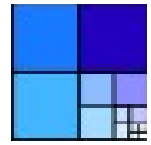
Motifs et classes de permutations : le point de vue des arbres de décomposition

Mathilde Bouvel

Institut Gaspard Monge, Séminaire Algo, 17 nov. 2009



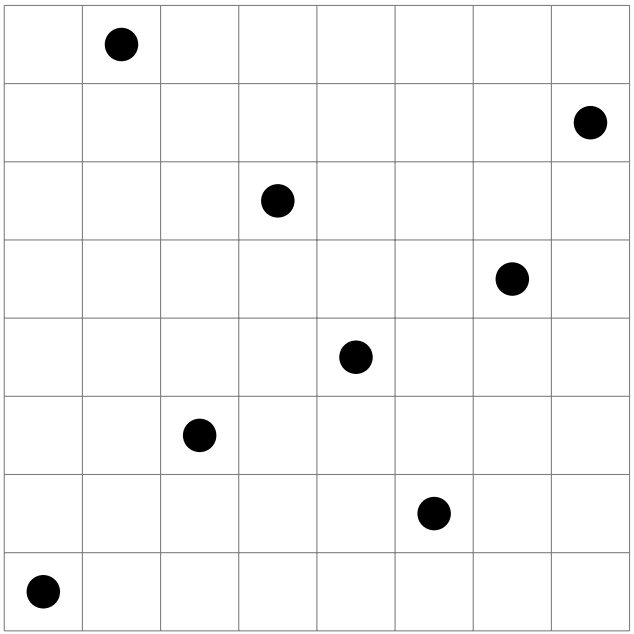
LIAFA

Plan de l'exposé

Représentation des permutations

Permutation : Bijection de $[1..n]$ dans lui-même. Ensemble S_n.

- Représentation linéaire :
  $\sigma = 1\ 8\ 3\ 6\ 4\ 2\ 5\ 7$
- Représentation sur deux lignes :
  $$\sigma = \begin{pmatrix} 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8 \\ 1\ 8\ 3\ 6\ 4\ 2\ 5\ 7 \end{pmatrix}$$
- Représentation en produit de cycles :
  $\sigma = (1)\ (2\ 8\ 7\ 5\ 4\ 6)\ (3)$
- Représentation graphique :



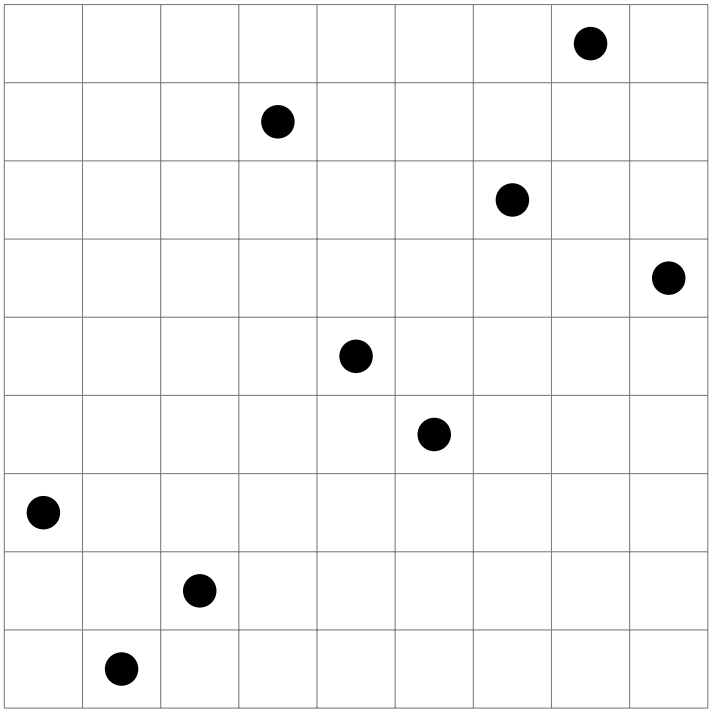
Motif dans les permutations

Relation (d'ordre) de motif $\preccurlyeq$:
$\pi \in S_k$ est un motif de $\sigma \in S_n$ si $\exists\ 1 \leq i_1 < \ldots < i_k \leq n$ tel que $\sigma_{i_1} \ldots \sigma_{i_k}$ est isomorphe en ordre ($\equiv$) à π.

Notation : $\pi \preccurlyeq \sigma$.

De manière équivalente :
La normalisation de $\sigma_{i_1} \ldots \sigma_{i_k}$ sur $[1..k]$ donne π.

Exemple : $1\,2\,3\,4 \preccurlyeq 3\,\mathbf{1}\,\mathbf{2}\,8\,\mathbf{5}\,4\,\mathbf{7}\,9\,6$
car $1\,2\,5\,7 \equiv 1\,2\,3\,4$.



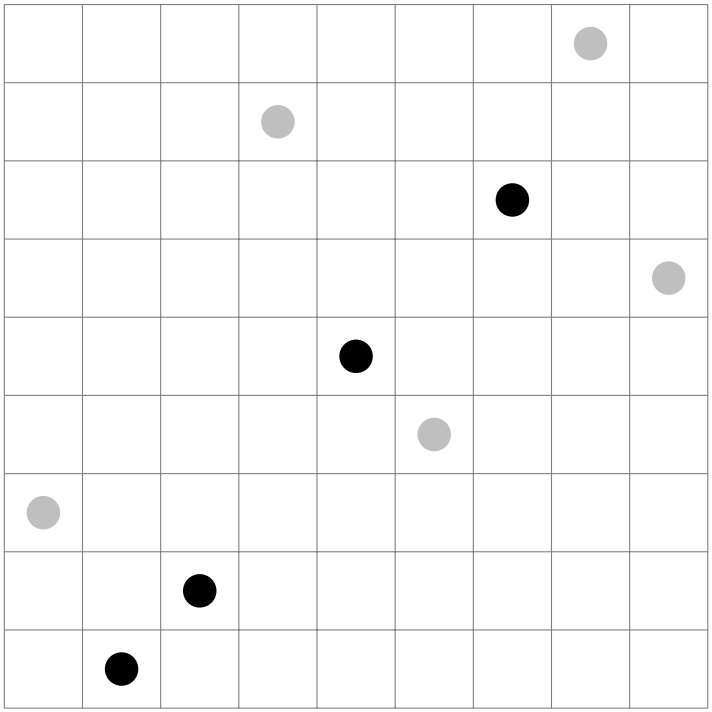
Motif dans les permutations

Relation (d'ordre) de motif $\preccurlyeq$:
$\pi \in S_k$ est un motif de $\sigma \in S_n$ si $\exists\ 1 \leq i_1 < \ldots < i_k \leq n$ tel que $\sigma_{i_1} \ldots \sigma_{i_k}$ est isomorphe en ordre ($\equiv$) à π.

Notation : $\pi \preccurlyeq \sigma$.

De manière équivalente :
La normalisation de $\sigma_{i_1} \ldots \sigma_{i_k}$ sur $[1..k]$ donne π.

Exemple : $1\,2\,3\,4 \preccurlyeq 3\,\mathbf{1}\,\mathbf{2}\,8\,\mathbf{5}\,4\,\mathbf{7}\,9\,6$
car $1\,2\,5\,7 \equiv 1\,2\,3\,4$.



Classes de permutations

Classe de permutations : ensemble fermé par le bas pour $\preccurlyeq$.
De manière équivalente : $\sigma \in \mathcal{C}$ et $\pi \preccurlyeq \sigma \Rightarrow \pi \in \mathcal{C}$

$S(B)$: la classe des permutations évitant tous les motifs de B.
Base B : lorsque B est une antichaîne pour $\preccurlyeq$.
La base B est unique.

