Chapitre 10 :
Algorithmes de tri 1

I Généralités

1.1 Introduction

Un algorithme de tri est un algorithme qui trie des données selon un ordre donné, l'ordre par défaut étant l'ordre croissant.

Il existe de nombreux algorithmes de tri qui ont des coûts différents mais aussi des propriétés différentes. Les coûts des algorithmes de tri seront étudiés dans un autre chapitre mais nous nous intéresserons ici à leurs propriétés.

1.2 Tri comparatif

Un algorithme de tri est dit comparatif s'il compare deux à deux certains éléments et modifie les données en fonction de l'élément le plus grand.

La majorité des algorithmes de tri sont comparatifs. Les algorithmes de tris non comparatifs nécessitent d'avoir des informations particulières comme, par exemple, la connaissance d'un majorant.

1.3 Tri stable

Un algorithme de tri est dit stable si, lorsque deux éléments sont égaux, l'algorithme ne modifie pas leur ordre.

Cette propriété permet d'éviter de faire des opérations inutiles sur des éléments qui n'ont pas à être triés.

1.4 Tri en place

Un algorithme de tri est dit en place si la mémoire utilisée est constante, autrement dit si on ne crée pas de copie des données à trier. Ainsi, dans un algorithme en place, les données initiales sont perdues car elles ont été modifiées par l'algorithme.

Certains algorithmes de tri peuvent être écrits de façon à être en place ou non alors que pour d'autres une des deux versions est plus naturelle.

II Un exemple : le tri à bulles

2.1 Principe du tri à bulles

L'algorithme du tri à bulles est simple mais assez coûteux.
Le principe de ce tri est de faire remonter les éléments les plus grands à la fin de la liste, comme des bulles qui remontent à la surface d'un liquide.
Pour cela, on va comparer les éléments 2 à 2 et les échanger si besoin.

On considère la liste $l=[1,7,9,5]$. Cette liste est de longueur 4.

- On considère les 4 premiers éléments de la liste.
  - $[①,⑦,9,5]$ comme $1 \leq 7$, on ne fait rien,
  - $[1,⑦,⑨,5]$ comme $7 \leq 9$, on ne fait rien,
  - $[1,7,⑨,⑤]$ comme $9 > 5$, on échange.
- On obtient $[1,7,5,9]$. Le dernier élément est trié, on ne considère que les 3 premiers éléments de la liste.
  - $[①,⑦,5,9]$ comme $1 \leq 7$, on ne fait rien,
  - $[1,⑦,⑤,9]$ comme $7 > 5$, on échange.
- On obtient $[1,5,7,9]$. Les 2 derniers éléments sont triés, on ne considère que les 2 premiers éléments de la liste.
  - $[①,⑤,7,9]$ comme $1 \leq 5$, on ne fait rien.

  On a obtenu la liste triée qui est : $[1,5,7,9]$.

2.2 Algorithme

```
1 def TriBulles(l) :
2     n=len(l)
3     for i in range(n-1) :
4         for j in range(0,n-i-1) :
5             if l[j]>l[j+1] :
6                 l[j],l[j+1]=l[j+1],l[j]
7     return l
```

3 Pour une liste de longueur n, il y a $n-1$ éléments à faire remonter, l'élément restant étant nécessairement bien placé.

4 On va faire remonter l'élément $n-i-1$.

5 et 6 Si l'ordre croissant n'est pas respecté, on procède à un échange.

2.3 Propriétés

La fonction TriBulles est :

- comparative : la ligne 5 effectue une comparaison,
- stable : dans la ligne 5, l'inégalité est stricte ainsi deux éléments égaux ne seront jamais échangés,
- en place : seule la liste en argument de la fonction est stockée en mémoire.