IFT436 - Série d'exercices #3 : le tri

Manuel Lafond

Exercice 1: Considérez le célèbre tri à bulle (Bubblesort):

```
fonction triABulle(T) //n est le nombre d'éléments
    pour i = 1..n − 1 faire
        pour j = 1..n − i faire
            si T[j] > T[j + 1] alors
                échanger T[j] avec T[j + 1] ;
            fin
        fin
    fin
```

Répondez aux questions suivantes:

a. Démontrez que cet algorithme est correct, c'est-à-dire que le tableau est trié à la fin de l'exécution.

   Indice: trouvez un invariant de boucle qui est maintenu par l'algorithme. Regardez la fin du tableau par rapport à i.

b. Quelle est le temps de cet algorithme en notation Θ, dans le pire cas?

Exercice 2: La *médiane* m d'un ensemble S de n valeurs est la valeur de S qui "sépare" en deux: la moitié des valeurs de S sont au plus m, et la moitié sont au moins m. En d'autres termes, la médiane de S est la $\lfloor n/2 \rfloor$-ième plus grande valeur de S.
Décrivez d'abord un algorithme qui trouve la médiane de S en temps $O(n \log n)$.

Ensuite, bien que ce ne soit pas simple, il est possible de trouver la médiane en temps $O(n)$ (n'essayez pas ça!). Montrez que ceci implique qu'on peut implémenter QuickSort en temps $O(n \log n)$ dans le pire cas.

Exercice 3: Lorsqu'un nouveau maire est élu dans une ville, sa première tâche est toujours de vérifier qu'il n'y a pas de maisons qui sont trop rapprochées sur une même rue. On peut résoudre ce problème en trouvant, sur une rue donnée, la distance minimale entre deux maisons (on traite les maisons comme des points en une dimension).
Étant donné un ensemble $M = \{x_1, x_2, \ldots, x_n\}$ d'entiers, on veut donc trouver $x_i, x_j \in S$ tels que $|x_i - x_j|$ est minimum parmi toutes les paires d'éléments de M. Vous pouvez représenter M comme bon vous semble, par exemple vous pouvez supposer que M est un tableau.
Donnez un algorithme qui résout ce problème. Donnez la complexité de votre algorithme avec la notation O, avec justification. Faites mieux que $O(n^2)$.

Exercice 4: Soit T un tableau. On dit que les positions i et j forment une *inversion* si $i < j$ mais $T[i] > T[j]$ (donc dit autrement, si $T[i]$ et $T[j]$ sont relativement mal placés). Notez qu'une inversion n'est *pas* une opération - elle n'échange pas des éléments.
Reconsidérez le tri par insertion. Dans la boucle imbriquée, un *décalage* est fait lorsque l'on applique l'instruction $T[i+1] = T[i]$. Montrez que le nombre de décalages fait par l'algorithme sur entrée T est égal au nombre d'inversions de T.

Exercice 5: Rappelons qu'un algorithme est *sur-place* s'il trie un tableau T en déplaçant directement les éléments dans T. Un algorithme de tri est *stable* si, dans le tableau retourné, les éléments de valeur égale apparaissent dans le même ordre que dans le tableau donné en entrée.
Pour chacun des algorithmes suivants, dites si l'implémentation présentée dans les notes de cours est (1) sur-place, et (2) stable.

- tri par insertion
- tri fusion
- QuickSort
- triComptage
- tri radix

Exercice 6: En réponse partielle à l'exercice précédent, la version de Quick-Sort présentée en classe et dans les notes n'est pas sur-place. Proposez une

version sur-place du QuickSort.
Indice: pour travailler récursivement sur des sous-tableaux de T, il vous faudra deux paramètres low et hi pour les positions du sous-tableau sur lequel vous travaillez. L'en-tête de votre fonction pourrait donc avoir la forme $quickSort(T, low, hi)$.

Exercice 7: Petit rappel IFT339: un arbre binaire de recherche (ABR) est une structure de données qui permet d'effectuer: l'insertion d'une valeur, la suppression d'une valeur, et la recherche de la valeur minimum. Chaque opération peut se faire en temps $O(h)$, où h est la hauteur de l'arbre (voir série exercices 2 pour définition de hauteur). On sait qu'il y a des techniques pour garantir que dans un ABR avec n éléments, on a toujours $h \in O(\log n)$ (par exemple, arbre rouge-noir, arbre AVL, B-tree, ...).
Donnez un algorithme de tri qui utilise un ABR. Quelle est la meilleure complexité que vous pouvez atteindre?