Réciproque : Toute classe $\mathcal{C}$ peut être caractérisée par sa base :

$$\mathcal{C} = S(B) \text{ pour } B = \{\sigma \notin \mathcal{C} : \forall \pi \preccurlyeq \sigma \text{ tel que } \pi \neq \sigma, \pi \in \mathcal{C}\}$$

Une base peut être finie ou infinie.

Énumération[Stanley-Wilf, Marcus-Tardos] : $|S_n(B)| \leq c_B^n$

Problématiques

- **Combinatoire** : étude de classes définies par leur base.

↪ Depuis Knuth en 1973 avec les triables par une pile.

↪ Énumération, génération exhaustive.

- **Algorithmique** : problématiques de l'algorithmique du texte.

↪ Recherche d'occurrence, de plus grand motif commun.

↪ En lien avec le test d'appartenance à une classe.

- **Combi. (et algo.)** : étude des classes dans leur ensemble.

↪ Une classe n'est pas toujours décrite par sa base.

↪ Automatiser la détection de structure dans une classe.

Décomposition par substitution : les grands principes

Analogue de la décomposition des entiers en facteurs premiers.

- Möhring et Radermacher en 1984 : contexte général.
- Spécialisation : décomposition modulaire des graphes.

Repose sur :

- un principe de construction d'objets (permutations) à partir d'objets plus petits : la substitution.
- des "briques de base" pour cette construction : permutations **simples**, graphes **premiers**.

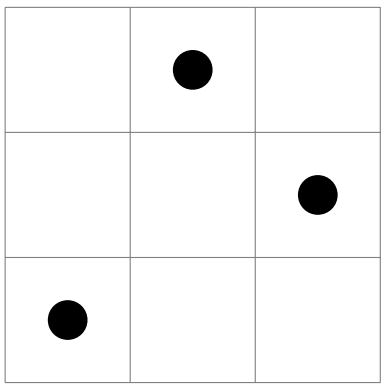
Propriétés essentielles :

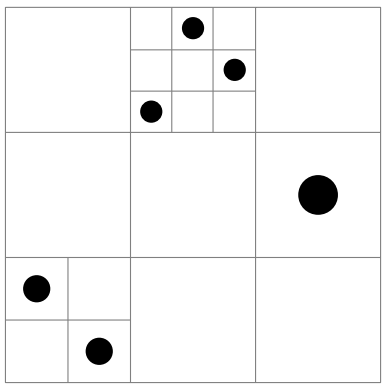
- tout objet peut être décomposé en ces "briques de base".
- cette décomposition est unique.

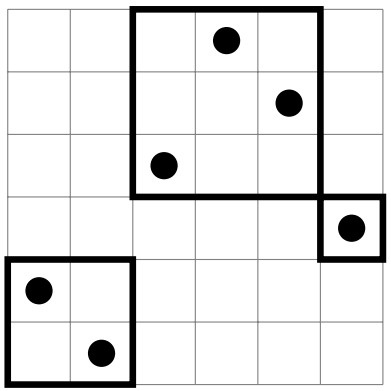
Principe de substitution sur les permutations

Substitution ou dilatation : $\sigma = \pi[\alpha^{(1)}, \alpha^{(2)}, \dots, \alpha^{(k)}]$.

Exemple : Ici, $\pi = 1\,3\,2$, et $\begin{cases} \alpha^{(1)} = 2\,1 \\ \alpha^{(2)} = 1\,3\,2 \\ \alpha^{(3)} = 1 \end{cases}$.







Donc $\sigma = 1\,3\,2[2\,1, 1\,3\,2, 1] = 2\,1\,4\,6\,5\,3$.

Permutations simples

Intervalle (ou bloc) = ensemble d'éléments de σ dont les positions **et** les valeurs sont des intervalles
Exemple : 5 7 4 6 est un intervalle de 2 5 7 4 6 1 3

Permutation simple = ne possède pas d'intervalle, sauf les intervalles triviaux : $1, 2, \ldots, n$ et σ
Exemple : 3 1 7 4 6 2 5 est simple.

Les plus petites simples : 1 2, 2 1, 2 4 1 3, 3 1 4 2

Arbres de décomposition : formalisent l'idée que les permutations simples sont les "briques de base" pour construire toutes les permutations.

Décomposition par substitution des permutations

Prop.[Albert Atkinson 2005] :
$\forall \sigma, \exists \pi$ simple unique, $\exists \alpha^{(i)}$ tels que $\sigma = \pi[\alpha^{(1)}, \dots, \alpha^{(k)}]$.
Si $|\pi| \geq 4$, les $\alpha^{(i)}$ sont uniques.
Si $\pi = 12$ (21), et $\alpha^{(1)}$ est $\oplus$ *($\ominus$) -indécomposable*, les $\alpha^{(i)}$ sont uniques.

Théorème : Chaque σ se décompose de manière unique en

- $12 \dots k[\alpha^{(1)}, \dots, \alpha^{(k)}]$, où les $\alpha^{(i)}$ sont $\oplus$-indécomposables
- $k \dots 21[\alpha^{(1)}, \dots, \alpha^{(k)}]$, où les $\alpha^{(i)}$ sont $\ominus$-indécomposables
- $\pi[\alpha^{(1)}, \dots, \alpha^{(k)}]$, où π est simple de taille ≥ 4

Remarques :

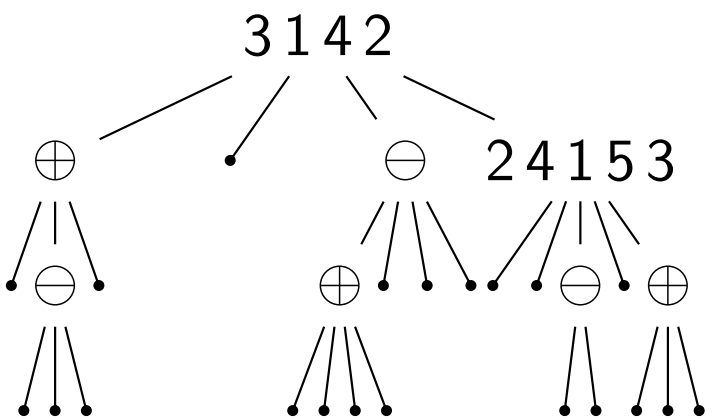
- Décomposition unique sans restriction supplémentaire.
- Les $\alpha^{(i)}$ sont les intervalles forts maximaux de σ.

Arbre de décomposition

Décomposition récursive dans les $\alpha^{(i)} \Rightarrow$ arbre de décomposition.

Exemple : Arbre de décomposition de $\sigma =$

10 13 12 11 14 1 18 19 20 21 17 16 15 4 8 3 2 9 5 6 7



Notations et propriétés :

- $\oplus = 12 \dots k$ et $\ominus = k \dots 21 =$ nœuds linéaires.
- π simple de taille $\geq 4 =$ nœud premier.
- Pas d'arête $\oplus - \oplus$ ou $\ominus - \ominus$.
- Arbres ordonnés.

- Ces conditions caractérisent les arbres de décomposition.

Bijection entre permutations et leurs arbres de décomposition.

