} = u_1$ et $u^{(2)} = u_2 \dots u_k$, pour $k = |u|$. On suppose que $k \geq 3$. D'après la définition 9.49, on a $\phi(u^{(2)}) = abu_3 \dots u_k$. D'un autre côté, w étant un mot d'épingles strict, en notant $n = |w|$, on a d'après la définition 9.49 $\phi(w) = cdw_2 \dots w_n$. Comme $\phi(u^{(2)})$ a une occurrence dans $\phi(w)$ à une position $j \geq 3$, cela signifie que $abu_3 \dots u_k$ a une occurrence dans $w_2 \dots w_n$.

De même, si $k = 2$, il existe $ab \in \phi(u_2)$ tel que ab a une occurrence dans $w_2 \dots w_n$.

Dans les deux cas, il existe un indice $i \geq 2$ tel que $a = w_i$, $b = w_{i+1}$ et pour tout $\ell \in [3..k]$, $u_\ell = w_{i+\ell-1}$.

On note $p = (p_1, p_2, \dots, p_n)$ la représentation en épingles de σ dont w est un codage. Alors $(p_{i+1}, p_{i+2}, \dots, p_{i+k-1})$ est une représentation en épingles dont $u^{(2)}$ est un codage. En outre, le lemme 9.51 assure que p_{i+1} se trouve dans le quadrant $\phi^{-1}(w_iw_{i+1}) = \phi^{-1}(ab) = u_2$ par rapport à $\{p_0, \dots, p_{i-1}\}$, et donc en particulier par rapport à p_1 (on rappelle que $i \geq 2$). On considère alors la représentation en épingles $q = (p_1, p_{i+1}, p_{i+2}, \dots, p_{i+k-1})$, qui correspond évidemment à un motif de σ. Les arguments précédents utilisant le lemme 9.51 permettent aussi de déduire immédiatement que $u = u_1u^{(2)}$ est un mot d'épingles codant q, d'où on conclut que q est une représentation en épingles de β, et donc que β est un motif de σ. □

On associe à chaque permutation simple β l'ensemble $P(\beta)$ de ses mots d'épingles (qui est vide si β n'est pas une permutation en épingles). D'après le lemme 8.47 et la remarque 9.2, cet ensemble contient au plus 64 éléments. On pose $E(\beta) = \{\phi(u) \,|\, u$ est un mot d'épingles strict correspondant à $\beta\} \cup \{v \in M \,|$ il existe un mot d'épingles quasi-strict u correspondant à β et $x \in \{LU, LD, RU, RD\} \cup \{UL, UR, DL, DR\}$ tel que $v = x\,\phi(u_2 \dots u_{|u|})\}$. Pour le second ensemble, la première lettre de $\phi(u_2 \dots u_{|u|})$ détermine l'ensemble auquel appartient x.

On voit que $|E(\beta)| \leq 320$.

Théorème 9.58. *Soit β une permutation simple et w un mot d'épingles strict correspondant à une permutation σ. Alors $\beta \not\leq \sigma$ si et seulement si $\phi(w)$ évite l'ensemble fini de facteurs $E(\beta)$.*

On peut remarquer que dans le théorème 9.58, il suffit de considérer un seul mot d'épingles strict correspondant à σ : il n'est pas nécessaire de les connaître tous.

Démonstration. On démontre la contraposée de l'équivalence annoncée.

Si $\beta \preceq \sigma$, alors par le lemme 9.12, il existe un mot d'épingles u correspondant à β tel que $u \trianglelefteq w$. D'après la remarque 9.8, u est un mot d'épingles strict ou quasi-strict. Si u est un mot d'épingles strict alors, par le théorème 9.53, $\phi(u)$ est un facteur de $\phi(w)$ et donc $\phi(w)$ a un facteur dans $E(\beta)$. Si u est un mot d'épingles quasi-strict alors par le lemme 9.56, $\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$ qui commence à une position $j \geq 3$. Soit x les deux lettres qui précèdent $\phi(u_2 \dots u_{|u|})$ dans $\phi(w)$. Comme $\phi(w) \in M$, $x\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$ qui appartient à $E(\beta)$.

Inversement on suppose que $\phi(w)$ a un facteur v dans $E(\beta)$. Si $v \in \{\phi(u) \,|u$ est un mot d'épingles strict correspondant à $\beta\}$ alors par le théorème 9.53, il existe u un mot d'épingles correspondant à β tel que $u \trianglelefteq w$ donc par le lemme 9.12, $\beta \preceq \sigma$. Sinon il existe u un mot d'épingles quasi-strict correspondant à β et $x \in \{LU, LD, RU, RD, UL, UR, DL, DR\}$ tel que $v = x\,\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$. Donc $\phi(u_2 \dots u_{|u|})$ est un facteur de $\phi(w)$ qui commence à une position $j \geq 3$ et par le lemme 9.57, $\beta \preceq \sigma$. □

9.4 Algorithme polynomial testant si une classe contient un nombre fini de simples

9.4.1 Les étapes de l'algorithme

On considère une classe de permutations $\mathcal{C}$ donnée par sa base finie $\mathcal{B} = \{\beta_1, \dots, \beta_k\}$. On note n_i la taille de chaque motif β_i et $n = \max n_i$. On dispose à présent des outils nécessaires pour donner l'algorithme en $\mathcal{O}(n^{3k})$ qui teste si $\mathcal{C}$ contient ou non un nombre fini de permutations simples.

On a vu au paragraphe 9.1.2 (lemme 9.17) que $\mathcal{C}$ contient un nombre infini des permutations simples si et seulement si elle contient

- un nombre infini de permutations parallèles,
- ou un nombre infini de permutations simples en chevron de type 1,
- ou un nombre infini de permutations simples en chevron de type 2,
- ou un nombre infini de permutations en épingles propres.

Par la conséquence 9.21, on peut tester les trois premiers points en temps $\mathcal{O}(k \cdot n \log n)$. Il reste donc à décrire un algorithme polynomial testant si $\mathcal{C}$ contient un nombre infini de permutations en épingles propres.

L'idée est comme avant d'utiliser la proposition 9.23, qui énonce que $\mathcal{C}$ contient un nombre infini de permutations en épingles propres si et seulement si $\mathcal{SP} \setminus \mathcal{E}$ est infini, $\mathcal{SP}$ désignant l'ensemble de mots d'épingles stricts, et $\mathcal{E}$ celui des mots d'épingles stricts w tels que $w \trianglerighteq u$ pour un mot d'épingles u codant un motif $\beta \in \mathcal{B}$.

Comme on l'a fait remarquer juste avant d'énoncer la proposition 9.23, seuls les motifs de $\mathcal{B}$ qui sont des permutations en épingles sont utiles dans la définition de $\mathcal{E}$. On commence donc par déterminer, pour chaque motif $\beta \in \mathcal{B}$, si β est ou non une permutation en épingles. Pour ce faire, on calcule l'arbre de décomposition de β, ce qui peut être fait en temps linéaire d'après le théorème 2.45. Puis on teste si l'arbre est de la forme indiquée par le théorème 8.25 caractérisant les arbres de décomposition des permutations en épingles. Le seul point qui demande une précision dans cette vérification consiste à avoir un algorithme pour décider si une permutation simple est ou non en épingles. On a donné à la remarque 9.25 un algorithme en $\mathcal{O}(|\alpha|^2)$ testant si une permutation simple α est ou non en épingles.

Au total, on dispose donc d'un algorithme en $\mathcal{O}(k \cdot n^2)$ déterminant le sous-ensemble $\mathcal{B}'$ de $\mathcal{B}$ composé des motifs β_i qui sont représentables en épingles.

L'étape suivante est de construire, pour chaque $\beta \in \mathcal{B}'$, un automate $\mathcal{A}_\beta$ qui accepte le langage $\mathcal{L}_\beta$ des mots d'épingles stricts w tels qu'il existe un mot d'épingles u codant β et vérifiant $w \trianglerighteq u$. On

souhaite bien sûr que $\mathcal{A}_\beta$ soit construit en temps polynomial (et donc de taille polynomiale), mais on veut aussi que $\mathcal{A}_\beta$ soit un automate *déterministe*. Cela permettra d'obtenir, par une construction produit, un automate $\mathcal{A}_{\mathcal{B}'}$, déterministe lui aussi, reconnaissant le langage union des $\mathcal{L}_\beta$ pour $\beta \in \mathcal{B}'$. Le coût de cette construction s'exprime comme le produit des tailles des automates $\mathcal{A}_\beta$.

L'intérêt d'avoir un automate $\mathcal{A}_{\mathcal{B}'}$ *déterministe* est de pouvoir calculer *en temps polynomial* un automate qui reconnait le langage complémentaire : sans la contrainte du déterminisme, la complémentation est une opération exponentielle. L'intersection avec le langage des mots d'épingles stricts est ensuite effectuée par produit avec l'automate (déterministe à quatre états) qui reconnait $\mathcal{SP}$. Cet automate est donnée sur la figure 9.13, figure de gauche.

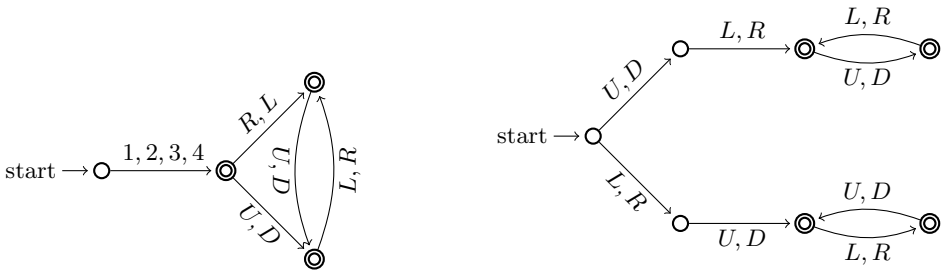


FIG. 9.13 – À gauche, un automate déterministe acceptant le langage des mots d'épingles stricts. À droite, un automate déterministe acceptant le langage des mots de longueur au moins 2 sans facteurs dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$. Ce langage correspond à l'ensemble des miroirs des $\phi(w)$ associés aux mots d'épingles stricts w.

On obtient à l'issue de ce processus un automate $\mathcal{A}$, dont la taille est du même ordre que celle de $\mathcal{A}_{\mathcal{B}'}$, qui accepte le langage $\mathcal{SP} \setminus \mathcal{E}$. Il suffit alors de tester si le langage reconnu par $\mathcal{A}$ est infini ou non. Une méthode classique pour le faire consiste à chercher dans $\mathcal{A}$ un cycle qui soit accessible (atteignable depuis l'état initial) et co-accessible (depuis lequel on peut rejoindre un état final), en effectuant un parcours en profondeur de $\mathcal{A}$. Cette dernière étape est linéaire en le nombre d'arêtes de $\mathcal{A}$. L'existence d'un tel cycle est une condition nécessaire et suffisante pour que le langage reconnu par $\mathcal{A}$ soit infini.

On apporte une légère modification à ce schéma général. Les automates $\mathcal{A}_\beta$ que l'on construit ne reconnaissent pas les mots d'épingles stricts w tels qu'il existe un mot d'épingles u codant β et vérifiant $w \trianglerighteq u$. Ils reconnaissent à la place les miroirs $\overleftarrow{\phi(w)}$ des mots $\phi(w)$ associés aux mots d'épingles stricts w tels qu'il existe un mot d'épingles u codant β et vérifiant $\phi(w) \in \mathcal{L}(u)$.

Le choix de travailler sur $\phi(w)$ plutôt que sur w lui-même est motivé par le théorème 9.55, qui exprime comment la relation de motif $\preceq$ sur les permutations est transformée en une relation de facteurs par morceaux sur $\phi(w)$. L'avantage de la condition $\phi(w) \in \mathcal{L}(u)$ par rapport à $w \trianglerighteq u$ (qui est d'après le lemme 9.12 une autre traduction de $\preceq$) est qu'elle est plus facilement testable algorithmiquement.

La deuxième variante consiste à travailler sur des miroirs de mots plutôt que sur les mots eux-mêmes. La raison de ce changement est purement technique, pour permettre d'avoir des automates déterministes. On peut se convaincre de son bien-fondé par la réflexion informelle suivante : le début des mots d'épingles codant une permutation β peut être très variable (comme en témoignent les théorèmes 9.44, 9.47 et 9.48), mais leur fin est généralement très contrainte. En faisant une lecture de droite à gauche, on repousse jusqu'au dernier moment le choix du mot d'épingles codant β que l'on est en train de lire.

Le reste du processus pour déterminer si $\mathcal{C}$ contient un nombre fini de permutations en épingles propres est identique, à ceci près qu'il ne faut plus complémenter dans le langage $\mathcal{SP}$ des mots d'épingles stricts, mais dans le langage des mots de taille au moins 2 sur l'alphabet $\{L, R, U, D\}$ ne

contenant aucun facteur de $\{UU, UD, DU, DD, RR, RL, LR, LL\}$. Ce langage correspond à l'ensemble des $\overleftarrow{\phi(w)}$ associés aux mots d'épingles stricts w. Un automate reconnaissant ce langage est donnée en figure 9.13, figure de droite.

Les paragraphes qui suivent expliquent comment construire, pour toute permutation σ de taille n, et en temps polynomial en n, un automate déterministe et possédant un seul état final (noté aussi $\mathcal{A}_\sigma$), qui accepte le langage $\left\{\overleftarrow{\phi(u^{(1)})\phi(u^{(2)})\dots\phi(u^{(j)})} \mid u \text{ est un mot d'épingles codant } \sigma \text{ dont la décomposition en facteurs forts suivant un chiffre est } u = u^{(1)}u^{(2)}\dots u^{(j)}\right\}$. On verra dans un deuxième temps comment ajouter des transitions pour obtenir un automate qui reconnaisse $\cup_{u \text{ codant } \sigma}\overleftarrow{\mathcal{L}(u)}$.

Par abus de notations, on écrira $\phi(v) = \phi(v^{(1)})\phi(v^{(2)})\dots\phi(v^{(j)})$ lorsque la décomposition en facteurs forts suivant un chiffre de v est $v^{(1)}v^{(2)}\dots v^{(j)}$.[l]

Les constructions des automates sont données en suivant la même organisation qu'au paragraphe 9.2 où on a décrit les ensembles de mots d'épingles associés à des permutations en épingles.

9.4.2 Construction de l'automate pour les cas de base

Permutation de taille 1

Les mots $\overleftarrow{\phi(w)}$ qui correspondent à $w \in \{1, 2, 3, 4\}$, c'est-à-dire à $\sigma = 1$, sont ceux acceptés par l'automate de la figure 9.14.

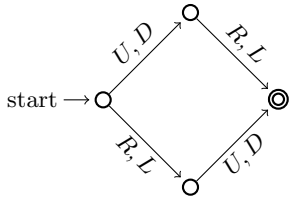


FIG. 9.14 – Automate reconnaissant le langage des miroirs des $\phi(w)$ pour les mots d'épingles w codant la permutation 1.

Permutations simples

On considère une permutation en épingles simple σ. La proposition 9.24 assure que σ possède au plus 64 mots d'épingles, que l'on sait calculer en temps $\mathcal{O}(n^2)$. Certains de ces mots d'épingles sont stricts, mais d'autres sont quasi-stricts (voir la proposition 9.7). On considère l'ensemble E des $\overleftarrow{\phi(w)}$, pour les mots d'épingles w codant σ, ϕ étant appliquée sur chaque facteur fort suivant un chiffre lorsque w n'est pas strict. On peut remarquer qu'alors l'ensemble $\overleftarrow{E(\sigma)}$ défini pour le théorème 9.58 correspond à l'intersection de l'ensemble des $\overleftarrow{\phi(w)}$ avec le langage des mots sans facteurs dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$.

Étant donné cet ensemble E d'au plus 64 mots, tous de taille n, on peut calculer en temps $\mathcal{O}(n)$ un automate déterministe qui reconnait E. On utilise pour cela la méthode de A. Aho et M. Corasick dans [AC75], pour construire seulement le squelette de leur automate, qui correspond au *trie* associé à l'ensemble de mots considéré : chaque état correspond à un préfixe d'un ou de plusieurs mots dans l'ensemble E, et à tout mot de E correspond un unique chemin qui parcourt l'ensemble de ses préfixes, par longueur croissante. La figure 9.15 illustre cette construction sur un

[l] On peut remarquer que $\phi(v)$ peut être un ensemble de mots (dès que deux chiffres apparaissent consécutivement dans v), et que les mots de $\phi(v)$ peuvent contenir des facteurs dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$.

exemple. Comme on veut construire des automates ayant un unique état final, la seule modification consiste à fusionner en un état final unique tous les états finaux de l'automate. Comme aucune transition ne sort d'aucun état final dans l'automate du *trie* construit, cette modification ne change pas le langage accepté par l'automate.

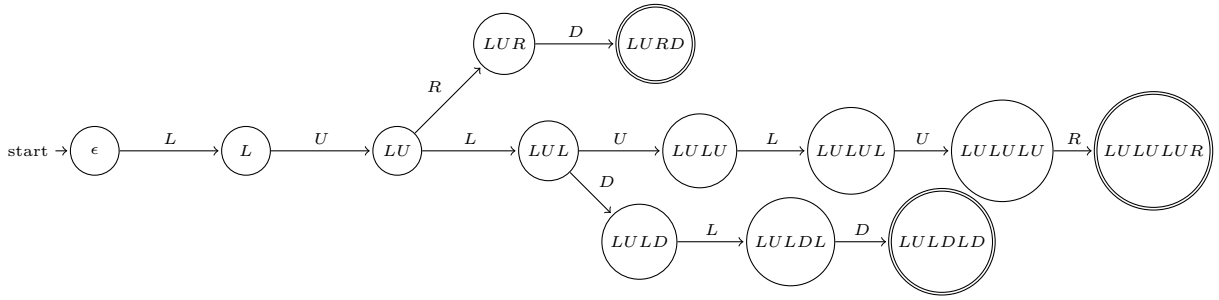


FIG. 9.15 – Exemple de construction du squelette de l'automate de A. Aho et M. Corasick pour l'ensemble de mots $\{LULULUR, LULDLD, LURD\}$.

