École normale supérieure **2012-2013**
Département d'informatique

Algorithmique et Programmation
TD n° 2 : Tris et hachage

Exercice 1. Construction d'un tas

Montrer qu'on peut construire un tas, à partir d'un tableau non-ordonné de taille n, en complexité $O(n)$.

Exercice 2. Algorithme efficace en moyenne pour le médian

1. Donner un algorithme pour trouver le médian d'un tableau en complexité $O(n \log n)$.
2. Donner une variante de l'algorithme du tri rapide pour trouver le médian.
3. Résoudre la récurrence
$$T(n) = T(\frac{9n}{10}) + O(n).$$
4. Donner la complexité de l'algorithme de la question 2 au pire des cas et en moyenne (pour la complexité moyenne, on ne demande pas de preuve).

Exercice 3. k-ième élément d'un tableau

Nous allons utiliser une variante du tri rapide pour déterminer le k-ième élément d'un ensemble de n éléments. Nous supposons que les n clés sont distinctes deux à deux.

1. Décrire une procédure Sélectionner qui détermine le k-ième élément du tableau.
2. Montrer que que sans hypothèse particulière sur l'algorithme de pivotage, la fonction Sélectionner peut réaliser $O(n^2)$ comparaisons.
3. Montrer que si l'algorithme de pivotage garantit que la taille du sous-tableau de l'appel récursif ne dépasse pas αn où $\alpha < 1$, la complexité en nombre de comparaisons de la fonction Sélectionner est $O(n)$.

On considère l'algorithme de choix du pivot suivant :

- Découper le tableau en $n/3$ blocs $\{B_1, \ldots, B_{n/3}\}$ de trois éléments ;
- Déterminer les éléments médians m_k des B_k, $k \in \{1, \ldots, n/3\}$;
- Utiliser l'algorithme récursivement pour déterminer l'élément médian p de la liste $m_1, \ldots, m_{n/3}$. Déterminer le rang de p dans le tableau, puis utiliser l'algorithme récursivement sur une partie du tableau.

4. Montrer que le pivot choisi est strictement supérieur à au moins $n/3 - O(1)$ éléments de t et est inférieur ou égal à au moins $n/3 + O(1)$ éléments de t.

5. Donner la complexité de la fonction SÉLECTIONNER.
6. Que devient la récurrence si, pour le choix du pivot, on découpe le tableau dans des blocs de 5 el'ements plutôt que de 3 ? Montrer que la complexité est alors meilleure.

Exercice 4. Maximum & Minimum

1. Écrire un algorithme naïf qui calcule le minimum et le maximum sur un tableau de n éléments et donner un équivalent du nombre de comparaisons au pire des cas.

2. Est-il possible de réduire le nombre de comparaisons à faire ? Décrire un algorithme avec moins de comparaisons au pire des cas.
3. Montrer une borne inférieure de n comparaisons.
4. Montrer une meilleure borne inférieure sur le nombre de comparaisons à effectuer.
   Indication : On pourra utiliser la méthode de l'*adversaire*.
   Soit $\mathcal{A}$ un algorithme qui trouve le maximum et le minimum. Donner une stratégie d'un adversaire qui choisit les réponses aux comparaisons de façon à obliger $\mathcal{A}$ à faire plus de comparaisons.