Quelques utilisations des arbres de décomposition (1/2)

- Calcul de l'arbre de décomposition : temps linéaire. [UY00]

Algorithmique :

- Recherche de motif : *NP*-difficile. [BBL98]
- Algorithmes "efficaces" pour la recherche de motif :
  - ↪ Recherche d'occurrence d'un motif. [BBL98,Iba97]
  - ↪ Plus grand motif commun entre 2 permutations (ou pour un ensemble de permutations) [BR06,BRV07]
- Algorithmes pour la bio-informatique :
  - ↪ Tri parfait par renversement [BBCP07,BCMR09]
  - ↪ *Perfect DCJ rearrangements* [BCCPT08]

Quelques utilisations des arbres de décomposition (2/2)

Exemples en combinatoire :

- Classes fermées par produit de substitution : tous les arbres sur un ensemble de nœuds donné (fermé par le bas) [AA05]
- Classes définies par une propriété : caractérisation des arbres plutôt que des permutations
  - ↪ Permutations séparables [Iba97]
  - ↪ Permutations en épingles [BBR08]
- Les permutations simples d'une classe sont une mesure de sa complexité intrinsèque
  - ↪ Test algorithmique de propriétés combinatoires des classes [AA05,BRV08,BBPR09]

Permutations séparables

Plusieurs définitions :

- par motifs exclus : $S(2413, 3142)$
- possédant un arbre de séparation = arbre binaire dont les nœuds sont $\oplus$ ou $\ominus$

Définition indépendante des arbres de décomposition. Mais :

- permutation séparable = dont l'arbre de décomposition n'a pas de nœud premier
- arbre de séparation = arbre de décomposition "binarisé"

Utilisation des arbres pour la recherche de motif [Iba97,BBL98]

Recherche d'occurrence d'un motif séparable [BBL98]

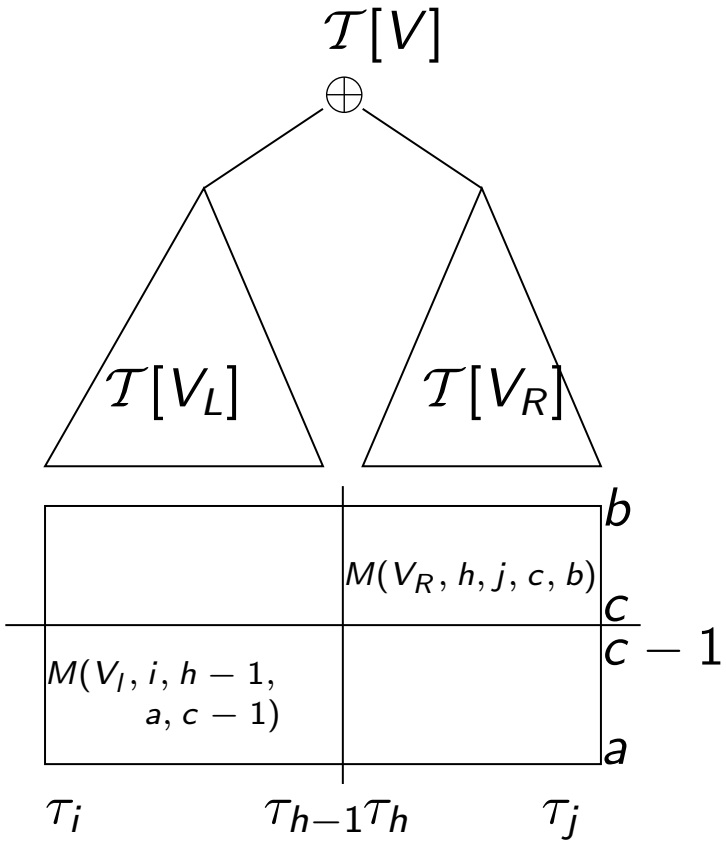
- Entrée : σ séparable (taille k), τ quelconque (taille n).
- Sortie : une occurrence de σ dans τ si elle existe.

Solution : Programmation dynamique

- En suivant le guide = arbre de séparation $\mathcal{T}$ de σ
- $\mathcal{T}[V]$ = sous-arbre de $\mathcal{T}$ enraciné en V
- $M(V, i, j, a, b)$ = une occurrence de $\mathcal{T}[V]$ dans $\tau_i \dots \tau_j$ n'utilisant que des valeurs τ_k entre a et b
- Calcul des tableaux $M(V, \_, \_, \_, \_)$ des feuilles vers la racine

Recherche d'occurrence d'un motif séparable [BBL98]

Algorithme de [BBL98]



Complexité :

[BBL98] :

- $\mathcal{O}(kn^6)$ en temps
- $\mathcal{O}(kn^4)$ en espace

[Iba97] :

- $\mathcal{O}(kn^4)$ en temps
- $\mathcal{O}(kn^3)$ en espace

$\Rightarrow$ Polynomiale

Généralisation avec les arbres de décomposition

Méthode :

- Toujours programmation dynamique.
- "Binariser" les nœuds linéaires.
- Traitement des nœuds premiers à ajouter.

Solutions apportées :

- Recherche d'occurrence d'un motif quelconque en $\mathcal{O}(kn^{2d+2})$
- Recherche de plus grand motif commun à deux permutations dont une séparable en $\mathcal{O}(\min(n_1, n_2)n_1 n_2^6)$
- Recherche de plus grand motif commun à deux permutations en $\mathcal{O}(\min(n_1, n_2)n_1 n_2^{2d_1+2})$

avec d = arité maximale d'un nœud premier

Recherche de plus motif commun restreint à une classe

Problèmes traités :

- Recherche de plus grand motif commun séparable à K permutations : $\mathcal{O}(n^{6K+1})$ et temps en $\mathcal{O}(n^{4K+1})$ en espace.
- Recherche de plus grand motif commun séparable à un ensemble de permutations (même séparables) : *NP*-difficile

Limitations de la recherche de motif dans une classe :

- Comparaison entre :
  - plus grand motif commun appartenant à $\mathcal{C}$
  - plus grand motif commun sans contrainte
- L'approximation obtenue est en $\sqrt{Opt}$ pour toute classe $\mathcal{C}$

Classes de permutations : l'ancien et le nouveau

Point de vue motifs exclus :

Définition par la base de motifs interdits

- Énumération
- Génération exhaustive

Exemples :

- $S(213, 312)$
- $S(4231)$
- $S(1\,2\ldots k)$

Structure dans les classes de permutations :

Définition par une propriété stable pour $\preceq$

- Caractérisation des permutations
  - ↪ par motifs interdits
  - ↪ par une description récursive
- Propriétés des séries génératrices
- Test d'appartenance

Exemples :

- Triables par pile $= S(231)$
- Séparables $= S(2413, 3142)$
- Permutations en épingles

Structure dans les classes de permutations

Théorème [AA05] : Si $\mathcal{C}$ contient un nombre fini de permutations simples, alors

- $\mathcal{C}$ a une base finie
- $\mathcal{C}$ a une série génératrice algébrique

Preuve : repose sur la décomposition par substitution.

Algorithmiquement :

- procédure de semi-décision
- calcul "très exponentiel" des simples dans $\mathcal{C}$

Série génératrice calculable à partir des simples dans $\mathcal{C}$

Nombre fini de permutations simples : décision

Théorème [BRV08] : On peut décider si $\mathcal{C}$ donnée par sa base finie contient un nombre fini de permutations simples.

