EA4 – Éléments d'algorithmique
TD n° 5

Exercice 1 : décomposition de permutation

Soit la permutation $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 5 & 9 & 7 & 2 & 1 & 3 & 10 & 8 & 4 & 6 \end{pmatrix}$

1. Quelles sont les orbites de σ ?

2. Quelle est l'orbite de 2, quel est son ordre ?

3. Quels sont les points fixes de σ ?

4. Décomposer σ en produit de cycles.

5. Calculer σ^{-1}.

6. Décomposer σ en produit de transpositions de deux façons différentes.

On rappelle qu'une *inversion* d'une permutation ς est un couple d'indices (i, j) tel que $i < j$ et $\varsigma(i) > \varsigma(j)$. On note $\texttt{Inv}(\varsigma)$ le nombre d'inversions de ς.

7. Quelles sont les inversions de σ ?

8. (*) Montrer que pour toute permutation $\varsigma \in \mathfrak{S}_n$ il existe $p > 0$ tel que $\varsigma^p = id_n$. Donner une borne supérieure de la valeur minimale de p en fonction de n, et en fonction de ς.

9. (*) Quel est le p minimal pour σ, la permutation en exemple ?

Exercice 2 : compter les inversions

Soit $\sigma \in \mathfrak{S}_n$ une permutation. Modifier l'algorithme de tri par fusion pour obtenir un algorithme qui calcule $\texttt{Inv}(\sigma)$ en temps $\Theta(n \log n)$ dans le pire des cas.

Exercice 3 : autour du tri par insertion

On se base sur la description suivante du tri par insertion (dans un tableau) :

```
def triInsertion(T) :
  for i in range(1, len(T)) :
    for j in range(i, 0, -1) :      # pour j de i à 1 par pas de -1
      if T[j - 1] < T[j] :
        break
      T[j - 1], T[j] = T[j], T[j - 1]
```

1. Montrer l'invariant suivant pour la boucle principale :

> *« À la fin du tour de boucle d'indice* `i`*, le sous-tableau* `T[:i+1]` *contient les mêmes éléments qu'initialement, triés en ordre croissant. »*

2. Comme vu en cours, cet algorithme fait exactement un échange par inversion du tableau initial. À combien d'affectations cela correspond-il ?

3. Comment diminuer ce nombre d'affectations ?

4. Combien de comparaisons l'algorithme ci-dessus effectue-t-il ?

5. Comment tirer parti de l'invariant montré à la question 1 pour diminuer le nombre de comparaisons effectuées ?

6. Quel est l'effet de ces améliorations sur la complexité de l'algorithme ?

Exercice 4 : opérations ensemblistes

Dans cet exercice, on représente des ensembles d'entiers par des tableaux *triés* et *sans doublon*. En vous inspirant de l'algorithme de fusion de deux listes triées, décrire des algorithmes permettant de calculer :

1. l'*union* de deux ensembles : $E \cup F = \{x \mid x \in E \text{ ou } x \in F\}$,

2. l'*intersection* de deux ensembles : $E \cap F = \{x \mid x \in E \text{ et } x \in F\}$,

3. la *différence* de deux ensembles : $E \setminus F = \{x \mid x \in E \text{ et } x \notin F\}$,

4. la *différence symétrique* de deux ensembles (ensemble formé des éléments appartenant à l'un des deux ensembles, mais pas aux deux) : $E \Delta F = \{x \mid x \in E \text{ ou (exclusif) } x \in F\}$.