Permutations de racine linéaire

On traite seulement le cas des permutations de racine $\oplus$, le cas de la racine $\ominus$ étant symétrique. On considère donc une permutation en épingles σ qui se décompose sous la forme $\sigma = \oplus[\xi_1, \xi_2, \ldots, \xi_q]$ où tous les ξ_i sont des épis montants.

Pour chaque $j \in [1..q]$, on peut construire des automates déterministes $\mathcal{A}^{(1)}(\xi_j)$ et $\mathcal{A}^{(3)}(\xi_j)$ qui acceptent l'ensemble des mots $\overleftarrow{\phi(v)}$ respectivement pour $v \in f^{(1)}(\xi_j)$ et pour $v \in f^{(3)}(\xi_j)$. De même, pour chaque $j \in [1..(q-1)]$, l'ensemble de mots d'épingles $f^{double}(\xi_j, \xi_{j+1})$ étant complètement déterminé par le lemme 9.34, on peut construire un automate déterministe $\mathcal{A}^{(double)}(\xi_j, \xi_{j+1})$ qui accepte l'ensemble des mots $\overleftarrow{\phi(v)}$ pour $v \in f^{double}(\xi_j, \xi_{j+1})$. Cet automate $\mathcal{A}^{(double)}(\xi_j, \xi_{j+1})$ est de la forme représentée en figure 9.16, sur lequel des états et des transitions supplémentaires viennent se greffer lorsque $f^{mix}(\xi_j, \xi_{j+1})$ est non vide. Sur cette figure, les automates $\mathcal{A}^{(int)}(\xi_i)$ sont définis comme ceux qui acceptent l'ensemble des mots $\overleftarrow{\phi(v)}$ pour $v \in f^{int}(\xi_i)$.

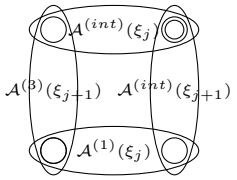


FIG. 9.16 – La forme des automates $\mathcal{A}^{(double)}(\xi_j, \xi_{j+1})$.

Pour construire ces automates, on peut comme avant utiliser la méthode de A. Aho et M. Corasick dans [AC75], pour construire le squelette de leur automate, qui correspond au *trie* associé à l'ensemble de mots considéré. Ici encore, on fusionne en un état final unique tous les états finaux de l'automate. Le temps nécessaire à la construction de chaque automate est alors $\mathcal{O}(|\xi_i|)$ ou $\mathcal{O}(|\xi_i| + |\xi_{i+1}|)$, et chacun des automates obtenus a une taille en $\mathcal{O}(|\xi_i|)$ ou $\mathcal{O}(|\xi_i| + |\xi_{i+1}|)$. La construction de l'ensemble de ces automates demande donc un temps $\mathcal{O}(\sum |\xi_i|) = \mathcal{O}(n)$.

Il est alors évident avec la proposition 9.36 que l'automate de la figure 9.17 reconnait l'ensemble des mots $\overleftarrow{\phi(u)}$ pour les mots d'épingles u qui codent σ. Sur cette figure, pour deux automates $\mathcal{A}^{(1)}(\xi_j)$ et $\mathcal{A}^{(1)}(\xi_{j+1})$, l'état initial de $\mathcal{A}^{(1)}(\xi_{j+1})$ est fusionné avec l'état final de $\mathcal{A}^{(1)}(\xi_j)$, et il en

va de même avec les automates $\mathcal{A}^{(3)}(\xi_{j+1})$ et $\mathcal{A}^{(3)}(\xi_j)$. En outre, les états finaux des automates $\mathcal{A}^{(double)}(\xi_j, \xi_{j+1})$ sont tous fusionnés en un unique état final.

Il est important de remarquer que toutes ces fusions d'états finaux sont possibles car, dans le squelette de l'automate obtenu par la construction de A. Aho et M. Corasick, il n'y a aucune transition qui sorte d'un état final.

La complexité en temps et en espace de cette construction est alors en $\mathcal{O}(n^2)$. À partir des automates calculés en temps $\mathcal{O}(n)$, l'automate de la figure 9.17 est obtenu en mélangeant $\mathcal{O}(q)$ automates (correspondant aux permutations $(\xi_i)_{1\leq i\leq q}$), répliqués chacun $\mathcal{O}(q)$ fois, et dont les tailles sont, pour chaque $i \in [1..q]$, en $\mathcal{O}(|\xi_i|)$ ou en $\mathcal{O}(|\xi_i| + |\xi_{i+1}|)$. La taille de cet automate est $\mathcal{O}(\sum_{i=1}^{q} q \cdot |\xi_i|) = \mathcal{O}(n \cdot \sum_{i=1}^{q} |\xi_i|) = \mathcal{O}(n^2)$.

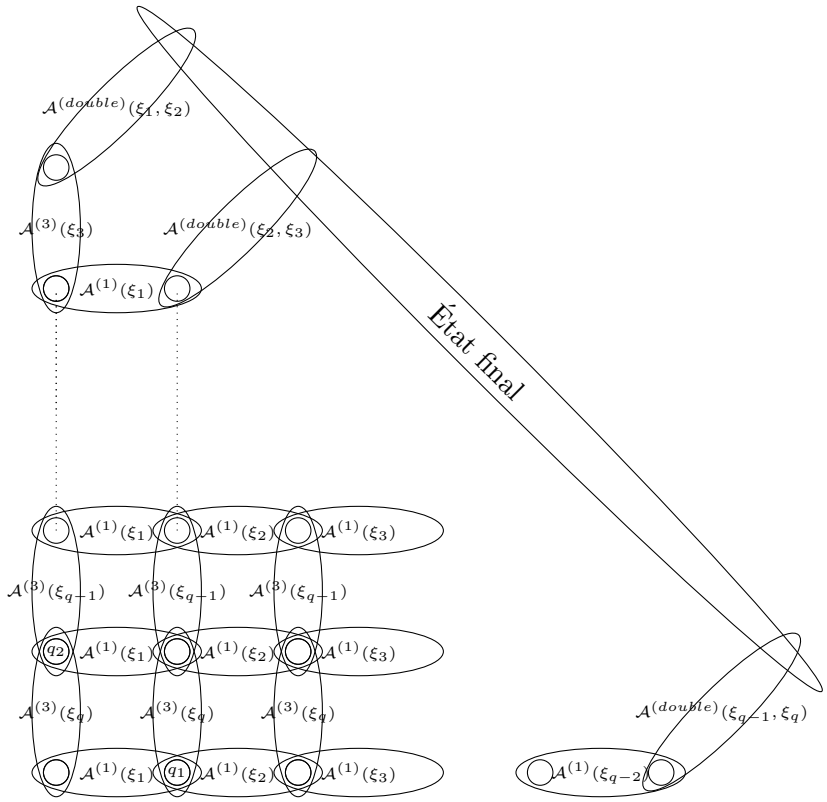
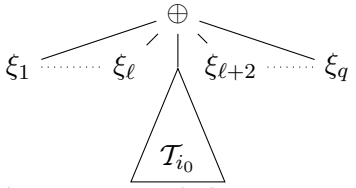


FIG. 9.17 – L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine linéaire dont tous les fils sont des épis. L'état initial est celui en bas à gauche.

9.4.3 Construction de l'automate pour les cas récursifs de racine linéaire

Ici encore, on restreint cette étude au cas où la racine est un nœud linéaire d'étiquette $\oplus$. Tous les fils de cette racine sont donc des épis montants, sauf un, que l'on note $\mathcal{T}_{i_0}$. L'arbre de décomposition de σ est alors de la forme



On note $\mathcal{A}(\mathcal{T}_{i_0})$ l'automate associé récursivement à la permutation dont $\mathcal{T}_{i_0}$ est l'arbre de décomposition.

On suppose d'abord que σ ne satisfait aucune des conditions du type iHj de la figure 9.10.

Alors le théorème 9.44 assure que le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ sont ceux reconnus par l'automate obtenu à partir de $\mathcal{A}(\mathcal{T}_{i_0})$ et du fragment d'automate représenté sur la figure 9.18, en fusionnant l'état noir de la figure 9.18 et l'état initial de $\mathcal{A}(\mathcal{T}_{i_0})$.

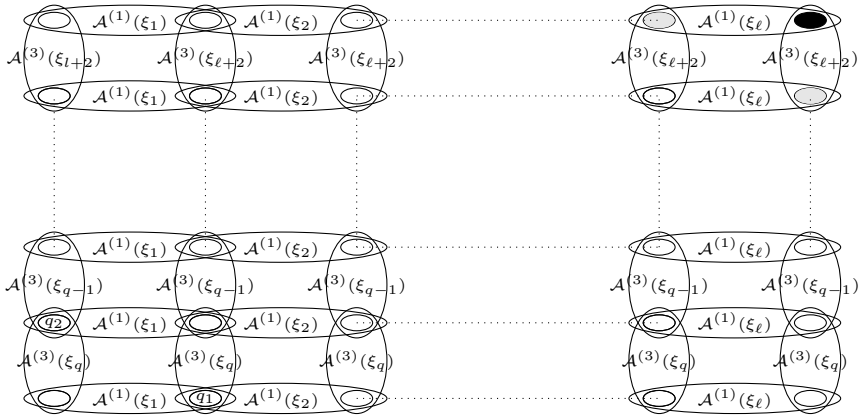


FIG. 9.18 – Fragment de l'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine linéaire ($\oplus$) dont un fil unique n'est pas un épi (montant). L'état initial est celui en bas à gauche.

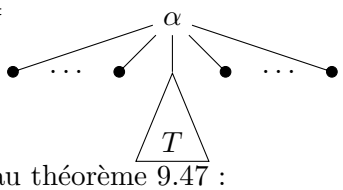
Lorsqu'une condition de la figure 9.10 est satisfaite par σ, l'automate a la même structure générale que dans le cas précédent. Certaines transitions viennent cependant s'ajouter, comme représenté dans les automates de la colonne de droite de la figure 9.10 (où les carrés sont une représentation compacte du fragment d'automate de la figure 9.18).

Les points de départ potentiels de ces nouvelles transitions sont repérés par les états gris sur la figure 9.18. Leurs points d'arrivée sont à l'intérieur de $\mathcal{A}(\mathcal{T}_{i_0})$, et correspondent à des états marqués dans $\mathcal{A}(\mathcal{T}_{i_0})$, qui sont chacun l'état initial d'un des automates représentés à l'intérieur de $\mathcal{A}(\mathcal{T}_{i_0})$ sur la figure 9.10.

La manière de marquer ces états sera explicitée au paragraphe 9.4.6. En supposant qu'on dispose du marquage, on vérifie sans difficulté et dans chacun des cas (aucune condition iHj satisfaite, ou pour chacune de ces conditions) que cette construction aboutit à un automate déterministe, avec un unique état final, qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ. De la même façon que pour le cas non récursif des permutations de racine linéaire, le temps de construction de $\mathcal{A}_\sigma$, aussi bien que sa taille, s'exprime en ajoutant $\mathcal{O}(n \cdot \sum_{i=1}^{q} |\xi_i|) = \mathcal{O}(n^2)$ au temps de construction ou à la taille de $\mathcal{A}(\mathcal{T}_{i_0})$.

9.4.4 Construction de l'automate pour les cas récursifs de racine première

Un unique fils de la racine n'est pas une feuille

Dans ce cas, on note comme au paragraphe 9.2 $\sigma =$ 

.

On rappelle la définition de la condition $\mathcal{C}$ donnée au théorème 9.47 :

$(\mathcal{C})$ $\begin{cases} \alpha \text{ est un quasi-épi montant (resp. descendant),} \\ T \text{ dilate un point de substitution auxiliaire de } \alpha, \text{ noté } x, \\ T \text{ est de la forme } \overset{\oplus}{\bullet}\!\!\diagup\!\!\overset{}{T'} \text{ (resp. } \overset{\ominus}{\bullet}\!\!\diagup\!\!\overset{}{T'} \text{) avec la feuille explicitée du côté déterminé par} \\ \text{la position du point auxiliaire dilaté de } \alpha, \text{ comme dans le lemme 9.45.} \end{cases}$

Dans le cas où la condition $\mathcal{C}$ du théorème 9.47 n'est pas satisfaite, ce théorème assure que l'automate de la figure 9.19 reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ. Sur cette figure, $Q_x(\alpha)$ n'est pas forcément un mot unique, mais les mots $Q_x(\alpha)$ coïncident tous à partir de leur troisième lettre, sauf dans le cas où x appartient à plusieurs cavaliers actifs de α. Dans ce cas particulier, soit α est de taille 5 ou 6, soit α est un quasi-épi dont T dilate le point de substitution principal (appartenant à deux cavaliers actifs). $Q_x(\alpha)$ se divise alors en deux (ou quatre si $|\alpha| = 5$) sous-ensembles de mots qui coïncident tous à partir de leur troisième lettre.

Le calcul de $Q_x(\alpha)$ demande d'identifier les cavaliers actifs de α contenant x. Pour le faire, on peut procéder comme dans la remarque 9.25, et comme on se restreint aux cavaliers actifs contenant x, le temps nécessaire pour calculer $Q_x(\alpha)$ est $\mathcal{O}(n)$. Le calcul de l'automate déterministe correspondant à $Q_x(\alpha)$ demande par les mêmes techniques que précédemment un temps $\mathcal{O}(n)$, et $\mathcal{O}(n)$ est aussi une borne sur la taille de cet automate.

FIG. 9.19 – L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont un fils unique T n'est pas une feuille, lorsque σ ne satisfait pas la condition $\mathcal{C}$.

Dans le cas où la condition $\mathcal{C}$ est satisfaite, et où $|T| \geq 3$, le même théorème assure que c'est l'automate un peu enrichi de la figure 9.20 qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ. Dans la figure 9.20, le branchement sur l'état initial de l'automate $\mathcal{A}(T')$ est effectué, comme dans le cas d'une racine linéaire, grâce à un marquage de l'état correspondant dans l'automate $\mathcal{A}(T)$. On verra comment obtenir ce marquage au paragraphe 9.4.6. En supposant qu'il est donné, de temps de la construction de l'automate $\mathcal{A}_\sigma$ est exactement le temps pour calculer $\mathcal{A}(T)$ auquel s'ajoute celui du calcul de $Q_x(\alpha)$ et de w_σ, qui est en $\mathcal{O}(n)$.

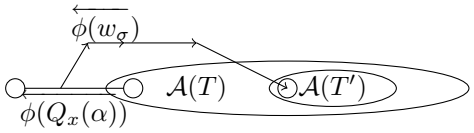
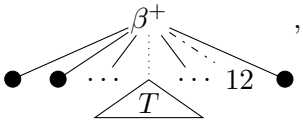


FIG. 9.20 – L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont un unique fils T n'est pas une feuille, lorsque σ satisfait la condition $\mathcal{C}$ et $|T| \geq 3$.

Dans le cas où la condition $\mathcal{C}$ est satisfaite et où $|T| = 2$, on est donc en présence d'une permutation σ qui est un quasi-épi montant dont le point de substitution auxiliaire est dilaté par 12. On s'aperçoit facilement avec le théorème 9.47 qu'une telle permutation admet exactement quatre représentations en épingles (deux lisant T en entier, et deux correspondant aux ensembles $P_{\{1,n\}}$ et $P_{\{2,n\}}$ du théorème 9.47), et donc un total d'au plus 32 mots d'épingles, calculables en temps $\mathcal{O}(n)$. On construit comme dans tous les cas précédents l'automate qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ sous la forme d'un *trie*, et en temps $\mathcal{O}(n)$. L'automate obtenu est aussi de taille $\mathcal{O}(n)$.

La racine possède deux fils non feuille

Il reste à examiner le cas où σ a une racine première dont deux fils ne sont pas réduit à des feuilles. Dans ce cas, $\sigma =$ 

, où β^+ est un quasi-épi montant, dont T dilate le

point de substitution principal et 12 le point de substitution auxiliaire.

Le théorème 9.48 montre que les mots d'épingles correspondant à σ sont très contraints. En reprenant les notations de ce théorème, l'automate qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ est donné sur la figure 9.21.

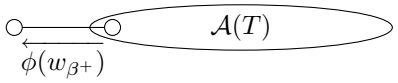


FIG. 9.21 – L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont deux fils ne sont pas des feuilles.

Le théorème 9.48 donne immédiatement que le langage de cet automate est celui des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ.

Au niveau de la complexité, le calcul de w_{β^+} et du *trie* associé est en $\mathcal{O}(n)$, et l'automate obtenu est donc de taille $|\mathcal{A}(T)| + \mathcal{O}(n)$.

9.4.5 Récapitulatif de complexité

Le tableau suivant récapitule la complexité des différentes étapes ci-dessus.