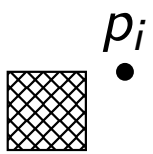
Prop. $\mathcal{C} = S(B)$ contient une infinité de simples ssi $\mathcal{C}$ contient :

1. soit une infinité de permutations parallèles
2. soit une infinité de permutations en chevrons simples
3. soit une infinité de permutations en épingles propres

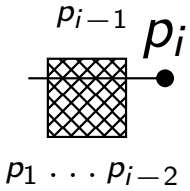
	Procédure de décision	Complexité
2. et 3. :	test d'occurrence de motifs de taille 3 ou 4 dans les $\beta \in B$.	Polynomiale
1. :	Décidabilité avec des techniques de théorie des automates	Doublement exponentielle

Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

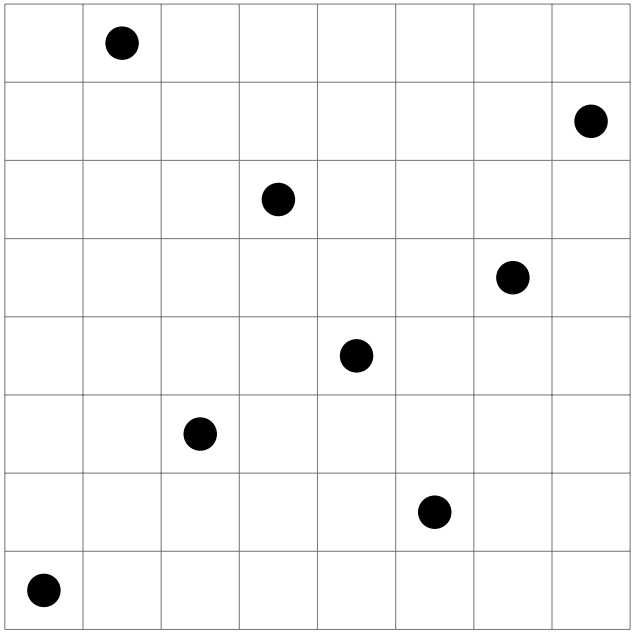
- et

- la condition de séparation 

- ou la condition d'indépendance 

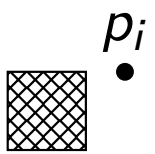
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :

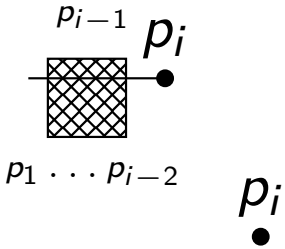


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \dots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

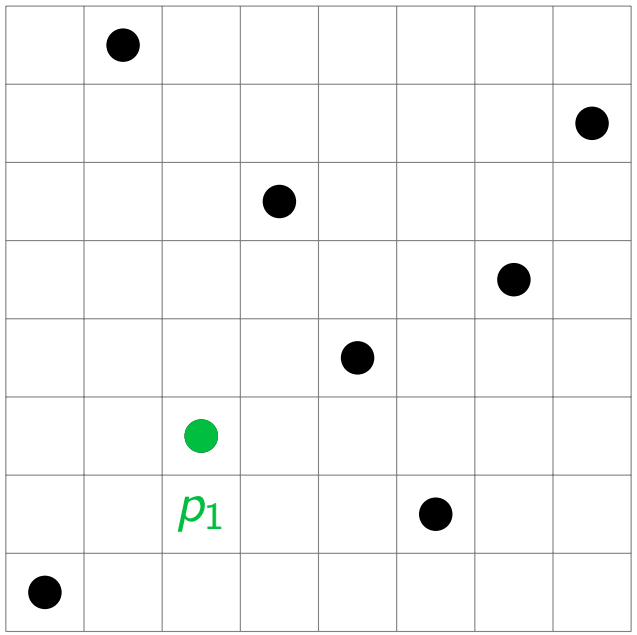
- et

- la condition de séparation 

- ou la condition d'indépendance 

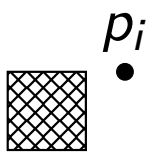
 = *boîte englobante* de $\{p_1, \dots, p_{i-1}\}$

Exemple :

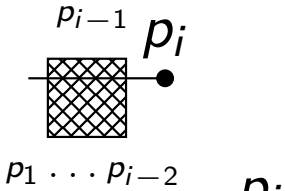


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \dots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

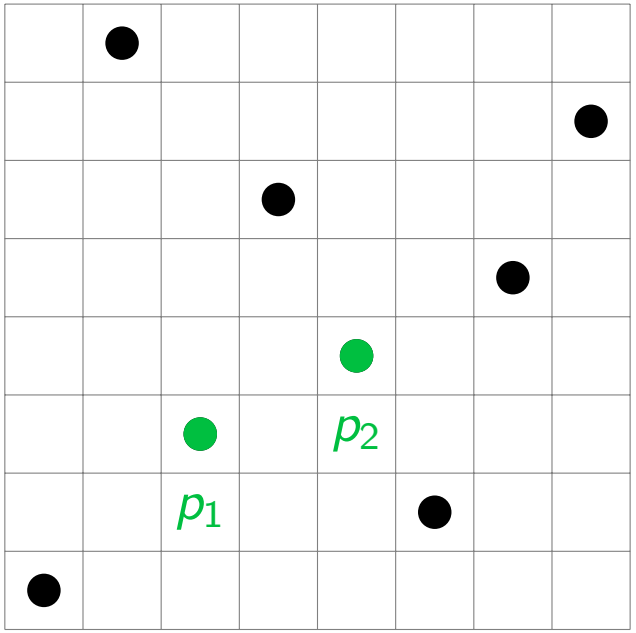
- et

- la condition de séparation 

- ou la condition d'indépendance 

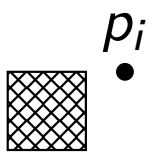
 = *boîte englobante* de $\{p_1, \dots, p_{i-1}\}$

Exemple :

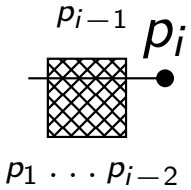


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

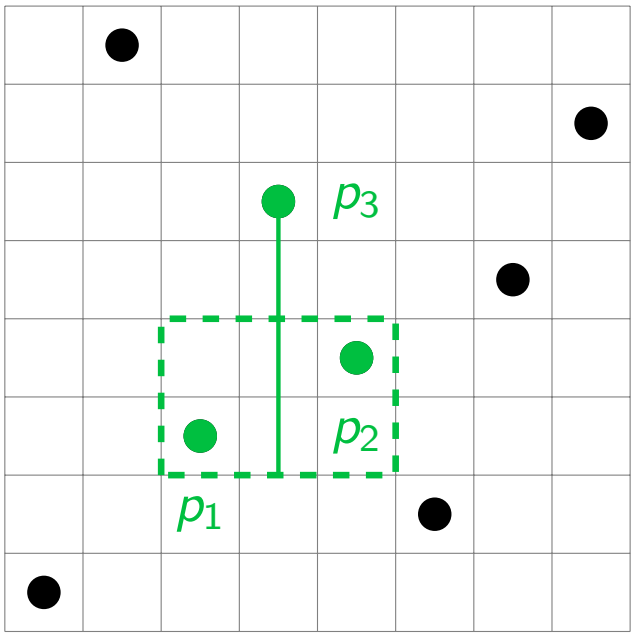
- et

- la condition de séparation 

- ou la condition d'indépendance 

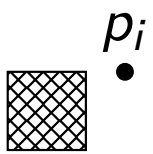
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :

