Module EA4 – Éléments d'Algorithmique II

Dominique Poulalhon
dominique.poulalhon@liafa.univ-paris-diderot.fr

Université Paris Diderot
L2 Informatique
Année universitaire 2014-2015

Interrogation n° 1 :

en TD la première semaine de mars

permutation de taille n = bijection de $[\![1,n]\!]$ dans lui-même

Permutations

permutation de taille n = bijection de $[\![1,n]\!]$ dans lui-même

$\mathfrak{S}_n$ = ensemble des permutations de taille n

permutation de taille n = bijection de $[\![1,n]\!]$ dans lui-même

$\mathfrak{S}_n$ = ensemble des permutations de taille n

notation bilinéaire : $\sigma = \begin{pmatrix} 1 & 2 & \dots & n \\ \sigma(1) & \sigma(2) & \dots & \sigma(n) \end{pmatrix}$

permutation de taille n = bijection de $[\![1,n]\!]$ dans lui-même

$\mathfrak{S}_n$ = ensemble des permutations de taille n

notation bilinéaire : $\sigma = \begin{pmatrix} 1 & 2 & \dots & n \\ \sigma(1) & \sigma(2) & \dots & \sigma(n) \end{pmatrix}$

notation linéaire : $\sigma = \sigma(1)\ \sigma(2)\ \dots\ \sigma(n)$

permutation de taille n $=$ bijection de $[\![1,n]\!]$ dans lui-même

$\mathfrak{S}_n$ $=$ ensemble des permutations de taille n

notation bilinéaire : $\sigma = \begin{pmatrix} 1 & 2 & \dots & n \\ \sigma(1) & \sigma(2) & \dots & \sigma(n) \end{pmatrix}$

notation linéaire : $\sigma = \sigma(1)\ \sigma(2)\ \dots\ \sigma(n)$

produit : $\sigma\tau = \sigma \circ \tau \ :\ i \longmapsto \sigma(\tau(i))$

permutation de taille n = bijection de $[\![1,n]\!]$ dans lui-même

$\mathfrak{S}_n$ = ensemble des permutations de taille n

notation bilinéaire : $\sigma = \begin{pmatrix} 1 & 2 & \ldots & n \\ \sigma(1) & \sigma(2) & \ldots & \sigma(n) \end{pmatrix}$

notation linéaire : $\sigma = \sigma(1)\ \sigma(2)\ \ldots\ \sigma(n)$

produit : $\sigma\tau = \sigma \circ \tau \ : \ i \longmapsto \sigma(\tau(i))$

Lemme

$\sigma, \tau \in \mathfrak{S}_n \implies \sigma\tau \in \mathfrak{S}_n$ *(loi de composition interne)* produit : $\sigma\tau = \sigma \circ \tau \ : \ i \longmapsto \sigma(\tau(i))$

inverse de σ : application τ telle que $\tau\sigma = id_n = 1\ 2\ \ldots\ n$

notation : σ^{-1}

$$i \xmapsto{\ \sigma\ } \sigma(i) \xmapsto{\ \tau = \sigma^{-1}\ } i$$

produit : $\sigma\tau = \sigma \circ \tau \ : \ i \longmapsto \sigma(\tau(i))$

inverse de σ : application τ telle que $\tau\sigma = \mathrm{id}_n = 1\ 2\ \ldots\ n$

notation : σ^{-1}

$$i \xmapsto{\ \sigma\ } \sigma(i) \xmapsto{\ \tau = \sigma^{-1}\ } i$$

Lemme

- $\sigma \in \mathfrak{S}_n \implies \sigma^{-1} \in \mathfrak{S}_n$
- $\sigma\sigma^{-1} = \sigma^{-1}\sigma = \mathrm{id}_n$
- $(\sigma^{-1})^{-1} = \sigma$

(on dit que $\mathfrak{S}_n$ a une structure de groupe)

Tris *vs.* permutations

tableau à trier

tableau trié

Tris *vs.* permutations

tableau à trier

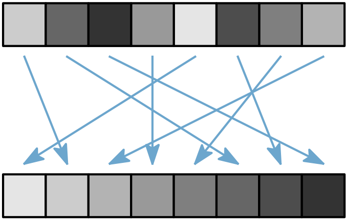

tableau trié

Tris *vs.* permutations

tableau à trier

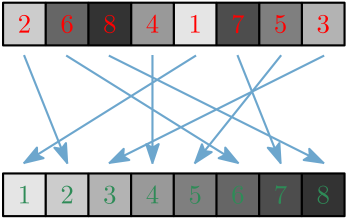

tableau trié

TRIS *vs.* PERMUTATIONS

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

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

TRIS *vs.* PERMUTATIONS

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

$$\circ\sigma^{-1}$$

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

Orbites, cycles

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \ldots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal = ordre de i = longueur de l'orbite

