Fiche d'exercices *IV*: Algorithmes de Tri

Exercice 1 (Drapeau Hollandais (Français) de Dijkstra.) *Donner un algorithme qui réorganise les éléments de trois couleurs différentes (bleue, blanche et rouge) d'un tableau de n éléments. On utilisera des permutations. Chaque couleur ne doit être testée qu'une fois et le nombre de permutations ne doit pas excéder n. Comment peut-on généraliser cette approche?*

Exercice 2 (Trier en place.) *On doit trier un tableau de n éléments qui ne peuvent être que de k types différents. Pour chaque élément, on connait son type. De plus, on connait l'ordre des k types. Donner un algorithme simple de tri en place d'un tel tableau d'éléments. Analyser sa complexité en fonction des deux paramètres n et k.*

Exercice 3 (Tri rapide.) *Combien de temps requiert le tri rapide (Hoare) pour trier n éléments identiques. Améliorer le tri rapide en tenant compte des éléments identiques.*

Exercice 4 (Tri rapide dans le cas le pire.) *Quelle est la complexité dans le cas le pire du tri rapide?*

Exercice 5 (Complexité.) *On définit* $C_1(n)$ *par* $C_1(n) = n - 1 + \frac{2}{n}\sum_{p=1}^{n} C_1(p)$ *avec* $C_1(0) = C_1(1) = 0$. *Démontrer que* $C_1(n) = 2(n+1)H_n - 4n$ *où* H_n *est l'harmonique* $H_n = \sum_{i=1}^{n} \frac{1}{i}$. *En déduire un équivalent lorsque* $n \to +\infty$.

Exercice 6 (Tri par tas – Heapsort.) *En utilisant un tas comme structure de données et l'idée du tri par sélection, donner un algorithme de tri de complexité* $\Theta(n \log n)$ *en nombre de comparaisons.*

Exercice 7 (Borne inférieure/Tri réseau.) *Montrer que tout tri ne permutant que des éléments consécutifs a une complexité en* $\Omega(n^2)$. *Donner un algorithme optimal dans ce modéle de calcul.*

Exercice 8 (Tri stable.) *On rappelle qu'un algorithme de tri est dit stable quand il préserve l'ordre originel des éléments égaux. Parmi les algorithmes de tri vue en cours, lesquels sont stables? Quel est l'interêt d'un algorithme de tri stable?*

Exercice 9 (Tri partiel.) *Donner un algorithme* A_1 *qui calcule les k premiers plus petits éléments d'un ensemble de n éléments. Étudier sa complexité. Donner un algorithme* A_2 *qui utilise comme sous-routine le tri du tableau. Comparer les complexités des algorithmes* A_1 *et* A_2*?*

Exercice 10 (Tri par compartiments.) *Écrire un algorithme de tri par sauts sans chaînage quand les éléments sont des entiers.*

Exercice 11 (Éléments distincts.) *Donner un algorithme qui détermine si tous les éléments d'un tableau sont distincts ou non. Quelle est la complexité de votre algorithme? Borne inférieure?*

Exercice 12 (Tri rapide et partition.) *Dans le tri rapide présenté en cours, la fonction* partition *teste le dépassement du tableau. En utilisant le principe de la sentinelle, comment modifier l'algorithme pour qu'il ne tienne plus compte du dépassement de tableau (indice hors domaine)?*

Exercice 13 (Tri évolué – Shell sort.) *Amélioration du tri par insertion simple proposée par D.L. Shell en 1959. On explique brièvement le principe ci-dessous:*

- *On tri tous les éléments qui sont à distance* 4 *par l'algorithme* $Tri4$*. Puis on forme des nouveaux groupes entre deux éléments consécutifs ainsi triés.*
- *On tri tous les éléments séparés par* 2 *positions en utilisant* $tri2$*. On forme les nouveaux groupes.*
- *On tri tous les éléments par un tri classique* $Tri1$*.*

Ainsi, soit on tri un petit nombre d'éléments ou des éléments relativement bien ordonnés.
Trier la liste suivante en utilisant ce principe:

$$44, 55, 12, 42, 94, 18, 06, 67$$

Donner l'algorithme correspondant à une suite d'incrément (telle que $h_t = 1$*). Knuth proposa la suite d'incréments suivante:* $h_{k-1} = 2h_k + 1$ *et* $t = \lfloor \log_2 n \rfloor - 1$*. Pour cette suite d'incréments, on peut montrer la complexité en* $O(n^{1.2})$ *du tri.*