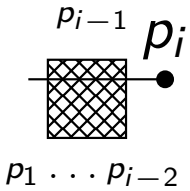


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extérioritéé 

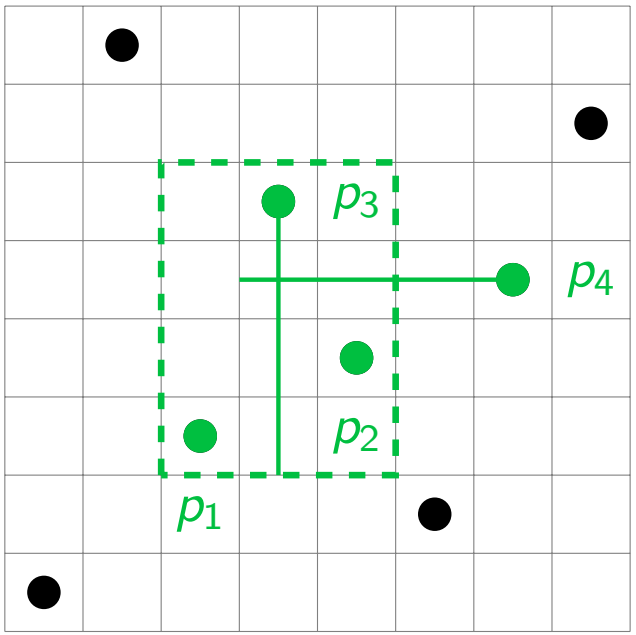
- et

- la condition de séparation 

- ou la condition d'indépendance 

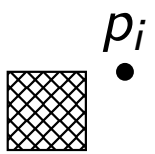
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :

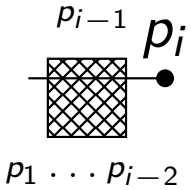


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

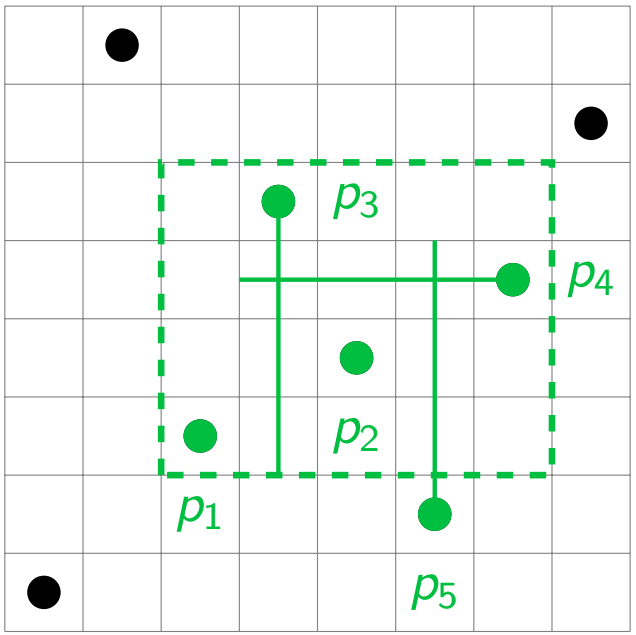
- et

- la condition de séparation 

- ou la condition d'indépendance 

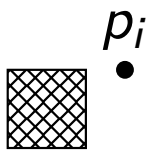
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :

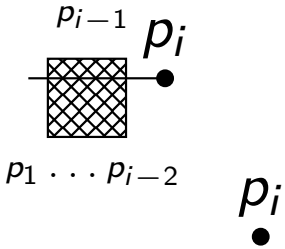


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

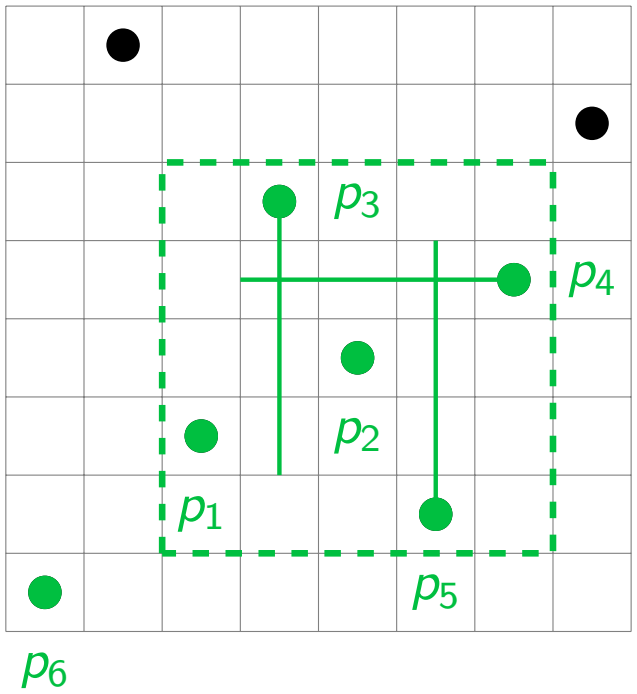
- et

- la condition de séparation 

- ou la condition d'indépendance 

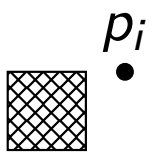
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :

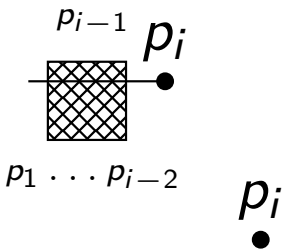


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \dots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

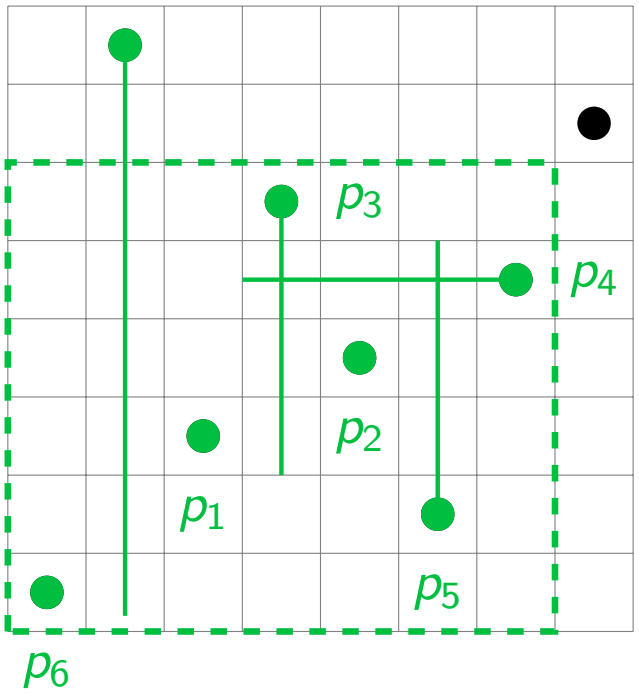
- et

- la condition de séparation 

- ou la condition d'indépendance 

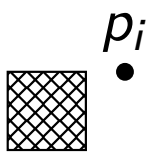
 = *boîte englobante* de $\{p_1, \dots, p_{i-1}\}$

Exemple :

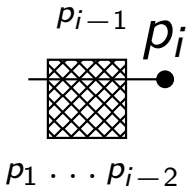


Classes des permutations en épingles

Permutation en épingles = qui admet une représentation en épingles, *i.e.* une séquence $(p_1, \ldots, p_n)$ où chaque p_i vérifie :

