Permutations séparables : combinatoire et algorithmique

Mathilde Bouvel

15 septembre 2006

Stage effectué au LIAFA
Équipe Algorithmique et Combinatoire
sous la direction de Dominique Rossin

Plan

Contexte et motivations

- Permutations à motifs exclus
  - Permutations séparables
  - Permutations triables par pile
- Problème algorithmique : recherche de plus grande "sous-structure" commune
  - Plus grand sous-graphe commun : *NP*-complet
  - Plus grand sous-arbre commun : dans *P*
  - Plus grand motif commun à deux permutations : *NP*-dur
  - Occurrence d'un motif dans une permutation : *NP*-complet
- Plus grand sous-arbre commun = plus grand motif commun à deux permutations triables par pile

Contexte et motivations

- Permutations à motifs exclus
  - Permutations séparables
  - Permutations triables par pile
- Problème algorithmique : recherche de plus grande "sous-structure" commune
  - Plus grand sous-graphe commun : *NP*-complet
  - Plus grand sous-arbre commun : dans *P*
  - Plus grand motif commun à deux permutations : *NP*-dur
  - Occurrence d'un motif dans une permutation : *NP*-complet
- Plus grand sous-arbre commun = plus grand motif commun à deux permutations triables par pile

Contexte et motivations

- Permutations à motifs exclus
  - Permutations séparables
  - Permutations triables par pile
- Problème algorithmique : recherche de plus grande "sous-structure" commune
  - Plus grand sous-graphe commun : *NP*-complet
  - Plus grand sous-arbre commun : dans *P*
  - Plus grand motif commun à deux permutations : *NP*-dur
  - Occurrence d'un motif dans une permutation : *NP*-complet
- Plus grand sous-arbre commun = plus grand motif commun à deux permutations triables par pile

Pourquoi les permutations séparables ?

- Lien fort entre sous-**arbre** commun et motif commun pour les permutations **triables par pile**
- Pour une surclasse des permutations triables par pile : vers la recherche de sous-graphe commun
- **Séparables** = surclasse des triables par pile
- Étude antérieure d'un problème **algorithmique** sur les séparables :
  - Recherche d'une occurrence d'un motif séparable dans une permutation quelconque : dans P [Bose, Buss, Lubiw]

Pourquoi les permutations séparables ?

- Lien fort entre sous-**arbre** commun et motif commun pour les permutations **triables par pile**
- Pour une surclasse des permutations triables par pile : vers la recherche de sous-graphe commun
- **Séparables** = surclasse des triables par pile
- Étude antérieure d'un problème **algorithmique** sur les séparables :
  - Recherche d'une occurrence d'un motif séparable dans une permutation quelconque : dans P [Bose, Buss, Lubiw]

Motifs dans les permutations

Représentation linéaire des permutations : $\pi =$ mot $\pi_1 \pi_2 \ldots \pi_n$

$\pi \in S_n$, $\tau \in S_k$ avec $k \leq n$

- La permutation π *contient* une occurrence du motif τ ssi $\exists$ $1 \leq i_1 < i_2 < \ldots < i_k \leq n$ tels que $\pi_{i_1} \pi_{i_2} \ldots \pi_{i_k}$ est isomorphe en ordre à τ : $\pi_{i_p} < \pi_{i_q}$ ssi $\tau_p < \tau_q$
- dans le cas contraire, π *évite* τ
- Par exemple, **1**3**5**6**24** contient 132 et évite 321

Notation :
$S(\tau) =$ l'ensemble de toutes les permutations qui évitent τ

Définition des permutations séparables

- Permutations séparables = $S(2413, 3142)$
- Permutations séparables = celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles $\pi_1, \pi_2, \ldots, \pi_n$ de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation

Définition des permutations séparables

- Permutations séparables = $S(2413, 3142)$
- Permutations séparables = celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles π_1, π_2, ..., π_n de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation

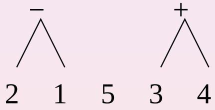
Définition des permutations séparables

- Permutations séparables = $S(2413, 3142)$
- Permutations séparables = celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles π_1, π_2, . . ., π_n de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation

2 1 5 3 4

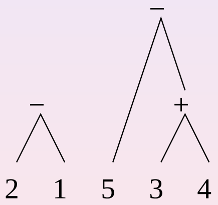
Définition des permutations séparables

- Permutations séparables = $S(2413, 3142)$
- Permutations séparables = celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles π_1, π_2, ..., π_n de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation



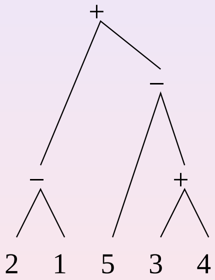
Définition des permutations séparables

- Permutations séparables $= S(2413, 3142)$
- Permutations séparables $=$ celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles π_1, π_2, $\ldots$, π_n de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation



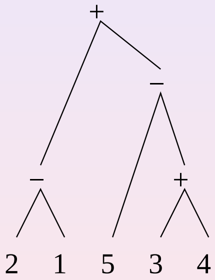
Définition des permutations séparables

- Permutations séparables = $S(2413, 3142)$
- Permutations séparables = celles possédant un arbre de séparation
    - arbre binaire plan
    - feuilles π_1, π_2, ..., π_n de gauche à droite
    - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation



Définition des permutations séparables

- Permutations séparables $= S(2413, 3142)$
- Permutations séparables $=$ celles possédant un arbre de séparation
  - arbre binaire plan
  - feuilles $\pi_1, \pi_2, \ldots, \pi_n$ de gauche à droite
  - à chaque noeud interne correspond un intervalle
- Remarque : 2413 et 3142 n'ont pas d'arbre de séparation



Énumération des permutations séparables

Nombre de permutations séparables de taille n : s_{n-1}, **nombres de Schröder**

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

Série génératrice :

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

Formule close :

$$s_n = \sum_{i=0}^{n} \binom{2n-i}{i} c_{n-i}$$

Énumération des permutations séparables

Nombre de permutations séparables de taille n : s_{n-1}, **nombres de Schröder**

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

- Preuve existante [West] : arbres de génération $\longrightarrow$ calculatoire
- Nouvelle preuve : bijective
  - permutations séparables
  - arbres de Schröder signés
  - chemins de Schröder

Plan

Complexité de la recherche d'une occurrence d'un motif

- Motif quelconque : problème *NP*-complet
- Motif **séparable** : problème résolu en temps **polynomial**
  - Algorithme de Bose, Buss et Lubiw (BBL) : $\mathcal{O}(kn^6)$ en temps, $\mathcal{O}(kn^4)$ en espace
  - Algorithme d'Ibarra : $\mathcal{O}(kn^4)$ en temps, $\mathcal{O}(kn^3)$ en espace

- Exploiter la structure d'**arbre de séparation** (précalcul)
- Outil clé = **programmation dynamique**

Complexité de la recherche d'une occurrence d'un motif

- Motif quelconque : problème NP-complet
- Motif **séparable** : problème résolu en temps **polynomial**
  - Algorithme de Bose, Buss et Lubiw (BBL) : $\mathcal{O}(kn^6)$ en temps, $\mathcal{O}(kn^4)$ en espace
  - Algorithme d'Ibarra : $\mathcal{O}(kn^4)$ en temps, $\mathcal{O}(kn^3)$ en espace

- Exploiter la structure d'**arbre de séparation** (précalcul)
- Outil clé = **programmation dynamique**

Complexité de la recherche d'une occurrence d'un motif

- Motif quelconque : problème *NP*-complet
- Motif **séparable** : problème résolu en temps **polynomial**
  - Algorithme de Bose, Buss et Lubiw (BBL) : $\mathcal{O}(kn^6)$ en temps, $\mathcal{O}(kn^4)$ en espace
  - Algorithme d'Ibarra : $\mathcal{O}(kn^4)$ en temps, $\mathcal{O}(kn^3)$ en espace