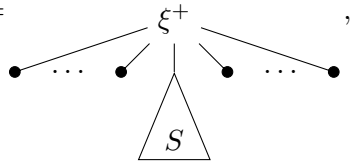
permutation	temps de construction	taille de l'automate	référence
taille 1	$\mathcal{O}(1)$	$\mathcal{O}(1)$	figure 9.14
simple	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$	lignes avant la figure 9.15
nœud $\oplus$ non récursif	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$	figure 9.17
nœud $\oplus$ récursif, un fils $\mathcal{T}$ non épi	$\mathcal{O}(n^2)$+ temps de construction pour $\mathcal{T}$	$\mathcal{O}(n^2)$+ taille de l'automate pour $\mathcal{T}$	figure 9.18
nœud premier récursif, $\mathcal{C}$ satisfaite, et $\mathcal{T}$ de taille 2	$\mathcal{O}(n)$	$\mathcal{O}(n)$	lignes après la figure 9.20
nœud premier récursif, un fils non feuille $\mathcal{T}$ (en dehors du cas précédent)	$\mathcal{O}(n)$+ temps de construction pour $\mathcal{T}$	$\mathcal{O}(n)$+ taille de l'automate pour $\mathcal{T}$	figures 9.19 et 9.20
nœud premier récursif, deux fils non feuille	$\mathcal{O}(n)$+ temps de construction pour $\mathcal{T}$	$\mathcal{O}(n)$+ taille de l'automate pour $\mathcal{T}$	figure 9.21

On peut remarquer que le temps de calcul pour les permutations simples σ est dominé par la recherche des mots d'épingles codant σ.

Proposition 9.59. *Pour toute permutation σ de taille n, la procédure ci-dessus permet de calculer en temps $\mathcal{O}(n^3)$ un automate déterministe qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à σ. Cet automate est de taille $\mathcal{O}(n^3)$.*

Pour que la preuve de la proposition 9.59 soit complète, il reste à justifier comment mettre sur les états des automates construits le marquage utilisé dans certains des cas de la construction.

9.4.6 Marquage des états

Pour pouvoir construire l'automate associé à σ dans le cas où σ satisfait la condition $1H1+$ de la figure 9.10, il faut savoir marquer, dans l'automate associé à $T =$ 

,

l'état q tel que l'automate qui se trouve après q pour l'ordre induit par le sens des transitions est $\mathcal{A}(S)$.

Si ξ^+ (qui représente un épi montant) est de taille au moins 5, ξ^+ est une permutation simple qui n'est pas un quasi-épi, et $\mathcal{A}(T)$ est donné par la figure 9.19 (le T de la figure désigne S ici). Il est alors facile de marquer l'état initial de $\mathcal{A}(S)$ dans $\mathcal{A}(T)$.

Si ξ^+ est de taille 4, il est possible que T satisfasse la condition $\mathcal{C}$. Si ça n'est pas le cas, on conclut comme au-dessus. Si au contraire T satisfait $\mathcal{C}$, on suppose d'abord que $|S| \geq 3$. L'automate $\mathcal{A}(T)$ est alors donné par la figure 9.20, et le marquage de l'état initial de $\mathcal{A}(S)$ dans $\mathcal{A}(T)$ est le même que dans le cas précédent.

Reste le cas où ξ^+ est de taille 4, où la condition $\mathcal{C}$ est satisfaite, et où $|S| = 2$. Dans ce cas, T est une permutation de taille 5, et on construit facilement un automate déterministe dont la taille est une constante, qui reconnait le langage des $\overleftarrow{\phi(v)}$ pour les mots d'épingles v associés à T, et où on marque un état q tel que l'automate qui se trouve après q pour l'ordre induit par le sens des transitions est $\mathcal{A}(S)$.

En dehors de ce cas, on remarque que dans la construction précédente, seuls les automates correspondant à des permutations de racine linéaire ont besoin d'avoir certains de leurs états marqués. Plus précisément, si T est un arbre de racine linéaire (sans perte de généralité, de racine $\oplus$), il faut savoir marquer au plus deux états.

Lorsque le fils de T le plus à gauche est une feuille, si on note T' l'arbre T privé de cette feuille, il faut savoir marquer dans l'automate $\mathcal{A}(T)$ l'état q tel que l'automate qui se trouve après q pour l'ordre induit par le sens des transitions est $\mathcal{A}(T')$. Sur les figures 9.17 et 9.18, qui sont les seuls cas possibles pour un arbre T de racine $\oplus$, ces états sont indiqués par q_1. Leur marquage est donc immédiat.

De même, lorsque le fils de T le plus à droite est une feuille, si on note T' l'arbre T privé de cette feuille, il faut savoir marquer dans l'automate $\mathcal{A}(T)$ l'état q tel que l'automate qui se trouve après q pour l'ordre induit par le sens des transitions est $\mathcal{A}(T')$. Ces états correspondent aux états q_2 sur les figures 9.17 et 9.18, et leur marquage est immédiat.

On propage récursivement ce marquage, lorsque la construction des automates est récursive. Ceci permet de pouvoir faire des branchements "à profondeur 2" comme dans le cas de la condition $2H3$ (où on repère non seulement l'état initial de $\mathcal{A}(S)$ mais aussi les états initiaux de $\mathcal{A}(T \cup a)$ et de $\mathcal{A}(T \cup b)$).

Le temps nécessaire au marquage est constant à chaque étape, et donc ne vient pas modifier la complexité annoncée pour la construction de $\mathcal{A}_\sigma$.

9.4.7 Assemblage de l'automate

On revient à la classe $\mathcal{C}$, donnée par sa base $\mathcal{B}$ de motifs exclus, qui est finie, et on considère l'ensemble $\mathcal{B}'$ des permutations de $\mathcal{B}$ qui sont des permutations en épingles (déterminé en temps $\mathcal{O}(k \cdot n^2)$, comme expliqué au paragraphe 9.4.1). On rappelle que n désigne la taille maximale d'un motif $\beta \in \mathcal{B}$, et k le nombre de motifs dans $\mathcal{B}$.

Pour chaque $\beta \in \mathcal{B}'$, on peut construire comme expliqué ci-dessus l'automate $\mathcal{A}_\beta$ qui reconnait le langage des $\overleftarrow{\phi(u^{(1)})\phi(u^{(2)})\dots\phi(u^{(j)})}$ pour les mots d'épingles u associés à β dont la décomposition en facteurs forts suivant un chiffre est $u = u^{(1)}u^{(2)}\dots u^{(j)}$. Le temps total pour cette construction est $\mathcal{O}(k \cdot n^3)$.

On rappelle que pour tout mot d'épingles u dont la décomposition en facteurs forts suivant un chiffre est $u = u^{(1)}u^{(2)}\dots u^{(j)}$, $\mathcal{L}(u) = \Sigma^* \cdot \phi(u^{(1)}) \cdot \Sigma^* \cdot \phi(u^{(2)}) \cdot \Sigma^* \dots \Sigma^* \cdot \phi(u^{(j)}) \cdot \Sigma^*$, où $\Sigma = \{L, R, U, D\}$.

Pour transformer $\mathcal{A}_\beta$ en un automate qui reconnait $\cup_{u \text{ codant } \beta} \overleftarrow{\mathcal{L}(u)}$, il suffit donc d'ajouter des boucles Σ^* sur les états qui correspondent à la fin de la lecture d'un facteur $\phi(u^{(i)})$ et au début de la lecture du facteur $\phi(u^{(i+1)})$ qui suit.

L'étude qui précède montre que les états de l'automate sur lesquels ces boucles doivent figurer sont

- les états appartenant à deux automates $\mathcal{A}^{(1)}$ et $\mathcal{A}^{(3)}$, y compris à l'intérieur des automates $\mathcal{A}^{(double)}$ (en remplaçant $\mathcal{A}^{(1)}$ ou $\mathcal{A}^{(3)}$ par $\mathcal{A}^{(int)}$) sur les figures 9.17 et 9.18,
- et l'état initial de $\mathcal{A}(T)$ sur les figures 9.19, 9.20 et 9.21.

Le point dont il faut se convaincre est que ces boucles sont suffisantes même vis-à-vis des transitions qui sont ajoutées autour de la structure principale des automates (dans $\mathcal{A}^{(double)}$, à partir des états gris de la figure 9.18, ou sur la figure 9.20 par exemple). Aucun de ces cas ne présente de difficulté, et les détails sont omis ici.

L'ajout de ces boucles ne préserve évidemment pas le déterminisme des automates, mais en appliquant la technique de A. Aho et M. Corasick dans [AC75] (qui étend l'algorithme de Knuth-Morris-Pratt), on peut ajouter des transitions de manière déterministe à l'automate, de sorte que le langage reconnu soit le même que pour l'automate $\mathcal{A}_\beta$ avec les boucles requises. Le temps nécessaire à cette construction est linéaire en la taille de l'automate (voir [AC75]), et donc en $\mathcal{O}(n^3)$ pour chaque $\beta \in \mathcal{B}'$.

Pour chaque motif $\beta \in \mathcal{B}'$, on dispose donc d'un automate déterministe, noté $\mathcal{A}^{\mathcal{L}}{}_\beta$, construit en temps $\mathcal{O}(n^3)$, dont la taille est aussi en $\mathcal{O}(n^3)$, et qui reconnait $\cup_{u \text{ codant } \beta} \overleftarrow{\mathcal{L}(u)}$. D'après le théorème 9.55, pour obtenir un automate qui reconnaisse les $\overleftarrow{\phi(w)}$, pour les mots d'épingles stricts w qui codent une permutation en épingles propre dont β est motif, il suffit de faire l'intersection avec le langage $\{\overleftarrow{\phi(w)} \mid w \text{ mot d'épingles strict}\}$. Ce langage est celui des mots de longueur au moins 2 sans facteur dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$, et son intersection avec $\cup_{u \text{ codant } \beta} \overleftarrow{\mathcal{L}(u)}$ est obtenue comme produit de $\mathcal{A}^{\mathcal{L}}{}_\beta$ avec l'automate de droite sur la figure 9.13. Cette opération ne change donc pas l'ordre de grandeur de la taille de l'automate. La construction de produit d'automates permet aussi d'en préserver le déterminisme.

On fait ensuite l'union des automates $\mathcal{A}^{\mathcal{L}}{}_\beta$ pour $\beta \in \mathcal{B}'$. On utilise une construction de produit pour réaliser l'union, de sorte à conserver des automates déterministes. Le temps nécessaire à cette opération est $\mathcal{O}((n^3)^k)$, et $\mathcal{O}((n^3)^k)$ est aussi la taille de l'automate obtenu, noté $\mathcal{A}_{\mathcal{B}'}$. L'automate $\mathcal{A}_{\mathcal{B}'}$ est déterministe et accepte les $\overleftarrow{\phi(w)}$ pour les mots d'épingles stricts w qui codent une permutation en épingles propre dont au moins un $\beta \in \mathcal{B}'$ est motif, c'est-à-dire qui n'appartiennent pas à $\mathcal{C}$.

Il suffit alors de complémenter $\mathcal{A}_{\mathcal{B}'}$, et de faire l'intersection avec l'automate des mots de longueur au moins 2 sans facteurs dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$. Le coût de cette opération est linéaire en la taille de l'automate, donc de l'ordre de $\mathcal{O}(n^{3k})$. L'automate obtenu est de taille $\mathcal{O}(n^{3k})$.

Ce dernier automate $\mathcal{A}^{\mathrm{c}}_{\mathcal{B}'}$ accepte les $\overleftarrow{\phi(w)}$ pour les mots d'épingles stricts w qui n'appartiennent pas à $\cup_{u \text{ codant } \beta \in \mathcal{B}'} \mathcal{L}(u)$, c'est-à-dire pour les mots d'épingles stricts w qui codent une permutation en épingles propre dont aucun $\beta \in B'$ n'est motif, c'est-à-dire qui appartiennent à $\mathcal{C}$.

Comme toute permutation en épingles propre admet au moins un codage par un mot d'épingles strict, et qu'il y a au plus 32 mots d'épingles stricts qui codent une permutation en épingles propre, le langage reconnu par $\mathcal{A}^{\mathrm{c}}_{\mathcal{B}'}$ est fini si et seulement s'il y a un nombre fini de permutations en épingles propres dans la classe $\mathcal{C}$.

On termine par la recherche d'un cycle accessible et co-accessible dans $\mathcal{A}^{\mathrm{c}}_{\mathcal{B}'}$: un tel cycle existe si et seulement si le langage de l'automate est infini si et seulement si $\mathcal{C}$ contient un nombre infini de permutations en épingles propres. Cette recherche est linéaire en la taille de $\mathcal{A}^{\mathrm{c}}_{\mathcal{B}'}$, de sorte que la complexité totale de l'algorithme donné ici est $\mathcal{O}(n^{3k})$.

En remettant cet algorithme dans son contexte (décrit en détail au paragraphe 9.1.2), on peut donc conclure que :

Théorème 9.60. *Étant donnée une classe de permutations $\mathcal{C}$, décrite par sa base finie $\mathcal{B}$ de motifs exclus, si on note n la taille maximale d'un motif dans $\mathcal{B}$ et k le nombre de motif de $\mathcal{B}$, l'algorithme*

présenté dans ce chapitre permet de tester si $\mathcal{C}$ contient un nombre fini de permutations simples en temps $\mathcal{O}(n^{3k})$.

Il faut remarquer cependant que cet algorithme ne permet pas de calculer les permutations simples qui appartiennent à $\mathcal{C}$, dans le cas où leur nombre est fini.

9.4.8 Cas particulier des classes fermées par produit de substitution

On termine ce chapitre en proposant une variante plus simple de cet algorithme dans le cas particulier des classes fermées par produit de substitution. L'algorithme ainsi modifié a une meilleure complexité : il fonctionne en temps $\mathcal{O}(n^2)$, n désignant la somme des tailles de motifs de la base.

On rappelle que les classes fermées par produit de substitution sont caractérisées au théorème 2.64 comme celles dont tous les motifs exclus sont des permutations simples, et que les mots d'épingles qui codent une permutation simple sont tous stricts ou quasi-stricts (remarque 9.8). La relation $u \trianglelefteq w$ a été étudiée en détails au paragraphe 9.3 lorsque u est strict ou quasi-strict, ce qui permet d'établir les résultats qui suivent.

Lemme 9.61. *Une classe de permutations fermée par produit de substitution $\mathcal{S}(\mathcal{B})$ contient des permutations en épingles propres arbitrairement longues si et seulement s'il existe des mots arbitrairement longs sur l'alphabet $\{L, R, U, D\}$ qui évitent l'ensemble de facteurs $\cup_{\beta\in\mathcal{B}} E(\beta) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$.*

Démonstration. $\mathcal{S}(\mathcal{B})$ contient des permutations en épingles propre arbitrairement longues si et seulement s'il existe des permutations en épingles propres arbitrairement longues qui n'ont pas de motif dans $\mathcal{B}$. D'après le théorème 9.58, ceci est encore équivalent au fait qu'il existe des mots d'épingles stricts w arbitrairement longs tels que $\phi(w)$ évite l'ensemble de facteurs $\cup_{\beta\in\mathcal{B}} E(\beta)$, c'est-à-dire à ce qu'il existe des mots arbitrairement longs sur l'alphabet $\{L, R, U, D\}$ qui évitent l'ensemble des facteurs $\cup_{\beta\in\mathcal{B}} E(\beta) \cup \{LL, LR, RR, RL, UU, UD, DD, DU\}$. □

Théorème 9.62. *Soit $\mathcal{S}(\mathcal{B})$ une classe de permutations fermée par produit de substitution ayant une base finie. Alors il existe un algorithme qui décide en temps $\mathcal{O}(n^2)$ où $n = \sum_{\beta\in\mathcal{B}} |\beta|$ si cette classe contient un nombre fini de permutations simples.*

Démonstration. D'après le lemme 9.17, on peut traiter séparément les permutations en chevron simples, les permutations parallèles et les permutations en épingles propres. Les permutations parallèles et les permutations en chevron simples peuvent être traitées en temps $\mathcal{O}(n \ln n)$ comme expliqué dans la conséquence 9.21.

Pour le cas des permutations en épingles propres, une première étape en $\mathcal{O}(n^2)$ (voir la remarque 9.25) consiste à déterminer les motifs de la base $\mathcal{B}$ qui sont des permutations en épingles, et à calculer les mots d'épingles correspondants. C'est en fait cette étape dont la complexité en temps est la plus grande.

On utilise ensuite le lemme 9.61 : décider s'il existe des mots arbitrairement longs sur l'alphabet $\{L, R, U, D\}$ qui évitent un ensemble fini de facteurs peut se faire en temps linéaire (en la somme des tailles des facteurs, soit ici en $\mathcal{O}(n)$), en utilisant l'algorithme de A. Aho et M. Corasick [AC75] pour construire un automate déterministe et complet. Alors il reste à tester si l'automate comporte un cycle ne contenant pas d'état final. Cette étape est aussi linéaire. □

Conclusion et perspectives

Pour conclure ce travail, on propose quelques perspectives et questions ouvertes relatives aux différents problèmes abordés. Ces développements sont regroupés par parties de la thèse.

Dans la continuité de la partie II, qui traite du problème NP-difficile de la recherche de motif dans les permutations, on doit évidemment envisager la question suivante : la recherche d'un motif de taille k dans une permutation de taille n peut-elle être résolue par un algorithme FPT, c'est-à-dire de complexité $\mathcal{O}(f(k)P(n))$, où P est un polynôme et f une fonction quelconque ? Un tel algorithme permettrait de concentrer tout le caractère exponentiel du problème sur le motif de taille k, la valeur de k étant supposée d'ordre inférieur à n pour une utilisation pratique.

La forme privilégiée d'un algorithme qui aurait cette complexité est la composée de deux étapes successives : la première consiste en un prétraitement du motif de taille k, dont la complexité n'est pas nécessairement polynomiale, et la seconde est un algorithme polynomial en n résolvant, à partir des données calculées dans la première phase, le problème d'occurrence de motif. Des algorithmes qui suivent ce schéma sont très souhaitables par exemple dans le contexte des bases de données, où on veut pouvoir tester une même requête (correspondant au motif) sur un grand nombre d'entrées.