- la condition d'extériorité 

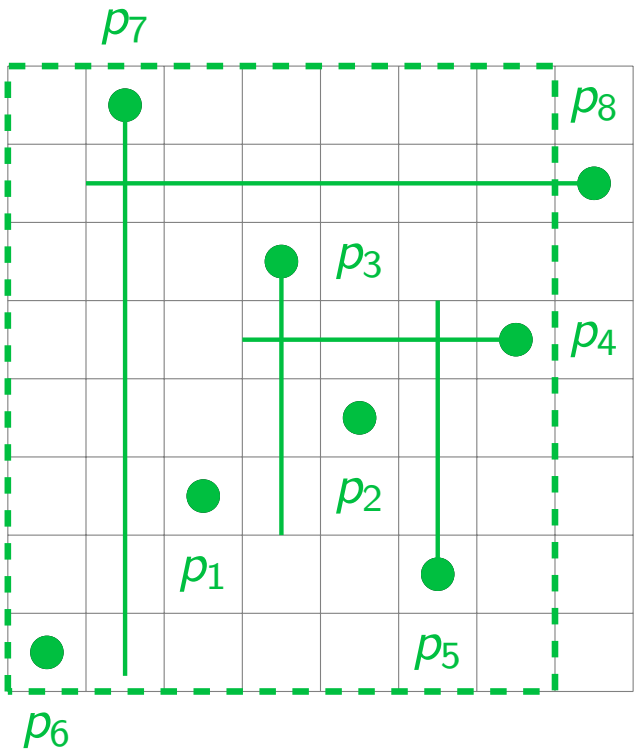
- et

- la condition de séparation 

- ou la condition d'indépendance 

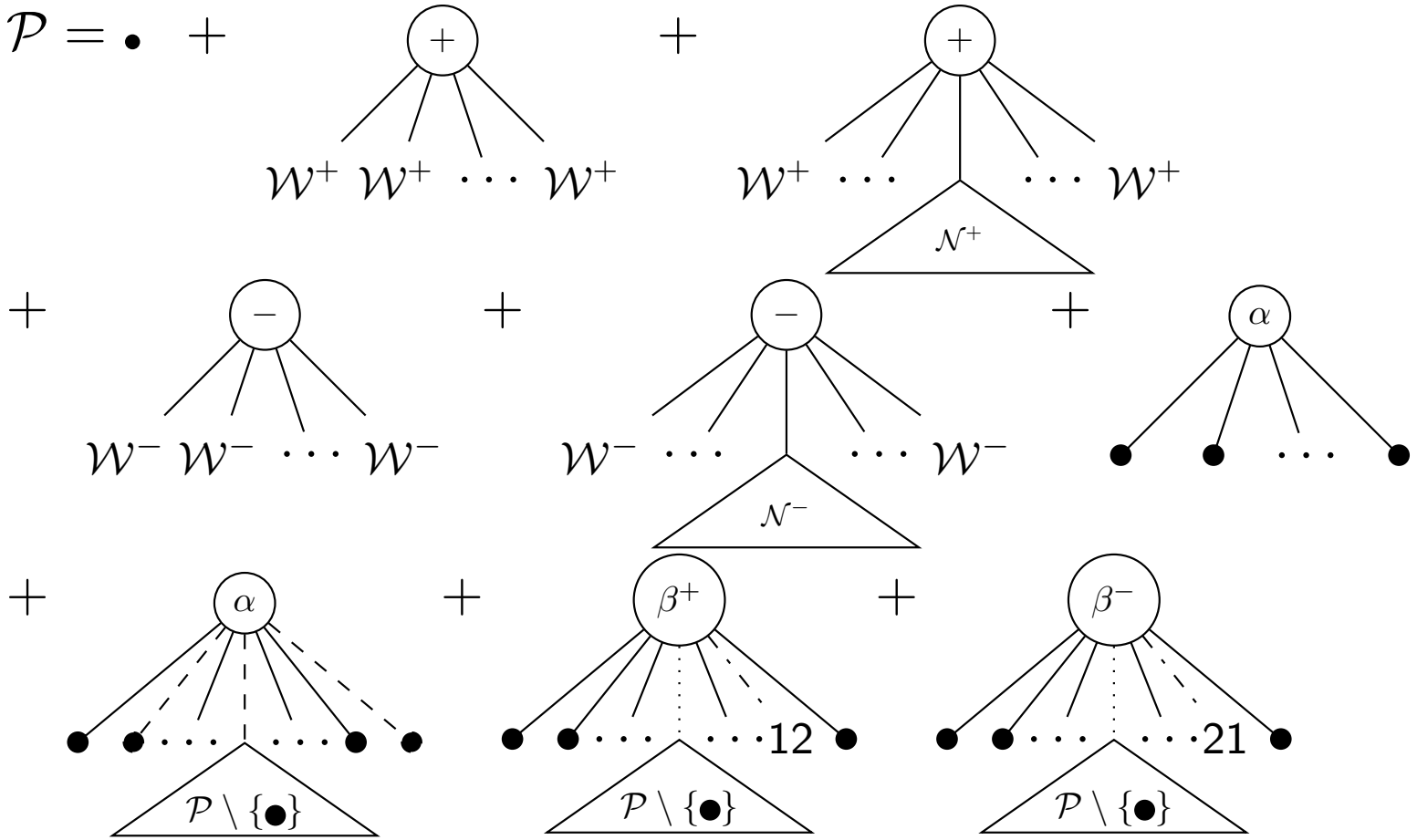
 = *boîte englobante* de $\{p_1, \ldots, p_{i-1}\}$

Exemple :



Résultats sur les permutations en épingles (1/2)

- Caractérisation des arbres de décomposition

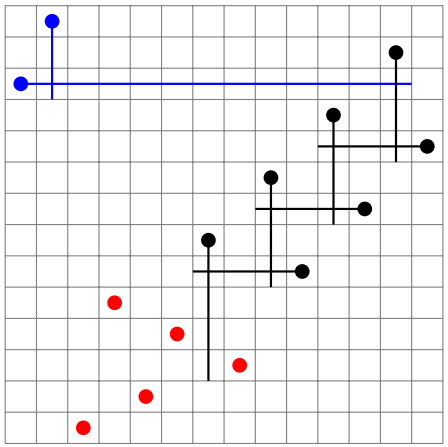


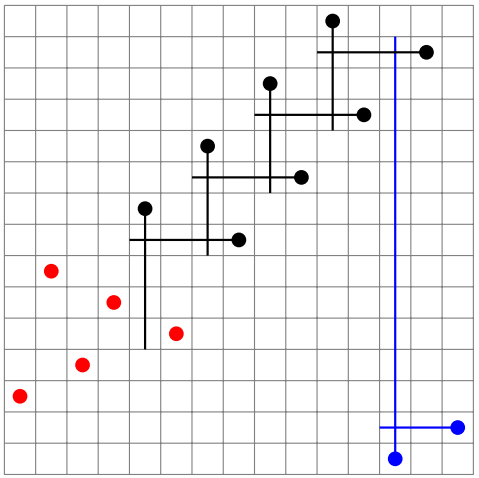
Résultats sur les permutations en épingles (2/2)

- Calcul de la série génératrice : rationnelle

$$P(z) = z\frac{8z^6-20z^5-4z^4+12z^3-9z^2+6z-1}{8z^8-20z^7+8z^6+12z^5-14z^4+26z^3-19z^2+8z-1}$$

- Base infinie (qui reste à déterminer)





- Algorithme polynomial calculant si le nombre de simples de $S(B)$ est fini, au lieu de la procédure de décision de [BRV08]

Algorithme polynomial pour le nombre fini de simples

Points communs avec [BRV08] :

- Codage par des mots d'épingles sur $\{1, 2, 3, 4, L, R, U, D\}$
- Construction d'automates

Étude des permutations en épingles $\Rightarrow$ meilleure compréhension des relations entre mots d'épingles et motifs dans les permutations

Spécificités :

- Construction polynomiale d'un automate **(déterministe, complet)** de langage $\mathcal{L}$ = mots d'épingles des permutations en épingles propres contenant un $\beta \in B$
- Ce langage est-il co-fini ? Polynomial.
- Oui ssi la classe contient un nombre fini de simples.