- Exploiter la structure d'**arbre de séparation** (précalcul)
- Outil clé = **programmation dynamique**

Algorithme BBL

- Entrée : motif τ séparable + arbre de séparation T ; permutation π
- Tableau de prog. dynamique : $M(V, i, j, a, b) = 1$ ou 0 selon qu'il existe ou non une occurrence du sous-motif de τ sous le noeud V dans $\pi_i \dots \pi_j$ utilisant des valeurs entre a et b

Exemple : V représente 21, $\pi = 6\ 4\ 2\ 5\ 3\ 1$

- $M(V, 2, 4, 3, 5) = 0$
- $M(V, 2, 5, 3, 5) = 1$
- $M(V, 2, 4, 2, 5) = 1$

Algorithme BBL

Équations de programmation dynamique :

- Feuilles : $M(V, i, j, a, b) = 1$ ssi il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi_h \leq b$
- V positif : $M(V, i, j, a, b) = Max\{M(V_L, i, h-1, a, c-1) \cdot M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$
- V négatif : $M(V, i, j, a, b) = Max\{M(V_L, i, h-1, c, b) \cdot M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$

Algorithme BBL

Équations de programmation dynamique :

- Feuilles : $M(V, i, j, a, b) = 1$ ssi il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi_h \leq b$
- V positif : $M(V, i, j, a, b) = Max\{M(V_L, i, h-1, a, c-1) \cdot M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$
- V négatif : $M(V, i, j, a, b) = Max\{M(V_L, i, h-1, c, b) \cdot M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$

Algorithme BBL

Équations de programmation dynamique :

- Feuilles : $M(V,i,j,a,b) = 1$ ssi il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi_h \leq b$
- V positif : $M(V,i,j,a,b) = Max\{M(V_L, i, h-1, a, c-1) \cdot M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$
- V négatif : $M(V,i,j,a,b) = Max\{M(V_L, i, h-1, c, b) \cdot M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$

Algorithme BBL

Équations de programmation dynamique :

- Feuilles : $M(V,i,j,a,b) = 1$ ssi il existe $h \in \{i, i+1, \dots, j\}$ tel que $a \leq \pi_h \leq b$
- V positif : $M(V,i,j,a,b) = Max\{M(V_L, i, h-1, a, c-1) \cdot M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$
- V négatif : $M(V,i,j,a,b) = Max\{M(V_L, i, h-1, c, b) \cdot M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$

Complexité de l'algorithme BBL

- Complexité en espace : $\mathcal{O}(kn^4)$
- Pour la complexité en temps :
  - $\mathcal{O}(kn^5)$ pour les feuilles
  - $\mathcal{O}(n^2)$ pour un noeud interne à partir des tableaux des fils
  - $\mathcal{O}(kn^6)$ au total

Adapter l'algorithme de Bose, Buss et Lubiw

- Plus grand motif commun : *NP*-dur dans le cas général
- Si une permutation est séparable, utiliser la structure d'arbre de séparation
- Programmation dynamique : $M(V, i, j, a, b) =$ **un plus grand motif commun** à $\pi[V]$, π séparable et à π' quelconque, indices dans π' entre i et j, valeurs dans π' entre a et b

- Complexité en espace : $\mathcal{O}(nn'^4 min(n, n'))$
- Complexité en temps : $\mathcal{O}(nn'^6 min(n, n'))$, voire $\mathcal{O}(nn'^6)$

Adapter l'algorithme de Bose, Buss et Lubiw

- Plus grand motif commun : NP-dur dans le cas général
- Si une permutation est séparable, utiliser la structure d'arbre de séparation
- Programmation dynamique : $M(V, i, j, a, b) =$ **un plus grand motif commun** à $\pi[V]$, π séparable et à π' quelconque, indices dans π' entre i et j, valeurs dans π' entre a et b

- Complexité en espace : $\mathcal{O}(nn'^4 min(n, n'))$
- Complexité en temps : $\mathcal{O}(nn'^6 min(n, n'))$, voire $\mathcal{O}(nn'^6)$

Comment ça marche ?