La question de l'existence d'un algorithme FPT pour le problème de la recherche de motif dans les permutations est ouverte depuis plusieurs années maintenant, et aucune réponse générale n'a pour l'instant été apportée. Des réponses partielles ont été obtenues en donnant des algorithmes polynomiaux en n et k, lorsque sont introduites des restrictions à des classes de permutations : restriction par la classe des permutations séparables dans la partie II, puis récemment restriction par la classe $\mathcal{S}(321)$ dans [GV].

La partie II utilise les arbres de décomposition pour développer des algorithmes de complexité légèrement meilleure que les procédures naïves : même si cela n'apparaît pas explicitement dans la thèse, un élargissement de l'algorithme de Bose, Buss et Lubiw [BBL98] utilisant les arbres de décomposition au lieu des arbres de séparation (de la même façon que le paragraphe 4.3 étend le paragraphe 4.2) fournit un algorithme pour la recherche d'un motif σ de taille k dans une permutation de taille n, dont la complexité n'est plus de l'ordre de n^k mais plutôt de n^d, d étant l'arité maximale d'un nœud premier dans l'arbre de décomposition de σ. On peut s'interroger sur une utilisation des arbres de décomposition pour trouver un algorithme FPT résolvant le problème de recherche de motifs dans les permutations, ou au contraire pour démontrer qu'un tel algorithme n'existe pas.

Concernant la partie III, la nature des perspectives ouvertes par les deux chapitres 6 et 7 est très différente.

Pour le chapitre 6, consacré à une étude combinatoire du modèle de duplication en tandem avec perte aléatoire, on signale les travaux de J.-L. Baril et R. Vernay dans [BV], qui étendent ce principe d'analyse combinatoire, avec le point de vue des classes de permutations, à un modèle d'évolution des génomes plus général : le modèle de *duplication miroir avec perte aléatoire*. On pourrait envisager que d'autres modèles pour l'évolution des génomes admettent aussi des caractérisations des permutations obtenues après p étapes d'évolution comme classe de permutations. Dans ces modèles, il est souhaitable que les informations combinatoires obtenues, couplées avec un retour aux motivations biologiques, puissent être utilisées pour raffiner les modèles ou leur analyse.

Pour le chapitre 7, qui étudie le modèle du tri parfait par renversements, la confrontation avec des données réelles semble être possible à plus court terme. Une possibilité en ce sens serait d'utiliser l'étude des permutations séparables dans ce contexte pour détecter des zones du génome dont l'évolution n'est pas aléatoire, et qui correspondraient à des sous-arbres sans nœuds premiers des arbres de décomposition. On peut aussi proposer plusieurs développements théoriques à ce travail. D'abord, et toujours avec une motivation bio-informatique, on pourrait étendre une analyse de ce type aux arbres des intervalles communs d'un ensemble de $d \geq 3$ permutations. On peut aussi se demander dans quelle mesure la précision du travail sur les permutations séparables peut être étendue aux arbres possédant un unique nœud premier, éventuellement d'arité fixée. Enfin, les outils de combinatoire analytique employés pour l'étude des permutations séparables devraient permettre d'obtenir des résultats encore plus précis : par exemple, l'étude des moments d'ordre 2 autoriserait à mesurer à quel point le nombre de renversements et leurs tailles sont concentrés (ou au contraire ne le sont pas) autour de leur valeur moyenne.

Les contributions de la partie IV sont relatives principalement aux permutations en épingles. Depuis leur récente définition, ces permutations ont été utilisées dans d'autres travaux [Bri07, Bria, ARS], et la description précise de cette classe, ainsi que les outils employés pour l'obtenir, sont autant d'outils supplémentaires pour l'utilisation des permutations en épingles dans des travaux futurs.

La principale question ouverte par la partie IV ne concerne cependant pas les contributions apportées par cette thèse, mais le contexte général de cette étude. En effet, grâce aux travaux de [BRV08] améliorés par le chapitre 9, on dispose d'un algorithme polynomial testant si une classe $\mathcal{C}$ contient un nombre fini de permutations simples. Pour pouvoir utiliser la méthodologie de [AA05] autorisant à calculer la série génératrice (toujours algébrique) d'une telle classe $\mathcal{C}$, encore faut-il être capable de déterminer quelles sont les permutations simples de $\mathcal{C}$. Pour le moment, on n'a pas de meilleure solution que la procédure naïve consistant à chercher les permutations simples de $\mathcal{C}$ par taille croissante, et à s'arrêter quand pour deux tailles consécutives il n'y a aucune permutation simple dans la classe. La complexité désastreuse de cette méthode pose donc la question de la conception d'un algorithme plus efficace calculant les permutations simples d'une classe qui en possède un nombre fini.

Deux autres développements possibles de la méthode exposée dans [AA05] pour l'étude des classes contenant un nombre fini de permutations simples sont l'utilisation de techniques de combinatoire analytique, pour obtenir davantage de propriétés sur ces classes, et la génération aléatoire de permutations dans une classe, en particulier par la méthode de Boltzmann.

Le premier point s'adresse dans un premier temps aux classes fermées par produit de substitution, dont les arbres de décomposition sont tous ceux construits sur un ensemble de nœuds donnés. Les techniques de combinatoire analytique développées dans [FS08], très adaptées dans un tel cadre, devraient permettre non seulement de calculer les séries génératrices, mais aussi d'en extraire le taux de croissance de la classe, ou la valeur moyenne de certains paramètres raffinant son énumération. Dans un deuxième temps, il est envisageable d'étendre l'utilisation de ces techniques aux classes de permutations en général, pas seulement celles qui sont fermées par produit de substitution.

Concernant la génération aléatoire de permutations appartenant à une classe, la connaissance de sa série génératrice, et des équations qui ont conduit à son calcul fournissent le cadre idéal pour l'application de la méthode de Boltzmann (voir [Piv08] et ses références). Une première ébauche d'un travail en ce sens a été entamé avec Carine Pivoteau, et il ne semble pas y avoir d'obstacle majeur à la conception de générateurs de Boltzmann pour toutes les classes de permutations entrant dans le cadre de l'étude de [AA05].

On termine en proposant des perspectives à plus long terme, que l'on peut voir comme perspectives de la partie I, en ce sens qu'elles portent sur les objets étudiés (classes de permutations et arbres de décomposition) en général, et non sur un point précis de leur étude.

La vision des classes de permutations à travers les arbres de décomposition a mis en évidence le rôle crucial des permutations simples dans leur étude. Au delà des travaux de [AA05] qui relient la série génératrice d'une classe à la série génératrice des permutations simples de cette classe, on peut se demander quel est le rôle des permutations simples du point de vue énumératif. En particulier, il est naturel de vouloir estimer la proportion de permutations simples dans une classe. On sait (voir [AAK03]) que cette proportion est $\frac{1}{e^2}$ pour l'ensemble de toutes les permutations. Mais à l'intérieur d'une classe, les permutations simples semblent être toujours (sur les exemples étudiés) en proportion négligeable. Pour approfondir cette question, on pourrait s'interroger sur la définition du taux de croissance des permutations simples dans une classe, sous la forme d'une racine n-ème du nombre de permutations simples de taille n dans la classe, et sur la comparaison de ce taux de croissance à celui de la classe complète.

Enfin, on propose un développement à très long terme, qui envisage un retour de la décomposition des permutations à celle des graphes. Les travaux présentés dans cette thèse sur la décomposition par substitution des permutations peuvent-ils être adaptés pour obtenir des résultats du même ordre dans le cadre de la décomposition modulaire des graphes ? Une piste serait le développement de techniques similaires à celles présentées dans l'introduction de la partie IV, pour prouver des résultats de la forme "si une classe de graphes vérifie une propriété, alors une forme de structure est mise en évidence". On peut imaginer que des énoncés de ce type permettent de trouver dans certains cadres des méthodes de calcul de la base (toujours finie) de mineurs exclus qui définit une classe de graphes fermée par mineur. De façon plus immédiate, on peut transposer directement le théorème de [AA05] dans le cadre des graphes, et s'interroger sur les propriétés satisfaites par les classes de graphes contenant un nombre fini de graphes premiers, et sur les conséquences algorithmiques et combinatoires qu'un tel résultat pourrait avoir.

Annexe

Chapitre 10

Séries génératrices ordinaires : un outil pour la combinatoire énumérative

Sommaire

10.1 Définition et principes fondamentaux 254
10.2 Classes décomposables et méthode symbolique 255
10.3 Analyse de singularité et évaluation asymptotique 258
10.4 Séries bivariées et espérance de paramètres 259

La combinatoire analytique n'est pas au centre de cette thèse, mais on est amené à plusieurs reprises à utiliser les outils qu'elle propose pour résoudre des problèmes d'énumération. L'ouvrage [FS08] de Philippe Flajolet et Robert Sedgewick expose les tenants et les aboutissants de la combinatoire analytique, et mon propos n'est pas ici de fournir une introduction générale à cette approche. Je souhaite cependant donner, sous la forme d'une "boîte à outils", quelques techniques de base de cette méthode, pour permettre aux néophytes d'appréhender les outils utilisés dans cette thèse, et pour leur suggérer, à travers les résultats qu'on peut démontrer avec ces techniques simples, la puissance de la combinatoire analytique dans ses développements plus approfondis.

Le principe au départ de la combinatoire analytique est de capturer l'énumération d'une classe combinatoire dans une fonction (analytique), au moyen de la série génératrice associée à la classe. Les séries génératrices sont des séries formelles, dont la structure algébrique traduit la structure combinatoire des classes d'objets. Cependant, ces séries formelles peuvent être vues comme des fonctions d'une variable complexe (notée z dans cette annexe, parfois notée aussi t ou x), qui peuvent être étudiées avec des techniques d'analyse complexe pour obtenir des résultats qui ont un sens du point de vue combinatoire.

Les aspects analytiques sont présentés dans cette annexe comme des "boîtes noires" : les résultats seront énoncés (dans un formalisme allégé), mais surtout on verra comment utiliser ces résultats à des fins énumératives.

10.1 Définition et principes fondamentaux

On considère une *classe combinatoire* $\mathcal{C}$, c'est-à-dire un ensemble d'objets pour lesquels on dispose d'une notion de *taille* : la taille, notée $|.|$ est une fonction de $\mathcal{C}$ dans $\mathbb{N}$ qui vérifie que pour tout entier n, le nombre d'objets de taille n dans $\mathcal{C}$ est fini. On note alors $\mathcal{C}_n$ le sous-ensemble des objets de taille n dans $\mathcal{C}$, et c_n le cardinal de $\mathcal{C}_n$.

Les propriétés énumératives de la classe $\mathcal{C}$ sont celles de la séquence d'énumération $(c_n)_{n\geq 0}$. Cette séquence d'énumération est capturée par la série formelle $C(z)$, appelée *série génératrice* de la classe $\mathcal{C}$:

$$C(z) = \sum_{\gamma\in\mathcal{C}} z^{|\gamma|} = \sum_{n\geq 0} c_n z^n.$$

La série $C(z)$ donnée ci-dessus est la série génératrice *ordinaire* associée à la classe $\mathcal{C}$. On s'intéressera uniquement ici à ce type de séries génératrices, mais on mentionne pour mémoire qu'il existe aussi des séries génératrices *exponentielles*, particulièrement adaptées dans l'étude des objets étiquetés, et définies par $C(z) = \sum_{n\geq 0} c_n \frac{z^n}{n!}$.

Pour toute classe combinatoire $\mathcal{C}$, la série génératrice $C(z)$ associée est une série formelle, mais on peut la considérer comme une fonction de la variable complexe z, dans un certain rayon de convergence autour de l'origine. On peut alors étudier $C(z)$ en tant que fonction : description d'équations satisfaites par $C(z)$, calcul d'une formule close pour $C(z)$, étude du comportement asymptotique de $C(z)$ à l'approche de ses singularités, ... L'idée de la combinatoire analytique est d'utiliser ces propriétés de la fonction $C(z)$ pour obtenir des résultats combinatoires d'énumération sur la classe $\mathcal{C}$, en particulier pour le calcul exact ou l'estimation asymptotique des coefficients c_n.

Le transfert de propriétés de $C(z)$ (en tant que fonction d'une variable complexe) aux coefficients c_n de la série génératrice repose sur des résultats de l'analyse complexe. Il n'est cependant pas nécessaire de connaître l'analyse complexe pour *utiliser* les techniques de la combinatoire analytique : les méthodes basiques (les seules utilisées ici) peuvent être utilisées comme des "boîtes noires" transformant des propriétés de la série $C(z)$ en propriétés de ses coefficients c_n. L'ouvrage [FS08] justifie bien sûr ces résultats, mais propose aussi les "boîtes noires" dont il est question, sous la forme de tableaux de correspondance entre propriétés de $C(z)$ et propriétés des coefficients c_n.

Dans tout ce chapitre, les définitions et méthodes présentées seront illustrées par l'exemple d'une classe combinatoire : celle des chemins de Dyck. Les exemples correspondants seront indiqués entre les symboles $\triangleright$ et $\triangleleft$.

Définition. Un chemin de Dyck est un chemin du plan, commençant en $(0, 0)$, composé de pas montants $(1, 1)$ et de pas descendants $(1, -1)$, ne passant jamais sous l'axe des abscisses, et terminant en un point d'ordonnée 0.

Une conséquence immédiate de cette définition est qu'un chemin de Dyck est composé d'un nombre pair de pas, noté $2n$, dont une moitié sont montants et l'autre moitié sont descendants. L'entier n est appelé *demi-longueur* du chemin.

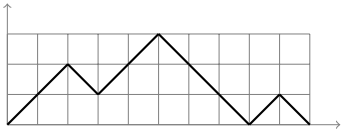


FIG. 10.1 – Un chemin de Dyck de demi-longueur 5.

Les chemins de Dyck sont en fait une représentation graphique des mots de parenthèses, les parenthèses ouvrantes (resp. fermantes) correspondant aux pas montants (resp. descendants).

À tout pas montant, on peut associer son pas descendant de la même façon qu'on associe sa parenthèse fermante à une parenthèse ouvrante dans un mot de parenthèses. Par exemple sur la figure 10.1 le premier pas (montant) est associé au huitième (descendant), le deuxième pas (montant) est associé au troisième (descendant) ...

▷ **Exemple [Chemins de Dyck].** La notion de taille usuelle sur la classe combinatoire des chemins de Dyck est la demi-longueur. On notera $\mathcal{D}$ la classe des chemins de Dyck, $\mathcal{D}_n$ l'ensemble (évidemment fini) des chemins de Dyck de taille n, et d_n le cardinal de $\mathcal{D}_n$. La série génératrice associée à la classe des chemins de Dyck est

$$\sum_{\gamma\in\mathcal{D}} z^{|\gamma|} = \sum_{n\geq 0} d_n z^n.$$

◁

Il est bien connu que les chemins de Dyck sont énumérés par les nombres de Catalan (séquence [A000108] dans [Slo07]) :

$$d_n = Cat(n) = \frac{1}{n+1}\binom{2n}{n}.$$

Cependant, on ne prendra pas ce résultat pour acquis, et on va voir comment appliquer les techniques de la combinatoire analytique à la classe des chemins de Dyck pour obtenir des propriétés énumératives de cette classe.

10.2 Classes décomposables et méthode symbolique

Une famille de classes combinatoires pour lesquelles l'utilisation des séries génératrices est particulièrement adaptée est la famille des classes *décomposables*.

Les classes décomposables sont celles construites (ou décrites) à partir d'objets vides (de taille 0) et d'atomes (objets de taille 1), et avec des opérations sur les classes, faisant intervenir aussi bien la classe $\mathcal{C}$ considérée que d'autres classes elles aussi décomposables. Parmi les opérations classiques sur les classes, on peut citer le produit cartésien, l'union disjointe, les opérations de séquence, de pointage, de substitution, ... auxquelles s'ajoutent des opérateurs de Pòlya comme le cycle, le multi-ensemble, l'ensemble, ...

L'*objet vide* (l'unique objet de taille 0) est noté ϵ.

L'*atome* (objet de taille 1, supposé unique) est noté $\mathcal{Z}$.

Le *produit cartésien* $\mathcal{A}\times\mathcal{B}$ est l'opération usuelle qui consiste à considérer les couples (a,b) d'éléments $a\in\mathcal{A}$ et $b\in\mathcal{B}$. Par extension, on peut aussi considérer les k-uplets d'éléments de $\mathcal{A}_1,\mathcal{A}_2,\ldots\mathcal{A}_k$, pour toute valeur de $k\geq 2$ fixée.

L'*union disjointe* $\mathcal{A}\uplus\mathcal{B}$ peut être envisagée (comme attendu) sur des ensembles disjoints, mais aussi sur des ensembles non disjoints : cela revient à faire l'union disjointe (au sens classique) de deux copies $\mathcal{A}'$ de $\mathcal{A}$ et $\mathcal{B}''$ de $\mathcal{B}$, l'appartenance d'un objet à $\mathcal{A}$ ou à $\mathcal{B}$ étant alors distinguée par un suffixe ($'$ pour les éléments de $\mathcal{A}$ et $''$ pour ceux de $\mathcal{B}$ sur cet exemple).

La *séquence* (ou suite) $Seq(\mathcal{A})$ associée à une classe $\mathcal{A}$ n'est définie que lorsque $\mathcal{A}$ ne contient pas l'objet vide (ϵ). C'est la classe formée de toutes les suites finies d'éléments de $\mathcal{A}$, c'est-à-dire de tous les k-uplets d'éléments de $\mathcal{A}$, pour toutes valeurs entières de $k\geq 0$. En particulier, $Seq(\mathcal{A})$ contient

- l'objet vide ϵ (suite de 0 élément de $\mathcal{A}$),
- la classe $\mathcal{A}$ (pour $k=1$),
- le produit cartésien $\mathcal{A}\times\mathcal{A}$ (lorsque $k=2$),
- $\mathcal{A}\times\mathcal{A}\times\mathcal{A}$, ...

et plus généralement $\mathcal{A}^k = \underbrace{\mathcal{A} \times \mathcal{A} \times \ldots \times \mathcal{A}}_{k}$ pour tout $k \geq 0$.