Orbites, cycles

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \dots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal = ordre de i = longueur de l'orbite

si i et j appartiennent à la même orbite γ, on note
$\gamma = (i \;\; \sigma(i) \;\; \sigma^2(i) \;\; \dots \;\; \sigma^{-1}(i)) = (j \;\; \sigma(j) \;\; \sigma^2(j) \;\; \dots \;\; \sigma^{-1}(j))$

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \ldots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal $=$ ordre de i $=$ longueur de l'orbite

si i et j appartiennent à la même orbite γ, on note
$\gamma = (i \;\; \sigma(i) \;\; \sigma^2(i) \;\; \ldots \;\; \sigma^{-1}(i)) = (j \;\; \sigma(j) \;\; \sigma^2(j) \;\; \ldots \;\; \sigma^{-1}(j))$

cycle : permutation dont une seule orbite est non triviale

Orbites, cycles

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \ldots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal $=$ ordre de i $=$ longueur de l'orbite

si i et j appartiennent à la même orbite γ, on note
$\gamma = (i \;\; \sigma(i) \;\; \sigma^2(i) \;\; \ldots \;\; \sigma^{-1}(i)) = (j \;\; \sigma(j) \;\; \sigma^2(j) \;\; \ldots \;\; \sigma^{-1}(j))$

cycle : permutation dont une seule orbite est non triviale

notation cyclique : $\sigma = \gamma_1 \; \gamma_2 \; \ldots \; \gamma_\ell$ où chaque γ_k est une orbite

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \ldots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal $=$ ordre de i $=$ longueur de l'orbite

si i et j appartiennent à la même orbite γ, on note
$\gamma = (i \;\; \sigma(i) \;\; \sigma^2(i) \;\; \ldots \;\; \sigma^{-1}(i)) = (j \;\; \sigma(j) \;\; \sigma^2(j) \;\; \ldots \;\; \sigma^{-1}(j))$

cycle : permutation dont une seule orbite est non triviale

notation cyclique : $\sigma = \gamma_1 \; \gamma_2 \; \ldots \; \gamma_\ell$ où chaque γ_k est une orbite

Exemple $\quad \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 6 & 1 & 5 & 3 \end{pmatrix} = (1\;2\;4)\;(3\;6)\;(5)$

Orbites, cycles

orbite d'un élément i : ensemble de ses itérés : $i, \sigma(i), \sigma^2(i), \ldots$
pour chaque i, il existe nécessairement un $p > 0$ tq $\sigma^p(i) = i$
p minimal = ordre de i = longueur de l'orbite

si i et j appartiennent à la même orbite γ, on note
$\gamma = (i \;\; \sigma(i) \;\; \sigma^2(i) \;\; \ldots \;\; \sigma^{-1}(i)) = (j \;\; \sigma(j) \;\; \sigma^2(j) \;\; \ldots \;\; \sigma^{-1}(j))$

cycle : permutation dont une seule orbite est non triviale

notation cyclique : $\sigma = \gamma_1 \; \gamma_2 \; \ldots \; \gamma_\ell$ où chaque γ_k est une orbite

point fixe : 1-cycle, *i.e.* élément $i \in [\![1, n]\!]$ tq $\sigma(i) = i$

support : $\mathrm{Supp}(\sigma) = \{i \in [\![1, n]\!] \;\text{ tq }\; \sigma(i) \neq i\}$

Notation cyclique

notation cyclique : $\sigma = \gamma_1 \, \gamma_2 \, \ldots \, \gamma_\ell$ où chaque γ_k est un cycle
= décomposition en produit de cycles disjoints

usuellement, on omet les points fixes

Notation cyclique

notation cyclique : $\sigma = \gamma_1\,\gamma_2\ \ldots\ \gamma_\ell$ où chaque γ_k est un cycle
$=$ décomposition en produit de cycles disjoints

usuellement, on omet les points fixes

Exemple $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 6 & 1 & 5 & 3 \end{pmatrix} = (1\ 2\ 4)\ (3\ 6)$

notation cyclique : $\sigma = \gamma_1\, \gamma_2\, \ldots\, \gamma_\ell$ où chaque γ_k est un cycle
= décomposition en produit de cycles disjoints

usuellement, on omet les points fixes

Exemple $\quad \sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 6 & 1 & 5 & 3 \end{pmatrix} = (1\ 2\ 4)\ (3\ 6)$