Calcul automatique de la série génératrice d'une classe

Chemin parcouru :

- Décider du nombre fini de simples

↪ Polynomial

- Calculer les simples dans la classe

↪ Exponentiel

- Calculer la s.g. (algébrique) à partir des simples

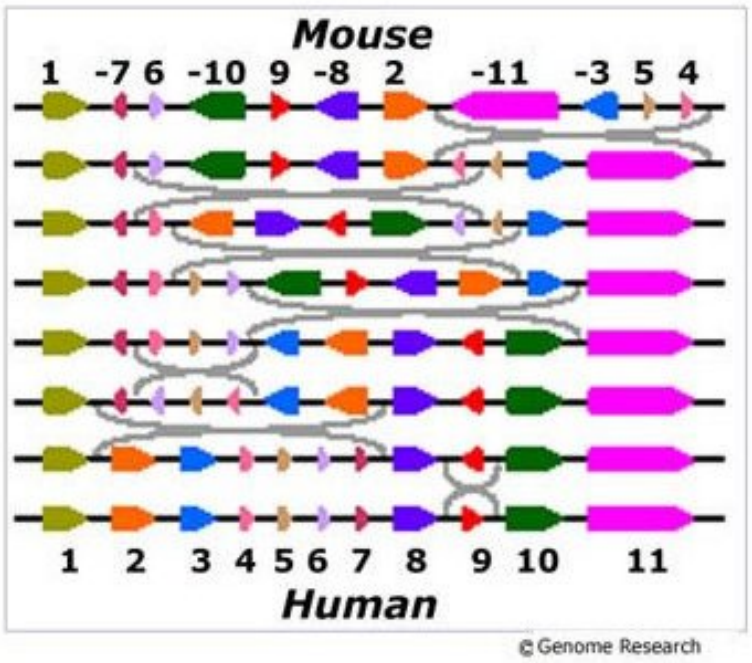
↪ Possible sur tout exemple

Étapes restantes :

- Calculer automatiquement la s.g. à partir des simples
- Calculer l'ensemble des simples dans une classe en temps polynomial
- Si $\mathcal{C}$ n'est pas donné par sa base finie ?

Motivations et modèle

- Génomes = permutations signées
- Renversement = renverser une fenêtre en changeant les signes



$$1\ \overline{7}\ 6\ \overline{10}\ 9\ \overline{8}\ 2\ \overline{11}\ \overline{3}\ 5\ 4$$

$$\Downarrow \text{ Renversement } \Downarrow$$

$$1\ \overline{7}\ 6\ \overline{10}\ 9\ \overline{8}\ 2\ \overline{4}\ \overline{5}\ 3\ 11$$

- Objectif = reconstruire des scénarios d'évolution
- Contrainte supplémentaire pour le tri parfait : **ne pas casser d'intervalle**
- Intervalle = groupe de gènes pour les deux espèces

Tri **(parfait)** par renversements

- Entrée : Deux permutations signées σ_1 et σ_2
- Sortie : Un scénario parcimonieux **(parfait)** de σ_1 à σ_2 ou $\overline{\sigma_2}$

On peut toujours supposer que $\sigma_2 = Id = 1\ 2 \dots n$

Tri par renversements :

- théorie de Hannenhalli-Pevzner
- Algorithmes polynomiaux : de $\mathcal{O}(n^4)$ à $\mathcal{O}(n\sqrt{n \log n})$

Remarque : problème *NP*-difficile sur les permutations non signées

Tri **parfait** par renversements :

- problème *NP*-difficile
- parcimonieux parfait $\not\Rightarrow$ parcimonieux
- Algorithme [BBCP] : algo FPT utilisant l'arbre de décomposition, en $\left(2^p \cdot n^{\mathcal{O}(1)}\right)$

Idée de l'algorithme

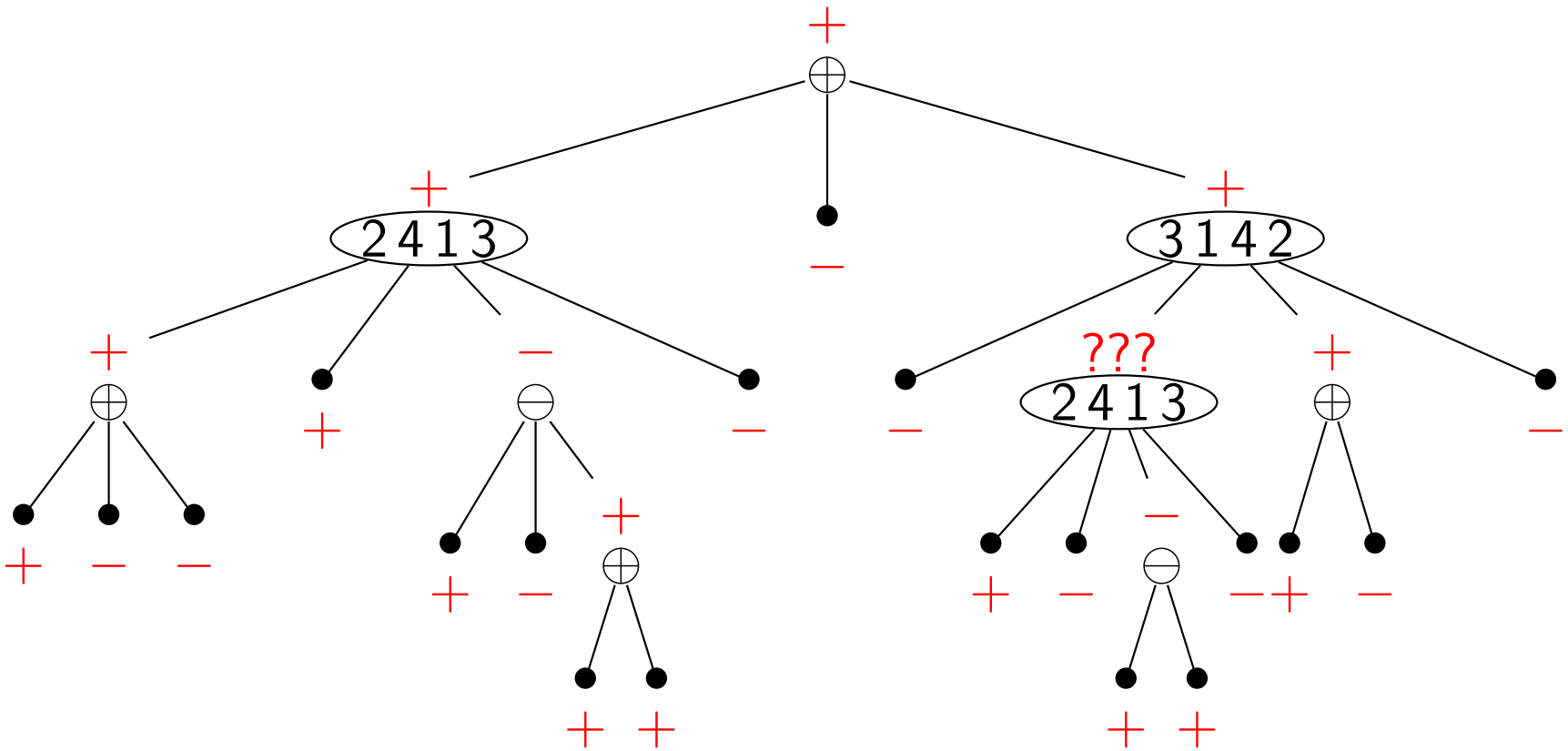
Donner un signe $+$ ou $-$ aux nœuds de l'arbre de décomposition de σ

- Feuille : signe de l'élément dans σ
- Nœud linéaire : $+$ pour $\oplus$ et $-$ pour $\ominus$
- Nœud premier dont le père est linéaire : signe de son père
- Autre nœud premier : ???
  - $\hookrightarrow$ Tester les signes $+$ et $-$ et choisir le scénario le plus court