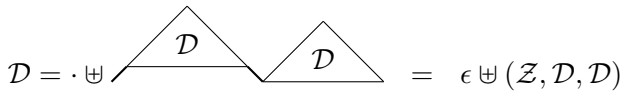
Plus précisément, $Seq(\mathcal{A})$ est l'union disjointe (ici infinie) de tous les $\mathcal{A}^k$ pour $k \in \mathbb{N}$.

Le *pointage* $\mathcal{A}^\bullet$ d'une classe $\mathcal{A}$ est utilisé dans le cas où $\mathcal{A}$ ne contient pas d'objet vide (ϵ). Tout élément de $\mathcal{A}$ est alors formé à partir d'atomes combinés entre eux. Le pointage d'un objet $\gamma \in \mathcal{A}$ consiste à particulariser (ou marquer, ou pointer) l'un de ces atomes. La classe $\mathcal{A}^\bullet$ est obtenue en considérant tous les pointages possibles de tous les objets de $\mathcal{A}$. Pour tout objet γ de $\mathcal{A}$, le nombre d'objets de $\mathcal{A}^\bullet$ qu'il génère est égal au nombre d'atomes dans γ, c'est-à-dire à $|\gamma|$.

La *substitution* $\mathcal{A} \circ \mathcal{B}$ d'une classe $\mathcal{B}$ dans une classe $\mathcal{A}$ consiste à remplacer chacun des atomes d'un objet de $\mathcal{A}$ par un élément de la classe $\mathcal{B}$, de taille quelconque. Dans la définition de $\mathcal{A} \circ \mathcal{B}$, on supposera que la classe $\mathcal{B}$ ne contient pas d'objet vide (ϵ).

Remarque. Dans le chapitre 8 (page 201), on substitue un élément d'une classe $\mathcal{B}$ dans un *unique* atome d'un objet d'une classe $\mathcal{A}$ (en l'occurrence une séquence d'au moins deux éléments), cet atome ayant été marqué au préalable par l'opération de pointage. Cette opération est moins standard, mais tout à fait admissible dans ce contexte.

▷ **Exemple [Chemins de Dyck].** On peut donner une description récursive des chemins de Dyck comme suit : un chemin de Dyck est soit le chemin vide (noté $\cdot$), soit commence par un pas montant. Dans ce cas, on repère le pas descendant associé. Cette paire de pas forme un atome, et délimite deux chemins de Dyck : l'un entre les deux pas, l'autre après le deuxième.



La classe $\mathcal{D}$ est décomposable et la décomposition qui en est proposée ci-dessus utilise seulement un objet vide, un atome, et récursivement la classe $\mathcal{D}$, avec les opérations d'union disjointe et de produit cartésien.

◁

Pour passer aux séries génératrices, il faut définir la taille d'un objet d'une classe $\mathcal{C}$ obtenue par chacune des opérations ci-dessus à partir des tailles des objets qui le composent. Cette définition est cohérente avec le principe que la taille d'un objet est égale au nombre des atomes qui le composent. Ici, les notions de tailles dans plusieurs classes cohabitent. Pour lever l'ambiguïté, on utilise la notation $|.|_{\mathcal{C}}$ pour indiquer la taille d'un objet dans la classe $\mathcal{C}$. Ainsi,

- $|\epsilon| = 0$
- $|\mathcal{Z}| = 1$
- pour tout $\gamma \in \mathcal{A} \uplus \mathcal{B}$, on a $|\gamma| = \begin{cases} |\gamma|_{\mathcal{A}} \text{ si } \gamma \in \mathcal{A} \\ |\gamma|_{\mathcal{B}} \text{ si } \gamma \in \mathcal{B}, \end{cases}$
- pour tout $\gamma = (\alpha, \beta) \in \mathcal{A} \uplus \mathcal{B}$, on a $|\gamma| = |\alpha|_{\mathcal{A}} + |\beta|_{\mathcal{B}}$,
- pour tout $\gamma = (\alpha_1, \ldots, \alpha_k) \in Seq(\mathcal{A})$, on a $|\gamma| = |\alpha_1|_{\mathcal{A}} + \ldots + |\alpha_k|_{\mathcal{A}}$,
- pour tout $\gamma^\bullet \in \mathcal{A}^\bullet$ obtenu à partir de $\gamma \in \mathcal{A}$, on a $|\gamma^\bullet| = |\gamma|_{\mathcal{A}}$,
- pour tout $\gamma \in \mathcal{A} \circ \mathcal{B}$ obtenu par substitution de k objets $\beta_1, \ldots, \beta_k$ dans un objet de $\mathcal{A}$ de taille k, on a $|\gamma| = |\beta_1|_{\mathcal{B}} + \ldots + |\beta_k|_{\mathcal{B}}$.

La *méthode symbolique* est un mécanisme formel qui offre une traduction directe des constructions combinatoires décrivant les classes décomposables en équations satisfaites par leur séries génératrices. Pour le cas des séries génératrices ordinaires, et pour quelques unes des opérations classiques, cette correspondance est résumée par le tableau suivant.

Opération	Sur les classes combinatoires	Sur les séries génératrices
Union disjointe	$\mathcal{A} \uplus \mathcal{B}$	$A(z) + B(z)$
Produit cartésien	$\mathcal{A} \times \mathcal{B}$	$A(z) \cdot B(z)$
Séquence	$Seq(\mathcal{A})$	$\frac{1}{1-A(z)}$
Pointage	$\mathcal{A}^\bullet$	$z \cdot \frac{\partial A(z)}{\partial z}$
Substitution	$\mathcal{A} \circ \mathcal{B}$	$A(B(z))$

À ce tableau, il faut ajouter les cas des objets de base (objet vide et atome) utilisés dans la description des classes décomposables :

Objets de base	Sur les classes combinatoires	Sur les séries génératrices
Objet vide	ϵ	1
Atome	$\mathcal{Z}$	z

Démonstration. On démontre à titre d'exemple que les équations sur les séries génératrices traduisent bien les décompositions des classes combinatoires, pour les opérations de produit cartésien et de pointage.
On considère donc deux classes $\mathcal{A}$ et $\mathcal{B}$, et la classe $\mathcal{C} = \mathcal{A} \times \mathcal{B}$. On note $A(z) = \sum a_n z^n$, $B(z) = \sum b_n z^n$ et $C(z) = \sum c_n z^n$ les séries génératrices qui leur sont respectivement associées. Par définition du produit cartésien, on a $c_n = \sum_{p=0}^{n} a_p b_{n-p}$. Ainsi,

$$C(z) = \sum_{n\geq 0}\sum_{p=0}^{n} a_p b_{n-p} z^n = \sum_{n\geq 0}\sum_{p=0}^{n} (a_p z^p)(b_{n-p} z^{n-p}) = \sum_{p\geq 0}(a_p z^p) \cdot \sum_{q\geq 0}(b_q z^q) = A(z) \cdot B(z).$$

On considère une classe $\mathcal{A}$ qui ne contient pas ϵ, et dont la série génératrice s'écrit donc $A(z) = \sum_{\gamma\in\mathcal{A}} z^{|\gamma|} = \sum_{n\geq 1} a_n z^n$. Par définition, la série génératrice de la classe pointée $\mathcal{A}^\bullet$ est

$$\sum_{\gamma\in\mathcal{A}} |\gamma| z^{|\gamma|} = \sum_{n\geq 1} n a_n z^n = z \sum_{n\geq 1} n a_n z^{n-1} = z\frac{\partial A(z)}{\partial z}.$$

□

Pour une classe décomposable, en appliquant la méthode symbolique on obtient donc une équation (ou un système d'équations) dont la série génératrice de la classe est solution.

▷ **Exemple [Chemins de Dyck].** L'équation décrivant la structure des chemins de Dyck se traduit sur les séries génératrices en

$$D(z) = 1 + z \cdot D(z)^2.$$

◁

Dans les cas les plus favorables, ces équations permettent de trouver des relations de récurrence sur les coefficients c_n de la série génératrice, que l'on peut résoudre pour donner une forme explicite des c_n, et ainsi obtenir une énumération explicite de la classe $\mathcal{C}$.

▷ **Exemple [Chemins de Dyck].** De l'équation $D(z) = 1 + z \cdot D(z)^2$, on déduit que

$$\sum_{n\geq 0} d_n z^n = 1 + z\big(\sum_{n\geq 0} d_n z^n\big)^2 = 1 + \sum_{n\geq 1}\big(\sum_{i=0}^{n-1} d_i d_{n-i-1}\big) z^n,$$

et donc on a l'équation de récurrence $d_n = \sum_{i=0}^{n-1} d_i d_{n-i-1}$ dès que $n \geq 1$, et la condition initiale $d_0 = 1$. Cette équation de récurrence caractérise bien les nombres de Catalan.

◁

Dans d'autres cas légèrement moins favorables (mais beaucoup plus nombreux), on peut extraire des équations satisfaites par $C(z)$ une formule close pour $C(z)$, considérée à présent comme fonction de la variable z.

L'équation satisfaite par $C(z)$ peut admettre plusieurs solutions formelles. Un moyen pour déterminer laquelle de ces solutions correspond à la série génératrice $C(z)$ cherchée est de considérer la valeur (ou la limite) de chaque solution en $z = 0$. Cette valeur (ou limite) doit par définition de la série génératrice être égale à c_0, c'est-à-dire au nombre d'objets de taille 0 dans la classe $\mathcal{C}$. Dans le cas (observé dans tous les exemples traités dans cette thèse) où une seule des solutions de l'équation satisfait cette contrainte, on obtient une formule close pour l'expression de la série génératrice $C(z)$ en tant que fonction de z.

▷ **Exemple [Chemins de Dyck].** L'équation satisfaite par la série génératrice $D(z)$ associée aux chemins de Dyck est une équation du second degré, qui admet deux solutions : $\frac{1+\sqrt{1-4z}}{2z}$ et $\frac{1-\sqrt{1-4z}}{2z}$. La première a une limite infinie en 0, et la seconde tend vers $1 = d_0$ lorsque z tend vers 0. On en déduit donc une formule close pour la série génératrice de la classe des chemins de Dyck :

$$D(z) = \frac{1 - \sqrt{1-4z}}{2z}.$$

◁

Le paragraphe suivant montre comment la donnée d'une formule close pour la fonction $C(z)$ permet d'obtenir des résultats d'énumération (non plus exacts mais asymptotiques) sur les coefficients c_n de la série génératrice.

10.3 Analyse de singularité et évaluation asymptotique

Une fois obtenue une formule close pour la série génératrice $C(z)$ d'une classe $\mathcal{C}$, on veut revenir à des propriétés énumératives, c'est-à-dire des propriétés sur les coefficients (c_n). Pour ce faire, on emprunte le chemin qui consiste à analyser $C(z)$ en tant que fonction de la variable complexe z.

On note habituellement $[z^n]C(z)$ le coefficient de z^n dans $C(z)$, c'est-à-dire $[z^n]C(z) = c_n$.

Les séries génératrices correspondent à des fonctions *régulières* (c'est-à-dire *analytiques* ou *holomorphes*) dans un disque de convergence autour de l'origine 0 du plan complexe.

Des irrégularités apparaissent quand on s'éloigne de l'origine : ce sont des *singularités* des fonctions considérées. Les séries génératrices ayant toujours des coefficients positifs (par définition), leurs singularités correspondent à des valeurs *réelles* de la variable z situées sur le rayon de convergence. Il s'agit en pratique des valeurs réelles pour lesquelles la formule close obtenue pour $C(z)$ passe d'un intervalle de réels où elle est définie à un intervalle où elle n'est pas définie. Ces singularités s'appellent des singularités *dominantes*.

▷ **Exemple [Chemins de Dyck].** La fonction $D(z) = \frac{1-\sqrt{1-4z}}{2z}$ n'est pas à proprement parler définie en 0, mais possède en 0 une limite finie (à savoir 1). On peut donc étendre la définition de $D(z)$ en 0 par $D(0) = 1$. La fonction $D(z)$ est alors analytique dans le disque de rayon $\frac{1}{4}$ centré en l'origine.

Pour les valeurs réelles de la variable z, la fonction $D(z)$ n'est définie que pour $z \leq \frac{1}{4}$. Ainsi, $D(z)$ possède une singularité dominante en $z = \frac{1}{4}$.

◁

Dans ce bref aperçu, on se place uniquement dans le cadre restreint où la singularité dominante est unique. On trouvera dans [FS08] les résultats concernant les cas avec plusieurs singularités dominantes.

La relation entre singularités et coefficients d'une série génératrice réside dans le fait que la singularité dominante détermine le comportement asymptotique des coefficients, et ce par deux aspects.

- D'une part, la valeur réelle de la variable z qui conduit à la singularité dominante donne le facteur exponentiel des coefficients $c_n = [z^n]C(z)$.
- D'autre part, la nature de cette singularité donne le facteur sous-exponentiel de l'équivalent asymptotique de $c_n = [z^n]C(z)$.

Pour le premier point, si la singularité dominante est obtenue pour la valeur $z = 1/\alpha$ avec $\alpha \geq 1$, alors un équivalent asymptotique du coefficient $[z^n]C(z)$ lorsque $n \to +\infty$ est de la forme $\alpha^n P(n)$, où $P(n)$ est une fonction de n, dominée par un polynôme en n, et qu'il reste à déterminer.

Le second point explicite la fonction $P(n)$ selon la nature de la singularité dominante. On calcule pour cela un équivalent asymptotique de $C(z)$ à l'approche de sa singularité dominante, et la forme de cet équivalent donne $P(n)$. Dans les cas résolus dans [FS08], $P(n)$ est de la forme $a \cdot n^d \cdot \log(n)^k$ pour un réel a, un nombre rationnel d et un entier relatif k.

Les résultats qui permettent de passer d'un équivalent asymptotique de $C(z)$ à l'approche de sa singularité dominante à un équivalent asymptotique des coefficients $c_n = [z^n]C(z)$ lorsque la taille n des objets tend vers $+\infty$ sont connus sous le nom de *propriétés de transfert*. Ces propriétés correspondent aux théorèmes VI.1 et VI.2, pages 381 et 385 de [FS08]. À la page 394 du même ouvrage, la figure VI.7 donne les cas les plus fréquents d'utilisation de ces propriétés de transfert. Cette thèse ne fait usage des propriétés de transfert que dans deux cas, résumés par l'extrait suivant de la figure VI.7 de [FS08] :

Nature de la singularité dominante	Coefficients de la série génératrice
$C(z) \sim \frac{1}{1-z}$	$c_n \sim 1$
$C(z) \sim \sqrt{1-z}$	$c_n \sim \frac{-1}{2\sqrt{\pi n^3}}$

▷ **Exemple [Chemins de Dyck].** La série génératrice des chemins de Dyck obtenue plus haut est $D(z) = \frac{1-\sqrt{1-4z}}{2z}$. La singularité dominante est obtenue pour $z = 1/4$. L'équivalent asymptotique de $d_n = [z^n]D(z)$ est donc de la forme $4^n P(n)$.

Lorsque $z \to 1/4$, un équivalent asymptotique de $D(z)$ est $2 - 2\sqrt{1-4z}$. La singularité dominante est donc du type $\sqrt{1-z}$, et par les propriétés de transfert on obtient donc l'équivalent suivant pour d_n lorsque $n \to +\infty$:

$$d_n \sim 4^n \times (-2) \cdot \frac{-1}{2\sqrt{\pi n^3}} \sim \frac{4^n}{\sqrt{\pi n^3}}.$$

◁

L'analyse de singularité, associée aux propriétés de transfert, est donc un moyen de trouver des propriétés asymptotiques sur la séquence d'énumération, à partir de la série génératrice.

10.4 Séries bivariées et espérance de paramètres

Un raffinement des séries génératrices ordinaires consiste à considérer des séries génératrices *bivariées*, où les objets ne sont plus seulement énumérés en fonction de leur taille, mais aussi en fonction de la valeur d'un second paramètre. Les techniques d'analyse de singularité permettent alors de calculer la valeur moyenne (ou espérance, ou moment d'ordre 1) du paramètre en question, et ses moments d'ordre supérieur (donnant accès entre autre à la variance du paramètre considéré).

De la même façon que la variable z "compte" la taille des objets dans une série génératrice ordinaire, on introduit dans une série génératrice bivariée une seconde variable, appelée variable catalytique et souvent notée u, qui "compte" le paramètre auquel on s'intéresse. La série génératrice bivariée associée à la classe $\mathcal{C}$ et à un paramètre $\mathcal{P}$ est alors définie par

$$C(z,u) = \sum_{n,p\geq 0} c_{n,p} u^p z^n,$$

où $c_{n,p}$ est le nombre d'objets de $\mathcal{C}$ de taille n pour lesquels le paramètre $\mathcal{P}$ considéré prend la valeur p.

▷ **Exemple [Chemins de Dyck].** Un paramètre que l'on peut analyser pour la classe $\mathcal{D}$ des chemins de Dyck est le nombre de retours d'un tel chemin sur l'axe $x = 0$.

Dans ce cas, la série génératrice bivariée $D(z,u)$ est $D(z,u) = \sum_{n,p\geq 0} d_{n,p} u^p z^n$, $d_{n,p}$ désignant le nombre de chemins de Dyck de demi-longueur n qui reviennent p fois sur l'axe des abscisses.

◁

Comme pour les séries génératrices ordinaires, dans le cas des classes décomposables, la structure de ces classes permet d'écrire des équations satisfaites par les séries génératrices bivariées. Il est cependant parfois nécessaire de remplacer l'une des variables u ou z par 1, par zu, ...