- Concaténation positive :
  $p \oplus p' = p_1 \cdots p_k (p'_1 + k) \cdots (p'_{k'} + k)$
- Concaténation négative :
  $p \ominus p' = (p_1 + k') \cdots (p_k + k') p'_1 \cdots p'_{k'}$
- Feuille V : $M(V, i, j, a, b) = 1$ ou motif vide selon qu'il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi'_h \leq b$ ou pas
- Noeud V positif : $M(V, i, j, a, b) =$ “Max” de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$.
- Noeud V négatif : $M(V, i, j, a, b) =$ “Max” de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, c, b) \ominus M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$.
- Remarque clé : la propriété d'être un plus grand motif commun est **héréditaire**.

Comment ça marche ?

- Concaténation positive :
  $p \oplus p' = p_1 \cdots p_k (p'_1 + k) \cdots (p'_{k'} + k)$
- Concaténation négative :
  $p \ominus p' = (p_1 + k') \cdots (p_k + k') p'_1 \cdots p'_{k'}$
- Feuille V : $M(V, i, j, a, b) = 1$ ou motif vide selon qu'il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi'_h \leq b$ ou pas
- Noeud V positif : $M(V, i, j, a, b) =$ "Max" de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$.
- Noeud V négatif : $M(V, i, j, a, b) =$ "Max" de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, c, b) \ominus M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$.
- Remarque clé : la propriété d'être un plus grand motif commun est **héréditaire**.

Comment ça marche ?

- Concaténation positive : $p \oplus p' = p_1 \cdots p_k (p'_1 + k) \cdots (p'_{k'} + k)$
- Concaténation négative : $p \ominus p' = (p_1 + k') \cdots (p_k + k') p'_1 \cdots p'_{k'}$
- Feuille V : $M(V, i, j, a, b) = 1$ ou motif vide selon qu'il existe $h \in \{i, i+1, \ldots, j\}$ tel que $a \leq \pi'_h \leq b$ ou pas
- Noeud V positif : $M(V, i, j, a, b) =$ "Max" de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, a, c-1) \oplus M(V_R, h, j, c, b) : i < h \leq j, a < c \leq b\}$.
- Noeud V négatif : $M(V, i, j, a, b) =$ "Max" de $\{M(V_L, i, j, a, b), M(V_R, i, j, a, b)\} \cup \{M(V_L, i, h-1, c, b) \ominus M(V_R, h, j, a, c-1) : i < h \leq j, a < c \leq b\}$.
- Remarque clé : la propriété d'être un plus grand motif commun est **héréditaire**.

Arbre de décomposition (décoré développé)

Cet arbre existe pour une permutation **quelconque**.
Décomposition en "intervalles communs" des permutations.
Lien avec la décomposition modulaire des graphes.
Calculable en **temps linéaire**.

- Essayer de construire un arbre de séparation.
- Quand la construction est bloquée, introduire de nouveaux noeuds dans l'arbre, dits noeuds **premiers**.
- Un noeud premier est étiqueté par une permutation simple (i.e. sans intervalle commun).
- Exemple : 246135 est simple ; 261**354** n'est pas simple.

Pour une permutation **séparable**, arbre de séparation = arbre de décomposition.

Arbre de décomposition (décoré développé)

Cet arbre existe pour une permutation **quelconque**.
Décomposition en "intervalles communs" des permutations.
Lien avec la décomposition modulaire des graphes.
Calculable en **temps linéaire**.