Remarque : c'est bien un produit de permutations :

$$\text{si } \gamma_1 = (1\ 2\ 4) \quad \text{et} \quad \gamma_2 = (3\ 6)$$

$$\textit{i.e.} \text{ si } \gamma_1 = (1\ 2\ 4)(3)(5)(6) \quad \text{et} \quad \gamma_2 = (3\ 6)(1)(2)(4)(5)$$

$$\text{alors } \sigma = \gamma_1 \circ \gamma_2 = \gamma_2 \circ \gamma_1$$

Proposition

si $\mathrm{Supp}(\sigma) \cap \mathrm{Supp}(\tau) = \emptyset$, *alors* $\sigma\tau = \tau\sigma$

transposition = cycle de longueur 2

Transpositions

transposition = cycle de longueur 2

action par produit à gauche : si $\sigma \in \mathfrak{S}_n$, alors

$$(i\ j)\,\sigma = (i\ j) \circ \sigma \ :\ k \longmapsto \begin{cases} i & \text{si } k = \sigma^{-1}(j) \\ j & \text{si } k = \sigma^{-1}(i) \\ \sigma(k) & \text{sinon} \end{cases}$$

Transpositions

transposition = cycle de longueur 2

action par produit à gauche : si $\sigma \in \mathfrak{S}_n$, alors

$$(i\ j)\ \sigma = (i\ j) \circ \sigma \ :\ k \longmapsto \begin{cases} i & \text{si } k = \sigma^{-1}(j) \\ j & \text{si } k = \sigma^{-1}(i) \\ \sigma(k) & \text{sinon} \end{cases}$$

= échange des valeurs i et j

action par produit à droite : si $\sigma \in \mathfrak{S}_n$, alors

$$\sigma\ (i\ j) = \sigma \circ (i\ j) \ :\ k \longmapsto \begin{cases} \sigma(j) & \text{si } k = i \\ \sigma(i) & \text{si } k = j \\ \sigma(k) & \text{sinon} \end{cases}$$

transposition = cycle de longueur 2

action par produit à gauche : si $\sigma \in \mathfrak{S}_n$, alors

$$(i\ j)\ \sigma = (i\ j) \circ \sigma\ :\ k \longmapsto \begin{cases} i & \text{si } k = \sigma^{-1}(j) \\ j & \text{si } k = \sigma^{-1}(i) \\ \sigma(k) & \text{sinon} \end{cases}$$

= échange des valeurs i et j

action par produit à droite : si $\sigma \in \mathfrak{S}_n$, alors

$$\sigma\ (i\ j) = \sigma \circ (i\ j)\ :\ k \longmapsto \begin{cases} \sigma(j) & \text{si } k = i \\ \sigma(i) & \text{si } k = j \\ \sigma(k) & \text{sinon} \end{cases}$$

= échange des éléments en positions i et j

Lemme

tout cycle de longueur ℓ peut s'écrire comme produit de $\ell - 1$ transpositions

Corollaire

toute permutation peut s'écrire comme produit d'au plus $n - 1$ transpositions

(on dit que les transpositions engendrent $\mathfrak{S}_n$)

Corollaire

tout tableau peut être trié en au plus $n - 1$ échanges

(l'histoire ne dit pas comment trouver les échanges à effectuer, donc combien de comparaisons sont nécessaires)

Aparté : génération aléatoire de permutations

`RandomPermutation(n)`

construire une des $n!$ permutations de taille n selon la loi de probabilité uniforme

(*i.e.* : si on exécute tous les comportements (aléatoires) possibles, chaque permutation doit être obtenue le même nombre de fois)

Aparté : génération aléatoire de permutations

RandomPermutation(n)

construire une des $n!$ permutations de taille n selon la loi de probabilité uniforme

(*i.e.* : si on exécute tous les comportements (aléatoires) possibles, chaque permutation doit être obtenue le même nombre de fois)

idée : construire une décomposition en produit de transpositions $(a_1\ b_1)(a_2\ b_2)\dots(a_\ell\ b_\ell)$ avec la contrainte suivante :

$$\forall i \leqslant \ell, \quad a_i < b_i \quad \text{et} \quad a_1 < a_2 < \cdots < a_\ell$$

Lemme

ces $n!$ décompositions distinctes donnent $n!$ produits distincts

Aparté : génération aléatoire de permutations

RandomPermutation(n)

construire une des $n!$ permutations de taille n selon la loi de probabilité uniforme

```
from random import randint
# générateur uniforme d'entiers dans un intervalle
def randomPerm(n) :
  l = [ i+1 for i in range(n) ]
  for i in range(n) :
    r = randint(i, n-1)
    if i != r : l[i], l[r] = l[r], l[i]
  return l
```