▷ **Exemple [Chemins de Dyck].** La décomposition des chemins de Dyck donnée page 256 justifie l'équation suivante sur $D(z,u)$:

$$D(z,u) = 1 + zu \cdot D(z,1) \cdot D(z,u).$$

En effet, si on considère un chemin de Dyck obtenu en composant deux chemins $\mathcal{D}_1$ et $\mathcal{D}_2$, qui ont respectivement n_1 et n_2 retours à zéro, le nombre de retours à zéro du chemin obtenu est $1 + n_2$ (et en particulier ne dépend pas de n_1).

◁

Lorsque l'équation satisfaite par la série génératrice bivariée considérée fait intervenir seulement $C(z,u)$ et $C(z,1)$, on peut utiliser la *méthode du noyau* par calculer une formule close pour $C(z,u)$. Elle consiste à calculer d'abord une formule close pour $C(z,1)$ en remplaçant u par 1 partout dans l'équation, puis à réinjecter le résultat obtenu dans l'équation satisfaite par $C(z,u)$, et enfin à isoler $C(z,u)$ pour en obtenir une forme explicite.

▷ **Exemple [Chemins de Dyck].** L'équation $D(z,u) = 1 + zu \cdot D(z,1) \cdot D(z,u)$ constitue un cadre idéal pour l'application de la méthode du noyau.

En remplaçant d'abord u par 1, on obtient l'équation $D(z,1) = 1 + zD(z,1)^2$, soit précisément l'équation satisfaite par $D(z)$ dans le paragraphe sur les séries génératrices ordinaires. On déduit donc que $D(z,1) = \frac{1-\sqrt{1-4z}}{2z}$.

En injectant la forme close obtenue pour $D(z,1)$, on aboutit à l'équation suivante, dont la seule inconnue est $D(z,u)$:

$$D(z,u) = 1 + zu \cdot \frac{1-\sqrt{1-4z}}{2z} D(z,u).$$

Sa résolution donne immédiatement

$$D(z,u) = \frac{2}{2-u+u\sqrt{1-4z}} = \frac{2-u-u\sqrt{1-4z}}{2-2u+2u^2 z}.$$

◁

Remarque. Le fait qu'on retrouve l'équation satisfaite par $D(z)$ lorsque l'on remplace u par 1 n'a rien d'une coïncidence. De façon générale, pour toute classe combinatoire $\mathcal{C}$, on a $C(z) = C(z,1)$. La raison en est que, par définition, $c_n = \sum_p c_{n,p}$.

En suivant sa définition, on peut exprimer la valeur moyenne (c'est-à-dire l'espérance) du paramètre $\mathcal{P}$ sur les objets de taille n par la formule suivante, faisant intervenir la série $C(z,u)$:

$$\mathbb{E}(\mathcal{P}_n) = \frac{\sum_p p c_{n,p}}{\sum_p c_{n,p}} = \frac{[z^n]\frac{\partial C(z,u)}{\partial u}|_{u=1}}{[z^n]C(z,1)}.$$

Si on a pu trouver une formule close pour la série génératrice bivariée $C(z,u)$ (que ce soit par la méthode du noyau ou par une méthode spécifique à la classe considérée), alors on peut obtenir une estimation asymptotique de $\mathbb{E}(\mathcal{P}_n)$ en appliquant les techniques d'analyse de singularité et en utilisant les propriétés de transfert d'une part pour la fonction $\frac{\partial C(z,u)}{\partial u}|_{u=1}$ et d'autre part pour $C(z,1)$.

On peut remarquer qu'une formule close pour $C(z,u)$ est suffisante, mais pas nécessaire. On peut se contenter d'exprimer seulement $\frac{\partial C(z,u)}{\partial u}|_{u=1}$ et $C(z,1)$ en tant que fonctions de z. C'est ainsi qu'on procède dans le chapitre 7 (pages 161 et 162).

On obtient alors des équivalents asymptotiques des deux quantités $[z^n]\frac{\partial C(z,u)}{\partial u}|_{u=1} = \sum_p p c_{n,p}$ et $[z^n]C(z,1) = c_n = \sum_p c_{n,p}$, et en faisant leur quotient, on déduit une estimation asymptotique de la valeur moyenne du paramètre $\mathcal{P}$ sur les objets de taille n quand $n \to +\infty$.

▷ **Exemple [Chemins de Dyck].** On a vu au paragraphe précédent que

$$[z^n]D(z) = [z^n]D(z,1) \sim \frac{4^n}{\sqrt{\pi n^3}} \quad \text{lorsque n } \to +\infty.$$

Il reste à analyser le comportement asymptotique de $[z^n]\frac{\partial D(z,u)}{\partial u}|_{u=1}$, par analyse de singularité puis en appliquant les propriétés de transfert.

En dérivant, on obtient $\frac{\partial D(z,u)}{\partial u} = \frac{2-2\sqrt{1-4z}}{(2-u+u\sqrt{1-4z})^2}$, et donc

$$\frac{\partial D(z,u)}{\partial u}|_{u=1} = \frac{2-2\sqrt{1-4z}}{(1+\sqrt{1-4z})^2} = \frac{1-3z+(z-1)\sqrt{1-4z}}{2z^2}.$$

La singularité dominante de $\frac{\partial D(z,u)}{\partial u}|_{u=1}$ correspond à $z = \frac{1}{4}$, et à l'approche de cette singularité, on a :

$$\frac{\partial D(z,u)}{\partial u}|_{u=1} \sim 2 - 6\sqrt{1-4z}.$$

Avec les propriétés de transfert, on conclut que lorsque $n \to +\infty$ on a :

$$[z^n]\frac{\partial D(z,u)}{\partial u}|_{u=1} \sim 4^n \times (-6) \cdot \frac{-1}{2\sqrt{\pi n^3}} \sim 3 \cdot \frac{4^n}{\sqrt{\pi n^3}}.$$

En quotientant, on conclut que le nombre moyen de retours sur l'axe des abscisses d'un chemin de Dyck de taille $n \to +\infty$ est

$$\frac{[z^n]\frac{\partial D(z,u)}{\partial u}|_{u=1}}{[z^n]D(z,1)} \sim \frac{3 \cdot 4^n/\sqrt{\pi n^3}}{4^n/\sqrt{\pi n^3}} \sim 3.$$

◁

L'étude de la dérivée d'ordre 1 par rapport à u de $C(z,u)$ permet donc d'estimer l'espérance du paramètre $\mathcal{P}$ compté par la variable u, c'est-à-dire le moment d'ordre 1 de $\mathcal{P}$. En étudiant les dérivées d'ordre supérieur de $C(z,u)$ par rapport à u, on obtient de la même façon des résultats sur les moments d'ordre supérieur de $\mathcal{P}$. En particulier la dérivée d'ordre 2 donne par les mêmes techniques une estimation asymptotique de la variance du paramètre $\mathcal{P}$, qui permet donc de cerner plus précisément le comportement de ce paramètre.

Table des figures

1.1 Représentation sur deux lignes d'une permutation. 11
1.2 Représentation graphique de la permutation $\sigma = 8\,3\,1\,9\,2\,6\,4\,5\,7$. 13
1.3 La notion de motif sur la représentation graphique des permutations. 16
1.4 Les opérations de miroir, de complément et d'inverse, sur la représentation graphique des permutations. 19

2.1 Un graphe G possédant trois composantes connexes, et où le sommet x a deux voisins. 34
2.2 Exemples de modules dans un graphe. 35
2.3 Décomposition modulaire d'un graphe et arbre de décomposition modulaire. 36
2.4 À gauche, les intervalles d'une permutation, sur sa représentation en une ligne. Au centre, ces mêmes blocs sur sa représentation graphique. À droite, un bloc est un carré dans la représentation graphique qui n'admet aucun point de la permutation sur ses côtés. 39
2.5 Une permutation simple (à gauche), et une permutation contenant un intervalle de taille 2 (à droite). 40
2.6 L'opération de dilatation sur la représentation graphique des permutations. 42
2.7 L'arbre de décomposition de la permutation $\sigma = 10\,13\,12\,11\,14\,1\,18\,19\,20\,21\,17\,16\,15\,4\,8\,3\,2\,9\,5\,6\,7$. 44
2.8 Les intervalles de la permutation $\sigma = 5\,1\,10\,9\,6\,7\,8\,11\,2\,4\,3$, avec les intervalles forts représentés en gras. L'ordre d'inclusion de ces intervalles forts est représenté par l'arbre des intervalles communs de σ, à droite. 48
2.9 Preuve du théorème 2.54 : contradiction dans le cas d'un nœud linéaire. 51
2.10 Preuve du théorème 2.54 : contradiction dans le cas d'un nœud premier. 51
2.11 Déroulement de l'algorithme 2.1 pour la permutation $\sigma = 2\,3\,4\,10\,1\,7\,5\,9\,8\,6$. 55

3.1 Le graphe de permutation associé à $\tau = 4\,1\,3\,2\,6\,5$. 70
3.2 Deux arbres de séparation pour $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$. 72
3.3 L'arbre de séparation contracté de la permutation $\sigma = 8\,5\,6\,7\,9\,1\,2\,3\,4$. 74
3.4 L'arbre de décomposition et l'arbre de décomposition développé de la permutation $\sigma = 3\,1\,4\,2\,9\,6\,7\,8\,5\,11\,17\,10\,15\,14\,13\,12\,16$. 75

4.1 Tri d'une permutation par une pile. 84
4.2 La bijection entre permutations triables par pile et arbres plans enracinés sur un exemple. 85
4.3 Décomposition d'un plus grand motif commun à deux permutations, dont une séparable. 89

5.1 Fusion et séparation de motifs séparables qui possèdent des occurrences dans $K = 2$ permutations, dans des intervalles de positions et de valeurs prescrits. 98
5.2 Les arbres de décomposition des permutations τ_i dans la réduction du problème du stable au problème de recherche de motif séparable commun. 102

5.3 Le graphe G utilisé comme exemple à la réduction du problème du stable à celui de recherche de motif séparable commun. 103
5.4 Plusieurs décompositions de τ en $u_1\tau_{i_1}u_2\tau_{i_2}\dots u_k\tau_{i_k}u_{k+1}$, lorsque σ a plusieurs occurrences $\tau_{i_1}\dots\tau_{i_k}$ dans τ. 105

6.1 Exemple d'une étape de duplication en tandem avec perte aléatoire, de largeur 4. . 114
6.2 Un exemple de scénario de duplication-perte. 115
6.3 Décomposition d'une permutation minimale à d descentes en séquences non vides de descentes séparées par des montées isolées. 119
6.4 Les éléments $\sigma_{i-1}\sigma_i\sigma_{i+1}\sigma_{i+2}$ autour d'une montée $\sigma_i\sigma_{i+1}$ dans une permutation σ minimale avec d descentes. 120
6.5 Un poset représentant un ensemble de permutations minimales à 16 descentes et 4 montées (par conséquent de taille 21). On donne aussi une permutation qui appartient à cet ensemble, et sa représentation graphique. 121
6.6 (a) La permutation $\sigma = 2\,1\,5\,3\,7\,4\,9\,6\,10\,8$ qui est minimale avec $d = 5$ descentes et qui est de taille $2d = 10$, (b) le poset qui représente l'ensemble des permutations minimales à $d = 5$ descentes et de taille $2d = 10$ et (c) l'étiquetage de ce poset associé à σ. 122
6.7 Un étiquetage du poset en échelle à $d = 3$ marches, et ses enfants dans la construction ECO. 123
6.8 Les quatre premiers niveaux de l'arbre de génération associé à la construction ECO pour les étiquetages des posets en échelle décrite ici. 124
6.9 Illustration sur un exemple de la bijection décrite entre les permutations minimales à d descentes de taille $2d$ (vues comme des étiquetages du poset en échelle à d marches) et les chemins de Dyck de demi-longueur d. 125
6.10 Une condition sur les étiquetages des posets en échelle pour qu'ils satisfassent les contraintes d'ordre imposées par le poset. 126
6.11 Représentation schématique d'une partition (A, B) satisfaisant la propriété du diamant. 127
6.12 Les partitions de $\{1, 2, \dots, d+2\}$ en deux parts de cardinal compris entre 2 et d qui ne satisfont pas la propriété du diamant. 128
6.13 Les différentes formes des permutations dans les ensembles S^1 et S^2. 129
6.14 Illustration de la définition de la bijection Φ^1 sur un exemple. 129
6.15 La classification des permutations de S^2 en cinq types, de A à E. 130
6.16 Définition de la bijection Φ^2 pour s tel que $|t| = 1$. 131
6.17 Définition de la bijection Φ^2 pour s tel que $t_1 < t_2 < s_{n-1} < s_n$ ou $t_1 < s_{n-1} < t_2 < s_n$. 132
6.18 Définition de la bijection Φ^2 pour s tel que $s_{n-1} < t_1 < t_2 < s_n$. 132
6.19 Définition de la bijection Φ^2 pour s tel que $t_1 < s_{n-1} < s_n < t_2$. 133
6.20 Définition de la bijection Φ^2 pour s tel que $s_{n-1} < t_1 < s_n < t_2$. 134
6.21 Définition de $\Phi^2(s)$ pour quelques exemples de sous-ensembles non intervallaires s de $\{1, 2, \dots, 6\}$ ($d = 5$), qui illustrent tous les cas possibles dans la construction de Φ^2. 135
6.22 Décomposition $\sigma = 12\dots p_1\widehat{\sigma}p_2(p_2+1)\dots n$ sur la représentation graphique de σ, et forme de $\widehat{\sigma}$. 137
6.23 *vp*-vecteurs et *vp*-domaine de $\sigma = 4\,1\,2\,3\,5\,7\,6$, sur sa représentation linéaire et sa représentation graphique. 138
6.24 Cas général de la variation de la taille des *vp*-vecteurs lors de la suppression d'un élément j d'un côté ou de l'autre de la diagonale. 139
6.25 Variation de la taille des *vp*-vecteurs lors de la suppression d'un élément j sur un exemple. 139

6.26 Le seul point fixe qui peut apparaître lors de la suppression d'un élément quasi-diagonal. . . . 140
6.27 Décomposition d'une permutation comme alternance de fenêtres libres et de *vp*-fenêtres. . . . 141
6.28 Trouver des bornes sur $c_p(\sigma)$. . . . 146

7.1 Arbre signé de la permutation $\sigma = 5\,\overline{6}\,\overline{7}\,9\,4\,\overline{3}\,1\,2\,\overline{8}\,\overline{10}\,\overline{17}\,13\,\overline{15}\,12\,11\,\overline{14}\,18\,\overline{19}\,\overline{16}$. . . . 154
7.2 Graphique représentant les valeurs $p_n = \frac{\sum_p 2^p T_{n,p}}{n!}$, jusqu'à $n = 25$. . . . 157
7.3 Description récursive de la classe combinatoire des arbres de Schröder. . . . 159

8.1 Représentation graphique de la permutation $\sigma = 12\,13\,11\,3\,1\,7\,10\,2\,9\,8\,5\,6\,4$ et boîte englobante de l'ensemble des points $\{7, 2, 9, 5, 6\}$. . . . 172
8.2 Une représentation en épingles de la permutation $\sigma = 1\,8\,3\,6\,4\,2\,5\,7$. Toutes les épingles $p_3, \ldots, p_8$ sont séparantes, sauf p_6 qui est indépendante. . . . 173
8.3 Deux représentations en épingles de $\sigma = 1\,8\,3\,6\,4\,2\,5\,7$. . . . 174
8.4 Les différents cas dans la preuve du lemme 8.7 sur une représentation en épingles p de $\sigma = 8\,1\,3\,4\,2\,7\,5\,6$. . . . 174
8.5 Un épi montant de taille 10 et un épi descendant de taille 11, chacun donnés avec une représentation en épingles. . . . 177
8.6 Deux exemples de quasi-épis de taille 10 et 11. . . . 178
8.7 Représentations graphiques des quasi-épis de taille 4, 5 et 6. . . . 179
8.8 L'arbre de décomposition $\mathcal{T}_\sigma$ et une représentation en épingles de la permutation $\sigma = 5\,4\,1\,2\,6\,3$. . . . 181
8.9 Une vision graphique de la manière dont une représentation en épingles p parcourt une permutation σ, lorsque la racine de $\mathcal{T}_\sigma$ est un nœud linéaire de signe $\oplus$. . . . 182
8.10 Tout nœud de l'arbre de décomposition $\mathcal{T}_\sigma$ d'une permutation σ est la racine de l'arbre de décomposition $\mathcal{T}_\pi$ d'un motif π de σ. . . . 186
8.11 La permutation π autour de ses deux fils non feuilles B et C. . . . 186
8.12 La seule permutation en épingles (à symétrie près) dont l'arbre de décomposition a une racine première (d'arité au moins 6) possédant deux fils qui ne sont pas des feuilles. . . . 187
8.13 Les quatre configurations possibles de paires de points formant un cavalier. . . . 190
8.14 Représentations en épingles jusqu'à une épingle p_i, pour $i < n$, dans la démonstration du lemme 8.45. . . . 190
8.15 Les permutations en épingles simples de taille $n \leq 6$ et leurs cavaliers actifs. . . . 191
8.16 Un épi et un quasi-épi dont les cavaliers actifs sont marqués. . . . 192
8.17 Cavaliers $\{p_1, p_2\}$ et $\{p_1, p_i\}$. . . . 192
8.18 Les différents cas pour le cavalier actif $\{p_1, p_i\}$. . . . 192
8.19 Cas du cavalier actif $\{p_i, p_n\}$, avec $i \geq 4$. . . . 194
8.20 Cas du cavalier actif $\{p_3, p_n\}$. . . . 195
8.21 Parmi les 9 positions possibles du point fictif servant d'origine au codage d'une représentation en épingles propre p par un mot d'épingles, 4 seulement donnent lieu à des mots d'épingles stricts. . . . 196
8.22 Les deux types de permutations qui ne sont pas simples mais admettent des représentations en épingles propres. . . . 197
8.23 La permutation $1\,5\,2\,4\,3$, qui est le point de départ de la construction de chacune des permutations σ_n, dont l'ensemble forme une antichaîne infinie dans la base de la classe des permutations en épingles. . . . 203
8.24 Les permutations σ_n pour $n = 8, 9, 10, 14$ et 15. . . . 204