Algorithme :

- Utiliser Hannenhalli-Pevzner sur les nœuds premiers
- Ajouter au scénario les nœuds dont le signe est différent de leur père (s'il est linéaire)

Exemple d'arbre de décomposition signé



Résultats de complexité

Résultats antérieurs :

- $\mathcal{O}(2^p n\sqrt{n \log n})$, où $p =$ nombre de nœuds premiers
- polynomial sur les permutations séparables ($p = 0$)

Analyse de complexité :

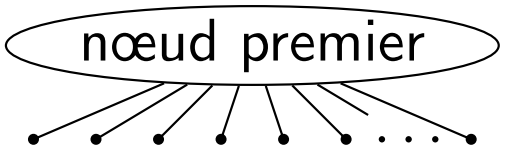
- polynomial avec probabilité 1 asymptotiquement
- polynomial en moyenne
- dans un scénario parcimonieux pour les permutations séparables
  - nombre moyen de renversements $\sim 1.2n$
  - taille moyenne d'un renversement $\sim 1.02\sqrt{n}$

Distribution de probabilité : toujours uniforme

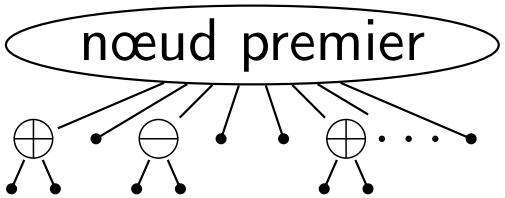
Forme "moyenne" des arbres de décomposition

Énumération des permutations simples : asymptotiquement $\frac{n!}{e^2}$

$\Rightarrow$ Asymptotiquement, une proportion $\frac{1}{e^2}$ d'arbres de décomposition sont réduits à un nœud premier.



Théorème : Asymptotiquement, la proportion d'arbres de décomposition avec une racine première dont les fils sont des feuilles ou des cerises est 1.



cerise = nœud linéaire à deux fils, qui sont des feuilles

Conséquence : Asymptotiquement, avec probabilité 1, l'algorithme tourne en temps polynomial.

Complexité moyenne

Complexité moyenne sur les permutations de taille n :

$$\frac{\sum_{p=0}^{n} |\{\sigma \text{ à } p \text{ nœuds premiers}\}| \; C \; 2^p n\sqrt{n \log n}}{n!}$$

Théorème : Lorsque $p \geq 2$, le nombre de permutations de taille n à p nœuds premiers est $\leq \frac{48(n-1)!}{2^p}$

Conséquence : La complexité moyenne sur les permutations de taille n est $\leq 50Cn\sqrt{n \log n}$.
En particulier, algorithme **polynomial en moyenne.**

Permutations séparables

Rappel : pas de nœud premier dans l'arbre de décomposition.

$$\text{Donc renversements} = \begin{cases} \text{nœuds internes sauf la racine} \\ \text{feuilles de signe différent du père} \end{cases}$$

Remarque :
Ici, scénario = ensemble d'intervalles ; ordre sans importance.

Il suffit de calculer :

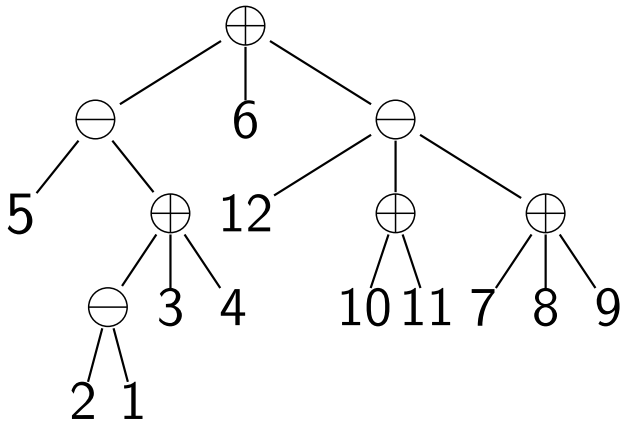
- nombre moyen de nœuds internes
- taille moyenne d'un sous-arbre

dans un arbre de décomposition d'une permutation séparable . . .

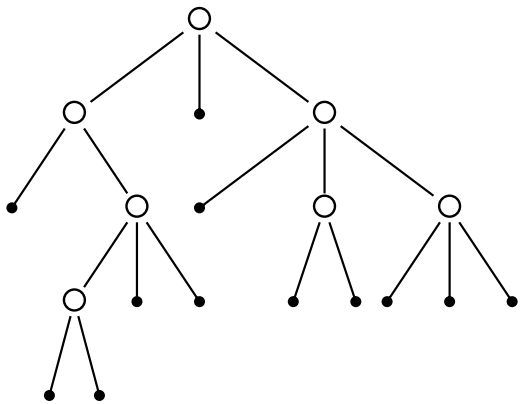
Bijection entre perm. séparables et arbres de Schröder

. . . c'est-à-dire dans un arbre de Schröder signé.

Permutation séparable



Arbre de Schröder



\+ signe $\oplus$ à la racine

taille de σ	$\longleftrightarrow$	nombre de feuilles
renversement (hors feuilles)	$\longleftrightarrow$	nœud interne (hors racine)
taille d'un renversement	$\longleftrightarrow$	nombre de feuilles du sous-arbre

Combinatoire analytique

Séries génératrices bivariées et valeur moyenne de paramètres :

- $S(x,y) = \sum s_{n,k} x^n y^k$, où $s_{n,k}$ = nombre d'arbres de Schröder à n feuilles et k nœuds internes.
- $S(x,y) = \sum s_{n,k} x^n y^k$, où $s_{n,k}$ = nombre d'arbres de Schröder à n feuilles dont les tailles des sous-arbres somment à k.

Étapes de l'analyse :

- Équations satisfaites par les séries.
- Équivalent de $\frac{\partial S(x,y)}{\partial y}|_{y=1}$ et $S(x,1)$ autour de la singularité dominante.
- Équivalent asymptotique de $\frac{\sum_k k s_{n,k}}{\sum_k s_{n,k}} = \frac{[x^n]\frac{\partial S(x,y)}{\partial y}|_{y=1}}{[x^n]S(x,1)}$.

Résultats asymptotiques

Arbres de Schröder :

- Nombre moyen de nœuds internes : $\sim \frac{n}{\sqrt{2}}$
- Valeur moyenne de la somme des tailles des sous-arbres : $\sim 2^{3/4}\sqrt{3-2\sqrt{2}}\,\sqrt{\pi n^3}$

Permutations séparables **signées** :

- Nombre moyen de renversements : $\sim \frac{1+\sqrt{2}}{2} n$
- Valeur moyenne de la somme des tailles des renversements : $\sim 2^{3/4}\sqrt{3-2\sqrt{2}}\,\sqrt{\pi n^3}$
- Taille moyenne d'un renversement : $\sim \frac{2^{7/4}\sqrt{3-2\sqrt{2}}}{1+\sqrt{2}}\sqrt{\pi n} \sim 1.02\sqrt{n}$

Perspectives

- Recherche de motif : *NP*-difficile. Mais FPT ($\mathcal{O}(f(k)n^{\mathcal{O}(1)})$) ?
- Calcul des séries génératrices des $S(B)$ contenant un nombre fini de simples
- Utilisation pour la génération aléatoire
- Analyse fine d'autres algorithmes utilisant des arbres de décomposition (DCJ)
- Extension des concepts de théorie de graphes vers les permutations, et vice-versa

- Autres questions algorithmiques et combinatoires sur les permutations. Suggestions ?