- Essayer de construire un arbre de séparation.
- Quand la construction est bloquée, introduire de nouveaux noeuds dans l'arbre, dits noeuds **premiers**.
- Un noeud premier est étiqueté par une permutation simple (i.e. sans intervalle commun).
- Exemple : 246135 est simple ; 261**354** n'est pas simple.

Pour une permutation **séparable**, arbre de séparation = arbre de décomposition.

Arbre de décomposition (décoré développé)

Cet arbre existe pour une permutation **quelconque**.
Décomposition en "intervalles communs" des permutations.
Lien avec la décomposition modulaire des graphes.
Calculable en **temps linéaire**.

- Essayer de construire un arbre de séparation.
- Quand la construction est bloquée, introduire de nouveaux noeuds dans l'arbre, dits noeuds **premiers**.
- Un noeud premier est étiqueté par une permutation simple (i.e. sans intervalle commun).
- Exemple : 246135 est simple ; 261**354** n'est pas simple.

Pour une permutation **séparable**, arbre de séparation = arbre de décomposition.

Algorithme

Le même qu'avant !

- Pour un noeud premier d'étiquette σ, on concatène les motifs trouvés dans les fils selon l'ordre de valeurs donné par σ :

V premier ayant d fils $V_1, \dots, V_d$. Pour calculer $M(V, i, j, a, b)$, on :

- coupe $\{i, \dots j\}$ en d intervalles d'indices $I_1, \dots, I_d$ de gauche à droite
- coupe $\{a, \dots b\}$ en d intervalles de valeurs $A_1, \dots, A_d$ (rangés par ordre croissant)
- associe l'intervalle d'indices I_k à l'intervalle de valeurs A_{σ_k}
- σ-concatène les plus grands motifs communs aux V_k et à π' utilisant l'intervalle d'indices I_k et l'intervalle de valeurs A_{σ_k} dans π'

Complexité

- Différence dans l'analyse de complexité : Pour un noeud premier d'arité d, calcul d'une case du tableau à partir des tableaux des fils en $\mathcal{O}(n'^{2d-2})$.
- Borne optimale : $d \leq n \rightarrow$ algorithme *a priori* non polynomial.
- Mais l'algorithme est polynomial pour la recherche de plus grand motif commun à une permutation dont **l'arbre de décomposition a tous ses noeuds premiers d'arité bornée par une constante** d et à une permutation quelconque.

Complexité

- Différence dans l'analyse de complexité : Pour un noeud premier d'arité d, calcul d'une case du tableau à partir des tableaux des fils en $\mathcal{O}(n'^{2d-2})$.
- Borne optimale : $d \leq n \rightarrow$ algorithme *a priori* non polynomial.

- Mais l'algorithme est polynomial pour la recherche de plus grand motif commun à une permutation dont **l'arbre de décomposition a tous ses noeuds premiers d'arité bornée par une constante** d et à une permutation quelconque.

Contribution du stage

Combinatoire :

- **Caractérisation des permutations séparables par leur arbre de décomposition**
- **Preuve bijective du résultat d'énumération**

Algorithmique :

- Recherche d'une occurrence d'un motif séparable : $\mathcal{O}(kn^4)$
- **Recherche d'un plus grand motif commun, une séparable :** $\mathcal{O}(nn'^6)$
- **Recherche d'un plus grand motif commun, arbre de décomposition :** $\mathcal{O}(nn'^{2d+2})$

Étude d'autres objets combinatoires (mots bicoloriés)

Bibliographie

- A. Bergeron, C. Chauve, F. de Montgolfier, M. Raffinot, *Computing Common Intervals of K Permutations, with Applications to Modular Decomposition of Graphs*, ESA 2005, 779-790
- P. Bose, J. F. Buss, A. Lubiw, *Pattern matching for permutations*, Information Processing Letters 65 (1998), 277-283
- L. Ibarra, *Finding pattern matchings for permutations*, Information Processing Letters 61 (1997) 293-295
- J. West, *Generating trees and the Catalan and Schröder numbers*, Discrete Mathematics 146, 247-262 (1995)