9.1 Les placements possibles de p_0 par rapport à p_1 et p_2 qui autorisent le codage par un mot d'épingles d'une représentation en épingles $p = (p_1, p_2, \ldots, p_n)$. 209
9.2 Les représentations en épingles associées aux mots d'épingles u et w de l'exemple 9.11. 210
9.3 De gauche à droite : deux permutations parallèles, une permutation en chevron simple de type 1 et une permutation en chevron simple de type 2. 212
9.4 Les épis montants de taille au plus 4, et deux exemples d'épis montants de taille 8 (de type (V, V)) et 9 (de type (V, H)), dont les cavaliers actifs sont marqués. 217
9.5 Les configurations telles que la permutation $\oplus[\xi_g, 1]$ admet des représentations en épingles où ξ_g est lu en deux fois, pour un épi montant ξ_g. 219
9.6 Démonstration des lemmes 9.39 et 9.40. 221
9.7 Le bloc $\mathcal{T}_{i_0}$ est lu en deux fois. 222
9.8 Les douze formes possibles d'un bloc $\mathcal{T}_{i_0}$ dans une permutation en épingles σ de racine $\oplus$ conduisant à la possibilité d'une lecture en deux fois de $\mathcal{T}_{i_0}$. 222
9.9 Les douze formes possibles de l'arbre de décomposition $\mathcal{T}$ d'une permutation en épingles σ de racine $\oplus$ pour lesquelles il existe un bloc $\mathcal{T}_{i_0}$ et une représentation en épingles de σ qui lit $\mathcal{T}_{i_0}$ en deux fois. 223
9.10 Les huit formes possibles (à symétrie près) d'un bloc $\mathcal{T}_{i_0}$ dans une permutation en épingles σ qui peut être lu en deux fois par une ou par plusieurs représentations en épingles de σ. 224
9.11 Positions possibles de p_n dans la preuve du lemme 9.45. 228
9.12 La forme des permutations σ ayant une racine première dont un seul fils T n'est pas une feuille, et telles qu'il existe une représentation en épingles de σ qui lit T en deux fois. 229
9.13 À gauche, un automate déterministe acceptant le langage des mots d'épingles stricts. À droite, un automate déterministe acceptant le langage des mots de longueur au moins 2 sans facteurs dans $\{UU, UD, DU, DD, RR, RL, LR, LL\}$. Ce langage correspond à l'ensemble des miroirs des $\phi(w)$ associés aux mots d'épingles stricts w. . . . 237
9.14 Automate reconnaissant le langage des miroirs des $\phi(w)$ pour les mots d'épingles w codant la permutation 1. 238
9.15 Exemple de construction du squelette de l'automate de A. Aho et M. Corasick pour l'ensemble de mots $\{LULULUR, LULDLD, LURD\}$. 239
9.16 La forme des automates $\mathcal{A}^{(double)}(\xi_j, \xi_{j+1})$. 239
9.17 L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine linéaire dont tous les fils sont des épis. 240
9.18 Fragment de l'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine linéaire $(\oplus)$ dont un fil unique n'est pas un épi (montant). 241
9.19 L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont un fils unique T n'est pas une feuille, lorsque σ ne satisfait pas la condition $\mathcal{C}$. 242
9.20 L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont un unique fils T n'est pas une feuille, lorsque σ satisfait la condition $\mathcal{C}$ et $|T| \geq 3$. 242
9.21 L'automate $\mathcal{A}_\sigma$ pour une permutation σ de racine première dont deux fils ne sont pas des feuilles. 243

10.1 Un chemin de Dyck de demi-longueur 5. 254

Index

antichaîne, 17
arbre, 14
arbre
    de composition, 32
    de décomposition, 44
    de décomposition développé, 75
    de génération, 123
    de séparation, 72
    de séparation contracté, 74
    de Schröder, 73
    de Schröder signé, 74
    des intervalles communs, 48
    non ordonné, 14
    plan, 14
arité, 14

base d'une classe de permutations, 17
bloc, 38
boîte englobante, 172

cerise, 60
classe
    combinatoire, 254
    décomposable, 255
    de permutations, 16
    de symétrie, 19
    fermée par produit de substitution, 56
    symétrique, 20
clique, 34
complément, 18
condition
    d'extériorité, 173
    d'indépendance, 173
    de séparation, 173
conjecture de Stanley-Wilf, 21
côtés de la boîte englobante, 172
cycle, 12

décomposition
    modulaire, 36
    par substitution, 32
    par substitution des permutations, 43
descente, 117
diagramme, 173
diamant, 120
dilatation, 41
direction
    d'un épi, 177
    d'un quasi-épi, 180
duplication en tandem avec perte aléatoire, 114

élément quasi-diagonal, 139
enumeration schemes, 28
épi, 177
épi
    descendant, 177
    montant, 177
épingle, 173
épingle
    indépendante, 173
    séparante, 173

facteur
    fort suivant un chiffre, 210
    suivant un chiffre, 210
fermeture par produit de substitution, 56
algorithmes FPT, 153

graphe
    complémentaire, 34
    de permutation, 70
    premier, 35
groupe symétrique, 12

insertion encoding, 28
intervalle, 38
intervalle
    d'une permutation signée, 153
    fort, 39
    fort maximal, 39
    minimal, 39
    trivial, 40
inverse, 18
inversion, 148
isomorphes en ordre, 15

méthode
    ECO, 123

symbolique, 256
miroir, 18
module, 34
module
fort, 35
fort maximal, 35
trivial, 34
montée, 117
mot d'épingles, 176
mot d'épingles
quasi-strict, 209
simple, 189
strict, 176
motif, 15
motif
à tirets, 26
barré, 26

nœud
linéaire, 45
premier, 45
nombres
de Schröder, 73
normalisation, 15

permutation, 10
permutation
$\ominus$-indécomposable, 42
$\oplus$-indécomposable, 42
alternante, 211
commutante, 158
en épingles, 174
en épingles propre, 211
en chevron, 211
exceptionnelle, 41
identité, 10
minimale à d descentes, 118
parallèle, 211
quotient, 49
séparable, 71
signée, 152
simple, 40
simple en chevron de type 1, 211
simple en chevron de type 2, 211
triable par pile, 84
permuting machine, 115
plus grand sous-arbre commun, 85
point
actif, 180
de substitution auxiliaire, 178
de substitution principal, 178
extérieur, 178
poset
en échelle, 122
position, 11
PQ-arbre, 49
produit de mélange, 219
produit de substitution (ou en couronne), 56

quadrant, 172
quasi-épi, 178
quasi-épi
descendant, 180
montant, 180

renversement, 152
représentation
en épingles, 173
en épingles propre, 173
en épingles simple, 189
en produit de cycles, 12
graphique, 13
linéaire, 11
sur deux lignes, 10

scénario parfait, 153
singularité, 258
singularité dominante, 258
sous-arbre, 14
sous-ensemble non intervallaire, 128
sous-séquence croissante maximale, 117
squelette, 42
stable, 34
structure
décomposable, 32
dégénérée, 32
de type infini, 32
linéaire, 32
première, 32
quotient, 32
substitution, 41

taille
d'une permutation, 10
pour une classe combinatoire, 254
transformation de Foata, 12
tri par renversements, 152

valeur, 11
vecteur, 138
vp-domaine, 138
vp-vecteur, 138

Bibliographie

[AA05] Michael H. Albert et Mike D. Atkinson : Simple permutations and pattern restricted permutations. *Discrete Mathematics*, 300(1-3):1–15, 2005.

[AAA+03] Michael H. Albert, Robert E. L. Aldred, Mike D. Atkinson, Hans P. van Ditmarsch, B. D. Handley, Chris C. Handley et Jaroslav Opatrny : Longest subsequences in permutations. *Australasian J. Combinatorics*, 28:225–238, 2003.

[AAA+07] Michael H. Albert, Robert E. L. Aldred, Mike D. Atkinson, Hans P. van Ditmarsch, Chris C. Handley, Derek A. Hotlon et Dennis J. McCaughan : Compositions of pattern restricted sets of permutations. *Australasian J. Combinatorics*, 37:43–56, 2007.

[AAAH01] Michael H. Albert, Robert E. L. Aldred, Mike D. Atkinson et Derek A. Holton : Algorithms for pattern involvement in permutations. *In ISAAC '01 : Proceedings of the 12th International Symposium on Algorithms and Computation*, volume 2223 de *Lecture Notes in Computer Science*, pages 355–366, London, UK, 2001. Springer-Verlag.

[AAB07] Michael H. Albert, Mike D. Atkinson et Robert Brignall : Permutation classes of polynomial growth. *Annals of Combinatorics*, 11(3-4):249–264, dec 2007.

[AAK03] Michael H. Albert, Mike D. Atkinson et Martin Klazar : The enumeration of simple permutations. *Journal of Integer Sequences*, 6, 2003.

[AC75] Alfred V. Aho et Margaret J. Corasick : Efficient string matching : An aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, June 1975.

[AER+06] Michael H. Albert, Murray Elder, Andrew Rechnitzer, P. Westcott et Mike Zabrocki : On the Stanley-Wilf limit of 4231-avoiding permutations and a conjecture of Arratia. *Advances in Applied Mathematics*, 36:96–105, 2006.

[ALR05] Michael H. Albert, Steve Linton et Nikola Ruškuc : The insertion encoding of permutations. *Electron. J. Combin.*, 12:Research Paper 47, 31 pp. (electronic), 2005.

[AMR02] Mike D. Atkinson, Max Murphy et Nikola Ruškuc : Partially well-ordered closed sets of permutations. *Order*, 2(19):101–113, 2002.

[AN81] David Avis et Monroe Newborn : On pop-stacks in series. *Utilitas Mathematica*, 19:129–140, 1981.

[AR08] Shlomo Ahal et Yuri Rabinovich : On the complexity of the subpattern problem. *SIAM J. Discrete Math.*, 22(2):629– 649, 2008.

[Arr99] Richard Arratia : On the Stanley–Wilf conjecture for the number of permutations avoiding a given pattern. *Electron. J. Combin.*, 6, 1999. Note, N1 4 pages (électronique).

[ARS] Mike D. Atkinson, Nikola Ruškuc et Rebecca Smith : Wreath-closed pattern classes. en préparation.

[AS02] Mike D. Atkinson et Timothy Stitt : Restricted permutations and the wreath product. *Discrete Mathematics*, 259(1-3):19–36, 2002.

[Atk] Mike D. Atkinson : Simple permutations and substitution closures. Exposé invité à la conférence Permutation Patterns 2007, disponible à `http://www-circa.mcs.st-and.ac.uk/PermutationPatterns2007/talks/atkinson.pdf`.

[Atk99] Mike D. Atkinson : Restricted permutations. *Discrete Math.*, 195(1-3):27–38, 1999.

[Bai96] D. F. Bailey : Counting arrangements of 1's and -1's. *Math. Mag.*, 69(2):128–131, 1996.

[Bax82] R. J. Baxter : *Exactly solved models in statistical mechanics*. Academic Press Inc., 1982.

[BBCP07] Sèverine Bérard, Anne Bergeron, Cédric Chauve et Christophe Paul : Perfect sorting by reversals is not always difficult. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 4(1):4–16, 2007.

[BBF07] Antonio Bernini, Mathilde Bouvel et Luca Ferrari : Some statistics on permutations avoiding generalized patterns. *Pure Mathematics and Applications*, 18(3-4), 2007.

[BBL98] Prosenjit Bose, Jonathan F. Buss et Anna Lubiw : Pattern matching for permutations. *Information Processing Letters*, 65:277–283, 1998.

[BBPR] Frédérique Bassino, Mathilde Bouvel, Adeline Pierrot et Dominique Rossin : A polynomial algorithm for deciding the finiteness of simple permutations contained in permutation classes. en préparation.

[BBR09] Frédérique Bassino, Mathilde Bouvel et Dominique Rossin : Enumeration of pin-permutations. *HAL 00348664*, 2009.

[BCC+08] Sèverine Bérard, Annie Chateau, Cédric Chauve, Christophe Paul et Eric Tannier : Perfect DCJ rearrangement. *In RECOMB-CG '08 : Proceedings of the international workshop on Comparative Genomics*, volume 5267 de *Lecture Notes in Computer Science*, pages 158–169, Berlin, Heidelberg, 2008. Springer-Verlag.

[BCM+] Matthias Bernt, Ming-Chiang Chen, Daniel Merkle, Hung-Lung Wang, Kun-Mao Chao et Martin Middendorf : Finding all sorting tandem duplication random loss operations. *In Combinatorial Pattern Matching 2009*, volume 5577 de *Lecture Notes in Computer Science*, pages 301–313. Springer.

[BCMR] Mathilde Bouvel, Cédric Chauve, Marni Mishna et Dominique Rossin : Average-case analysis of perfect sorting by reversals. *In CPM '09 : Proceedings of the 20th annual symposium on Combinatorial Pattern Matching*.

[BCMR05] Anne Bergeron, Cédric Chauve, Fabien de Montgolfier et Mathieu Raffinot : Computing common intervals of K permutations, with applications to modular decomposition of graphs. pages 779–790. ESA, 2005.

[BCMR08] Anne Bergeron, Cédric Chauve, Fabien de Montgolfier et Mathieu Raffinot : Computing commons interval of K permutations, with applications to modular decomposition of graphs. *SIAM J. Discrete Math.*, 22(3):1022–1039, 2008.

[BCP08] Sèverine Bérard, Cédric Chauve et Christophe Paul : A more efficient algorithm for perfect sorting by reversals. *Information processing letters*, 106(3):90–95, 2008.

[BDLPP99] Elena Barcucci, Alberto Del Lungo, Elisa Pergola et Renzo Pinzani : ECO : A methodology for the enumeration of combinatorial objects. *J. Difference Equ. Appl.*, 5:435–490, 1999.

[BDPR07] Antonio Bernini, Filippo Disanto, Renzo Pinzani et Simone Rinaldi : Permutations defining convex permutominoes. *Journal of Integer Sequences*, 10, 2007.

[Ber05] Anne Bergeron : A very elementary presentation of the hannenhalli-pevzner theory. *Discrete Applied Mathematics*, 146(2):134–145, 2005.

[BESV] Robert Brignall, Shalosh B. Ekhad, Rebecca Smith et Vincent Vatter : Almost avoiding permutations. À paraître dans *Discrete Mathematics*.

[BFP05] Antonio Bernini, Luca Ferrari et Renzo Pinzani : Enumerating permutations avoiding three Babson-Steingrímsson patterns. *Annals of Combinatorics*, 9:137–162, 2005.

[BGK] Mathilde Bouvel, Vladimir Grebinski et Gregory Kucherov : Combinatorial search on graphs motivated by bioinformatics applications : A brief survey. *In Graph-theoretic concepts in computer science : 31st international workshop, WG 2005*.

[BHV08a] Robert Brignall, Sophie Huczynska et Vincent Vatter : Decomposing simple permutations, with enumerative consequences. *Combinatorica*, 28(4):385–400, 2008.

[BHV08b] Robert Brignall, Sophie Huczynska et Vincent Vatter : Simple permutations and algebraic generating functions. *J. Combin. Theory Ser. A*, 115(3):423–441, 2008.

[Bil05] Philip Bille : A survey on tree edit distance and related problems. *Theoretical Computer Science*, 1-3(337):217–239, 2005.

[BLRS08] Paolo Boldi, Violetta Lonati, Roberto Radicioni et Massimo Santini : The number of convex permutominoes. *Information and Computation*, 206:1074–1083, 2008.

[BLS99] Andreas Brandstädt, Van Bang Le et Jeremy P. Spinrad : *Graph classes : a survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.

[BM83] Hermann Buer et Rolf H. Möhring : A fast algorithm for the decomposition of graphs and posets. *Mathematics of Operations Research*, 8(2):170–184, May 1983.

[BM02] Mireille Bousquet-Mélou : Four classes of pattern-avoiding permutations under one roof : Generating trees with two labels. *Electr. J. Comb.*, on(2), 2002.

[BMMR] Mireille Bousquet-Mélou, Roberto Mantaci et Fanja Rakotondrajao : Polynomial classes of permutations avoiding two patterns. Permutation Patterns 2009.

[BMS05] Anne Bergeron, Julia Mixtacki et Jens Stoye : *Mathematics of Evolution and Phylogeny*, chapitre The inversion distance problem. Oxford University Press, 2005.

[Bón97] Miklós Bóna : Permutations avoiding certain patterns : The case of length 4 and some generalizations. *Discrete Mathematics*, 175:55–67, 1997.

[Bón99] Miklós Bóna : The solution of a conjecture of Stanley and Wilf for all layered patterns. *J. Comb. Theory Ser. A*, 85(1):96–104, 1999.

[Bón04a] Miklós Bóna : *Combinatorics of Permutations*. CRC Press, Inc., Boca Raton, FL, USA, 2004.

[Bón04b] Miklós Bóna : Sharper estimates for the number of permutations avoiding a layered or decomposable pattern. *In Formal Power Series and Algebraic Combinatorics*, 2004.

[Bou06] Mathilde Bouvel : Autour des permutations séparables. Mémoire de D.E.A., École Normale Supérieure de Cachan, Master MPRI, 2006.

