TD10 : Algorithmes à pivots

1 Pivot

Réordonner un tableau en fonction d'un pivot est une opération classique utilisée dans de nombreux algorithmes. Étant donné un tableau t, et un pivot p, nous voulons réorganiser les valeurs du tableau en deux parties. La première partie contient toutes les valeurs du tableau qui sont plus petites que p. La seconde les valeurs plus grandes.

a) Étant donné le tableau suivant, et un pivot de valeur 7, réorganisez le tableau en fonction du pivot. Notez que la solution n'est pas unique.

10	2	12	3	6	11	9	1	13	4	5	8

L'un des intérêts de cette solution est qu'elle peut se faire en place sans nécessiter de tableau supplémentaire, en ne réalisant que des échanges de valeurs dans le tableau. Vous aurez par contre besoin de stocker et faire évoluer plusieurs indices sur le tableau lors de l'algorithme.

b) En discutant avec vos voisins, essayez de trouver quels indices stocker, et comment obtenir le tableau pivoté via une séquence d'échanges.

c) Quelle est la complexité de votre solution ?

d) Modifiez votre algorithme pour qu'il ne traite qu'une portion du tableau entre deux indices.

e) Modifiez votre algorithme pour que tous les éléments égaux au pivot finissent entre les deux zones.

2 Quicksort

Le *quicksort* (tri rapide) est un algorithme de tri de type *diviser pour régner* fondé sur le pivot. Cet algorithme consiste à déterminer une valeur pivot parmi les valeurs du tableau. Le tableau est ensuite pivoté, et la valeur choisie comme pivot est alors à sa place finale dans le tableau. Les parties gauche et droite du tableau sont ensuite triées indépendamment. Une version basique de l'algorithme est la suivante :

```
Procédure quicksort(t, début, fin)
    if fin − début > 1 then
        p ← t[début]
        mid ← pivot(t, p, début, fin)
        quicksort(t, début, mid)
        quicksort(t, mid +1, fin)
```

f) Exécutez l'algorithme sur le tableau suivant.

5	1	7	4	3	6	2

g) Quelle est la complexité de cet algorithme dans le pire cas ? le meilleur ?

h) À votre avis, dans des cas d'utilisation réels, ce pire cas arrive-t-il fréquemment ?

i) Avez-vous des idées pour améliorer la complexité dans le pire des cas ?

3 Sélection

Le but de l'algorithme quickselect est de renvoyer la k$^{\text{ième}}$ plus petite valeur d'un tableau.

j) Décrivez un algorithme capable de calculer les k plus petites valeurs d'un tableau, en remplissant un tableau de taille k fourni en paramètre.

k) Quelle est la complexité de cet algorithme en fonction de k et n dans le meilleur et le pire cas ?

Lorsque seul le k$^{\text{ième}}$ élément est recherché et k grandit, cette complexité peut devenir prohibitive. Dans ce cas, il est possible d'utiliser un algorithme de type diviser pour régner assez proche du quicksort : *quickselect*. Cet algorithme utilise également un pivot pour ordonner partiellement le tableau, et orienter sa recherche.

l) Essayez de trouver cet algorithme.

m) Quelle est sa complexité dans le meilleur et le pire cas ?