

Tri rapide

I. Définition

Le tri rapide ou tri pivot (en anglais quicksort) est un algorithme de tri fondé sur la méthode de conception diviser pour régner.

La complexité moyenne du tri rapide pour n éléments est $O(n \log n)$, ce qui est optimal pour un tri par comparaison, mais la complexité dans le pire des cas est $O(n^2)$. Malgré ce désavantage théorique, c'est en pratique un des tris les plus rapides, et donc un des plus utilisés. Le pire des cas est en effet peu probable lorsque l'algorithme est correctement mis en œuvre.

II. Description de l'algorithme

La méthode consiste à placer un élément de la liste (appelé pivot) à sa place définitive, en permutant tous les éléments de telle sorte que tous ceux qui sont inférieurs au pivot soient à sa gauche et que tous ceux qui sont supérieurs au pivot soient à sa droite.

Cette opération s'appelle le partitionnement. Pour chacune des sous-listes, on définit un nouveau pivot et on répète l'opération de partitionnement. Ce processus est répété récursivement, jusqu'à ce que l'ensemble des éléments soit trié.

Concrètement, pour partitionner une sous-liste :

- le pivot est placé à la fin, en l'échangeant avec le dernier élément de la sous-liste;
- tous les éléments inférieurs au pivot sont placés en début de la sous-liste ;
- le pivot est déplacé à la fin des éléments déplacés.

Pour faciliter l'algorithme, on choisira le pivot comme le dernier élément de la sous-liste. Comme ça nous n'avons pas à le déplacer.

III. Exemple

On souhaite trier la liste suivante : [3, 7, 8, 5, 2, 1, 9, 5, 4]

<u>[3, 7, 8, 5, 2, 1, 9, 5, 4]</u> → [3, 2, 1, 4, 7, 8, 9, 5, 5]	Le pivot est 4 et on parcourt la liste pour placer en début de liste les éléments plus petits ou égaux à 4. A la fin, 4 est à sa place définitive.
[3, 2, 1, 4, 7, 8, 9, 5, 5] → [1, 2, 3, 4, 5, 5, 9, 7, 8]	On va appliquer le même algorithme sur deux sous listes avant et après 4. Les pivots sont en gras et après le partitionnement, ils sont à leur place définitive. Il nous reste trois sous liste à trier, mais la deuxième n'a qu'un seul élément donc elle est déjà triée.
[1, 2, 3, 4, 5, 5, 9, 7, 8] → [1, 2, 3, 4, 5, 5, 9, 8, 7]	Les pivots sont en gras et après le partitionnement, ils sont à leur place définitive. Les deux dernières sous-listes n'ont qu'un seul élément, donc elles sont triées et tout est trié.

IV. Pseudo-code

```
partitionner(liste L, entier premier, entier dernier)
    j := premier
    pour i de premier à dernier
        si L[i] <= L[dernier] alors
            échanger L[i] et L[j]
            j := j + 1
    renvoyer j-1

tri_rapide(liste L, entier premier, entier dernier)
    si premier < dernier alors
        pivot := partitionner(L, premier, dernier)
        tri_rapide(L, premier, pivot-1)
        tri_rapide(L, pivot+1, dernier)
```

Coder cette algorithme en Python et tester le.