[BP02] Guillaume Bourque et Pavel A. Pevzner : Genome-scale evolution : reconstructing gene orders in the ancestral species. *Genome Res.*, 12:26–36, 2002.

[BP08] Mathilde Bouvel et Elisa Pergola : Posets and permutations in the duplication-loss model : Minimal permutations with d descents. *CoRR*, abs/0806.1494, 2008.

[BR06] Mathilde Bouvel et Dominique Rossin : The longest common pattern problem for two permutations. *Pure Mathematics and Applications*, 17(1-2):55–69, 2006.

[BR09] Mathilde Bouvel et Dominique Rossin : A variant of the tandem duplication - random loss model of genome rearrangement. *Theoretical Computer Science*, 410:847–858, 2009.

[Bria] Robert Brignall : Grid classes and partial well order. Pré-publication, disponible sur la page web de l'auteur.

[Brib] Robert Brignall : A survey of simple permutations. Accepté.

[Bri07] Robert Brignall : Wreath products of permutation classes. *Electron. J. Combin.*, 14(1), 2007.

[BRV07] Mathilde Bouvel, Dominique Rossin et Stéphane Vialette : Longest common separable pattern among permutations. *In CPM '07 : Proceedings of the 18th annual symposium on Combinatorial Pattern Matching*, volume 4580 de *Lecture Notes in Computer Science*, pages 316–327, Berlin, Heidelberg, 2007. Springer-Verlag.

[BRV08] Robert Brignall, Nikola Ruškuc et Vincent Vatter : Simple permutations : decidability and unavoidable substructures. *Theoret. Comput. Sci.*, 391(1-2):150–163, 2008.

[BS00a] Eric Babson et Einar Steingrímsson : Generalized permutation patterns and a classification of the Mahonian statistics. *Séminaire Lotharingien de Combinatoire*, 2000.

[BS00b] Miklós Bóna et Daniel A. Spielman : An infinite antichain of permutations. *Electr. J. Comb*, 7, N2, 2000.

[BV] Jean-Luc Baril et Rémi Vernay : Whole mirror duplication-random loss model and pattern avoiding permutations. *In Permutation patterns 2009.*

[BXHP05] Binh-Minh Bui Xuan, Michel Habib et Christophe Paul : Revisiting T. Uno and M. Yagiura's algorithm. *In Lecture notes in computer science*, volume 3827, pages 146–155. ISAAC, Springer, 2005.

[CCMR06] Kamalika Chaudhuri, Kevin Chen, Radu Mihaescu et Satish Rao : On the tandem duplication-random loss model of genome rearrangement. *In SODA '06 : Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 564–570, New York, NY, USA, 2006. ACM.

[CDM99] Sinisa Crvenkovic, Igor Dolinka et Petar Markovic : A survey of algebra of tournaments. *Novi Sad Journal of Mathematics*, 29(2):95–130, 1999.

[CH94] Alain Cournier et Michel Habib : A new linear algorithm for modular decomposition. *In* Sophie Tison, éditeur : *19th International Colloquium Trees in Algebra and Programming - CAAP'94*, volume 787 de *Lecture Notes in Computer Science*, pages 68–84. Springer, 1994.

[CHL01] Maxime Crochemore, Christophe Hancart et Thierry Lecroq : *Algorithmique du texte*. Vuibert, 2001. 347 pages.

[CHM02] Christian Capelle, Michel Habib et Fabien de Montgolfier : Graph decomposition and factorising permutations. *Discrete Mathematics and Theoretical Computer Sciences*, 5(1), 2002.

[Cib09] Josef Cibulka : On constants in the Füredi–Hajnal and the Stanley–Wilf conjecture. *J. Comb. Theory Ser. A*, 116(2):290–302, 2009.

[CK08] Anders Claesson et Sergey Kitaev : Classification of bijections between 321- and 132-avoiding permutations. *Séminaire Lotharingien de Combinatoire*, 60, 2008.

[CKS08] Anders Claesson, Sergey Kitaev et Einar Steingrímsson : Decompositions and statistics for $\beta(1,0)$-trees and nonseparable permutations. *Advances in Applied Mathematics*, 42:313–328, 2008.

[CL04] M. C. Chen et Richard C. T. Lee : Sorting by transpositions based on the first increasing substring concept. *In BIBE '04 : Proceedings of the 4th IEEE Symposium on Bioinformatics and Bioengineering*, page 553, Washington, DC, USA, 2004. IEEE Computer Society.

[Cla01] Anders Claesson : Generalised pattern avoidance. *European J. Combin.*, 22:961–971, 2001.

[CLP06] Sylvie Corteel, Guy Louchard et Robin Pemantle : Common intervals in permutations. *Discrete Mathematics and Theoretical Computer Science*, 8(1):189–214, 2006.

[CM92] Robert Cori et Antonio Machì : Maps, hypermaps and their automorphisms : a survey, i, ii, iii. *Expo. Math. 10*, pages 403–427, 429–447, 449–467, 1992.

[CM05] Anders Claesson et Toufik Mansour : Enumerating permutations avoiding a pair of Babson-Steingrímsson patterns. *Ars Combin.*, 77:17–31, 2005.

[Com74] Louis Comtet : *Advanced Combinatorics*. Dordrecht, 1974.

[CPS85] Derek G. Corneil, Yehoshua Perl et Lorna K. Stewart : A linear recognition algorithm for cographs. *SIAM Journal on Computing*, 14(4):926–934, 1985.

[CW92] Maw-Shang Chang et Fu-Hsing Wang : Efficient algorithms for the maximum weight clique and maximum weight independent set problems on permutation graphs. *Information Processing Letters*, 43(6):293–295, 1992.

[CW07] Sylvie Corteel et Lauren K. Williams : Tableaux combinatorics for the asymmetric exclusion process. *Advances in applied mathematics*, 39(3):293–310, 2007.

[Dam91] Peter Damaschke : Induced subgraph isomorphism for cographs is np-complete. *In WG '90 : Proceedings of the 16th international workshop on Graph-theoretic concepts in computer science*, pages 72–78, New York, NY, USA, 1991. Springer-Verlag New York, Inc.

[Den] Alain Denise : Structures aléatoires, modèles, et analyse des génomes. Habilitation à diriger des recherches, Université de Paris-Sud, France, 2001.

[DST07] Yoan Diekmann, Marie-France Sagot et Eric Tannier : Evolution under reversals : Parsimony and conservation of common intervals. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, 4(2):301–109, 2007.

[DT03] Serge Dulucq et Hélène Touzet : Analysis of tree edit distance algorithms. *In Proceedings of the 14th annual symposium on Combinatorial Pattern Matching (CPM), Lecture Notes in Computer Science*, 2676:83–95, 2003.

[Edm60] Jack Edmonds : A combinatorial representation of polyhedral surfaces. *Notices Am. Math. Soc.*, 7:643, 1960.

[EELW07] Henrik Eriksson, Kimmo Eriksson, Svante Linusson et Johan Wästlund : Dense packing of patterns in a permutation. *Annals of Combinatorics*, 11(3-4):459–470, 2007.

[EHtPR98] Andrzej Ehrenfeucht, Tero Harju, P. ten Pas et Grzegorz Rozenberg : Permutations, parenthesis words, and Schröder numbers. *Discrete Mathematics*, 190:259–264, 1998.

[Eli04] Sergi Elizalde : *Statistics on pattern-avoiding permutations*. Thèse de doctorat, MIT, 2004.

[EMS03] Nadia El-Mabrouk et David Sankoff : The reconstruction of doubled genomes. *SIAM J. Comput.*, 32(3):754–792, 2003.

[FFG+06] Irene Fanti, Andrea Frosini, Elisabetta Grazzini, Renzo Pinzani et Simone Rinaldi : Polyominoes determined by permutations. *Fourth Colloquium on Mathematics and Computer Science Algorithms, Trees, Combinatorics and Probabilities*, 2006.

[Fra53] Roland Fraïssé : On a decomposition of relations which generalizes the sum of ordering relations. *Bull. Amer. Math. Soc.*, 59:389, 1953.

[FS08] Philippe Flajolet et Robert Sedgewick : *Analytic Combinatorics*. Cambridge University Press, 2008.

[FV04] Martin Figeac et Jean-Stéphane Varré : Sorting by reversals with common intervals. *In Algorithms in Bioinformatics, WABI 2004*, volume 3240 de *Lecture Notes in Computer Science*, pages 26–37. Springer Berlin / Heidelberg, 2004.

[Gal67] Tibor Gallai : Transitiv orientierbare Graphen. *Acta Mathematica Academiae Scientiarum Hungaricae*, 18(1-2):25–66, 1967.

[GJ79] Michael R. Garey et David S. Johnson : *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[Gui95] Olivier Guibert : *Combinatoire des permutations à motifs exclus, en liaison avec mots, cartes planaires et tableaux de Young*. Thèse de doctorat, Université Bordeaux 1, 1995.

[GV] Sylvain Guillemot et Stéphane Vialette : Pattern matching for 321-avoiding permutations. *In ISAAC 2009*.

[Hab] Michel Habib : Notes du cours d'algorithmique des graphes du master MPRI. Notes de cours rédigées par Philippe Gambette, disponibles à l'adresse `http://philippe.gambette.free.fr/SCOL/Graphes.pdf`.

[HMP04] Michel Habib, Fabien de Montgolfier et Christophe Paul : A simple linear-time modular decomposition algorithm for graphs, using order extension. *In SWAT'04, 9th Scandinavian Workshop on Algorithm Theory*, 2004.

[HP] Michel Habib et Christophe Paul : A survey on algorithmic aspects of modular decomposition. en préparation.

[HP99] Sridhar Hannenhalli et Pavel A. Pevzner : Transforming cabbage into turnip : polynomial algorithm for sorting signed permutations by reversals. *J. ACM*, 46(1):1–27, 1999.

[HS01] Steffen Heber et Jens Stoye : Finding all common intervals of k permutations. *In 12th Annual Symposium Combinatorial Pattern Matching, (CPM 2001)*, volume 2089 de *Lecture Notes in Computer Science*, pages 207–218. Springer Verlag, 2001.

[HV06] Sophie Huczynska et Vincent Vatter : Grid classes and the Fibonacci dichotomy for restricted permutations. *The Electronic Journal of Combinatorics*, 13(1), 2006.

[Iba97] Louis Ibarra : Finding pattern matchings for permutations. *Information Processing Letters*, 61(6):293–295, 1997.

[Ill97] Pierre Ille : Indecomposable graphs. *Discrete Mathematics*, 173:71–78, 1997.

[Kan92] Viggo Kann : On the approximability of the maximum common subgraph problem. *In STACS '92 : Proceedings of the 9th Annual Symposium on Theoretical Aspects of Computer Science*, pages 377–388, London, UK, 1992. Springer-Verlag.

[Kla00] Martin Klazar : The Füredi-Hajnal conjecture implies the Stanley-Wilf conjecture. *In Formal Power Series and Algebraic Combinatorics, Moscow 2000*, pages 250–255. Springer, 2000.

[Kle95] Philip N. Klein : Computing the edit-distance between unrooted ordered trees. *In ESA '98 : Proceedings of the 6th Annual European Symposium on Algorithms*, volume 1461 de *Lecture Notes In Computer Science*, pages 91–102. Springer-Verlag, 1995.

[KM03] Sergey Kitaev et Toufik Mansour : A survey on certain pattern problems. Preprint available at http ://www.ru.is/kennarar/sergey/publications.html, 2003.

[Knu73] Donald E. Knuth : *Fundamental Algorithms*, volume 1 de *The Art of Computer Programming*. Addison-Wesley, Reading MA, 3-ème édition, 1973.

[Lab04] Anthony Labarre : Réarrangement de génomes, tri par inversions et distances entre permutations. Mémoire de D.E.A., Université Libre de Bruxelles, Brussels, Belgium, JUN 2004.

[Lab05] Anthony Labarre : A new tight upper bound on the transposition distance. *In Proc. Fifth Workshop Algorithms in Bioinformatics, R. Casadio and*, volume 3692 de *Lecture Notes in Computer Science*, pages 216–227. Springer, 2005.

[Lab06] Anthony Labarre : New bounds and tractable instances for the transposition distance. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 3(4):380–394, 2006.

[LEMTS03] Jean-François Lefebvre, Nadia El-Mabrouk, Elisabeth R. M. Tillier et David Sankoff : Detection and validation of single gene inversions. *Bioinformatics*, 19(1): 190–196, 2003.

[Mac16] Percy A. MacMahon : *Combinatory Analysis.* Cambridge University Press, 1915-1916.

[Mie] Grégory Miermont : Cartes aléatoires. Notes du cours donné à la fédération de recherche en mathématiques de Paris centre, disponibles à `http://www.math.u-psud.fr/~miermont/courscartes.pdf`.

[Mon03] Fabien de Montgolfier : *Décomposition modulaire des graphes. Théorie, extensions et algorithmes.* Thèse de doctorat, Université Montpellier II, 2003.

[MP01] Frederic Maffray et Myriam Preissmann : A translation of Tibor Gallai's paper : Transitiv orientierbare Graphen. *In* J. L. Ramirez-Alfonsin et B. A. Reed, éditeurs : *Perfect graphs*, pages 25–66. J. Wiley, 2001.

[MR84] Rolf H. Möhring et Franz J. Radermacher : Substitution decomposition for discrete structures and connections with combinatorial optimization. *Annals of Discrete Math*, 19:257–356, 1984.

[MR06] Anne Micheli et Dominique Rossin : Edit distance between unlabeled ordered trees. *RAIRO - Theoretical Informatics and Applications*, 40:593–609, 2006.

[MT04] Adam Marcus et Gábor Tardos : Excluded permutation matrices and the Stanley-Wilf conjecture. *J. Comb. Theory, Ser. A*, 107(1):153–160, 2004.

[Noo96] John Noonan : The number of permutations containing exactly one increasing subsequence of length three. *Discrete Math.*, 152(1–3):307–313, 1996.

[Nov99] Jean-Christophe Novelli : *Combinatoire des tableaux et des rubans.* Thèse de doctorat, Université Paris 7, 1999.

[Pau] Christophe Paul : Aspects algorithmiques de la décomposition modulaire. Habilitation à diriger des recherches, Université de Montpellier 2, France, 2006.

[Piv08] Carine Pivoteau : *Génération aléatoire de structures combinatoires : méthode de Boltzmann effective.* Thèse de doctorat, Université Paris VI – Pierre et Marie Curie, 2008.

[PR09] Christophe Prieur et Stéphane Raux : Liens proches dans les réseaux sociaux : la dynamique des commentaires de flickr. *HAL : inria-00384644*, 2009.

[Pud] Lara Pudwell : Enumeration schemes for permutations avoiding barred patterns. Soumis.

[Reg81] Amitai Regev : Asymptotic values for degrees associated with strips of young diagrams. *Adv. Math.*, 41(2):115–136, 1981.

[Sch61] C. Schensted : Longest increasing and decreasing subsequences. *Canadian Journal of Mathematics*, 1961.

[Sch98] Gilles Schaeffer : *Conjugaison d'arbres et cartes combinatoires aléatoires.* Thèse de doctorat, Université Bordeaux 1, 1998.

[Slo07] Neil J. A. Sloane : The on-line encyclopedia of integer sequences, 2007. disponible à l'adresse `www.research.att.com/~njas/sequences/`.

[Spi92] Jeremy Spinrad : P4-trees and substitution decomposition. *Discrete Applied Mathematics*, 39(3):263–291, 1992.

[SS85] R. Simion et F. Schmidt : Restricted permutations. *European Journal of Combinatorics*, 6:383–406, 1985.

[ST93] James H. Schmerl et William T. Trotter : Critically indecomposable partially ordered sets, graphs, tournaments and other binary relational structures. *Discrete Mathematics*, 113:191–205, 1993.

[Sta97] Richard P. Stanley : *Enumerative Combinatorics*, volume 1. Cambridge University Press, 1997.

[Sum71] David P. Sumner : *Indecomposable graphs*. Thèse de doctorat, Univ. of Massachusetts, 1971.

[TBS07] Eric Tannier, Anne Bergeron et Marie-France Sagot : Advances on sorting by reversals. *Discrete Appl. Math.*, 155:881–888, 2007.

[TCHP08] Marc Tedder, Derek G. Corneil, Michel Habib et Christophe Paul : Simple, linear-time modular decomposition. *In International Colloquium on Automata, Languages and Programming - ICALP*, numéro 5125 de Lecture Notes in Computer Science, pages 634–645, 2008.

[UY00] Takeaki Uno et Mutsunori Yagiura : Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 14:209–227, 2000.

[Val01] Gabriel Valiente : Subtree isomorphism and related problems. Teaching at Advanced Graph Algorithms, Riga, 2001.

[Vat03] Vincent Vatter : Permutations avoiding two patterns of length three. *Electron. J. Combin*, 9(2), 2003.

[Vat08] Vincent Vatter : Enumeration schemes for restricted permutations. *Comb. Probab. Comput.*, 17(1):137–159, 2008.

[Wes90] Julian West : *Permutations with forbidden subsequences and stack-sortable permutations*. Thèse de doctorat, MIT, 1990.

[Wes95] Julian West : Generating trees and the Catalan and Schröder numbers. *Discrete Mathematics*, 146:247–262, 1995.

[Zei98] Doron Zeilberger : Enumeration schemes and, more importantly, their automatic generation. *Annals of Combinatorics*, 2:185–195, 1998.

[ZS89] Kaizhong Zhang et Dennis Shasha : Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262, 1989.

[ZWS95] Kaizhong Zhang, Jason T. L. Wang et Dennis Shasha : On the editing distance between undirected acyclic graphs and related problems. *In International Journal of Foundations of Computer Science*, pages 395–407. Springer-Verlag, 1995.