Les méthodes de Tris et de recherche

Loubna EL FAQUIH
TDI 1A
ISMONTIC 2016/2017

Les algorithmes de Tri

- Tris élémentaires
  - Le tri par sélection
  - Le tri par insertion
  - Le tri a bulles
- Tris avancés
  - Tri rapide
  - …

Tri par sélection

Méthode :

on cherche l'élément de plus petite valeur pour l'échanger avec l'élément en première position ;

puis on cherche l'élément ayant la deuxième plus petite valeur pour l'échanger avec l'élément en deuxième position ;

et ainsi de suite.

20	6	1	3	1	7
1	6	20	3	1	7
1	1	20	3	6	7
1	1	3	20	6	7
1	1	3	6	20	7
1	1	3	6	7	20

Il faut :

- 1 boucle pour parcourir le tableau et sélectionner tous les éléments ;
- 1 boucle pour rechercher le minimum parmi les éléments non triés.

Tri par sélection : Algorithme

- Supposons que le tableau est noté T et sa taille N

```
Pour i De 0 A N-2 Faire
  indice ← i
  Pour j De i+1 A N-1 Faire
    Si (T[j] < T[indice]) Alors
      indice ← j
    Finsi
  FinPour
  temp ← T[indice]
  T[indice] ← T[i]
  T[i] ← temp
FinPour
```

Recherche de l'élément concerné

Permutation des deux valeurs

Tri par insertion

Méthode :

on considère les éléments les uns après les autres en insérant chacun à sa place parmi les éléments déjà triés.

20	6	1	3	1	7
6	20	1	3	1	7
1	6	20	3	1	7
1	3	6	20	1	7
1	1	3	6	20	7
1	1	3	6	7	20

Il faut :

- 1 boucle pour parcourir le tableau et sélectionner l'élément à insérer ;
- 1 boucle pour décaler les éléments plus grands que l'élément à insérer ;
- insérer l'élément.

Tri par insertion : Algorithme

- Supposons que le tableau est noté Tab et sa taille N

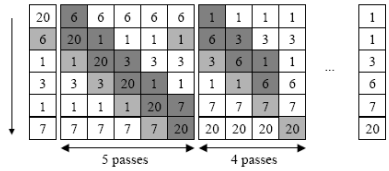
```
Pour i De 1 A N-1 Faire
  mem ←Tab[i]
  j ← i
  TantQue ( j > 0 et Tab[j-1] > mem )
    Tab[j] ← Tab[j-1]
    j ← j-1
  FTQ
  Tab[j] ← mem
FinPour
```

Tri à bulles

Méthode :

on parcours autant de fois le tableau en permutant 2 éléments adjacents mal classés qu'il le faut pour que le tableau soit trié.

Tableau prit dans ce sens



Il faut :

- 1 boucle pour parcourir tout le tableau et sélectionner les éléments un à un ;
- 1 boucle pour permuter les éléments adjacents.

7

Tri à bulles : Algorithme

- Supposons que le tableau est noté T et sa taille N

```
Permut ← 1
TantQue (Permut = 1)
  Permut ← 0
  Pour i De 0 A N-2 Faire
    Si (T[i] > T[i+1]) Alors
      temp ← T[i]
      T[i] ← T[i+1]
      T[i+1] ← temp
      Permut ← 1
    FinSi
  FinPour
FTQ
```

8

Tri à bulles : Algorithme

- Supposons que le tableau est noté T et sa taille N

```
Permut ← Vrai
TantQue (Permut)
  Permut ← Faux
  Pour i De 0 A N-2 Faire
    Si (T[i] > T[i+1]) Alors
      temp ← T[i]
      T[i] ← T[i+1]
      T[i+1] ← temp
      Permut ← Vrai
    FinSi
  FinPour
FTQ
```

9

Tri par comptage

- Le tri par comptage consiste pour chaque élément du tableau a compter combien d'éléments sont plus petits que lui, grâce a ce chiffre on connait sa position dans le tableau résultat.
  - Liste initiale :

    52 10 1 25 62 3 8 55 4 23
  - Poids du mot

    7 4 0 6 9 1 3 8 2 5
  - Position

    8 5 1 7 10 2 4 9 3 6

10

Exercice

Ecrire l'algorithme de Tri par comptage

Pour réaliser ce tri, utilisez plusieurs tableaux

(Prenez garde aux valeurs identiques !)

11

Solution

```
//Remplissage du tableau Poids par les positions
Pour i de 0 a N-1 Faire
    Poids[i] ← 1
    Pour j de 0 a N-1 Faire
        Si (T[j] < T[i]) alors
        Poids[i] ← Poids[i] + 1
        Fin si
    Fin pour
Fin pour
//Elimination des positions doubles du tableau Poids
Pour i de 0 a N-1 Faire
    Pour j de 0 a N-1 Faire
    Si ((Poids[i] = Poids[j])  et (i <> j)) alors
        Pour s de 0 a N-1 Faire
            Si (Poids[j] < Poids[s])
                Poids[s] ← Poids[s] + 1
            Fin si
        Fin pour
    Poids[j] ← Poids[j] + 1
    Fin si
    Fin pour
    Fin pour
//Tri des éléments dans le tableau temporaire X
Pour i de 0 a N-1 Faire
    X[ Poids[i] ] ← T[i]
Fin pour
//Copie des éléments triés dans T
Pour i de 0 a N-1 Faire
    T[i] ← X[i]
Fin pour
```

12

Les méthodes de Recherche Rapide

Recherche séquentielle

- Le principe de l'algorithme de recherche séquentielle consiste à parcourir un tableau d'éléments dans l'ordre de ses indices jusqu'à ce qu'un élément recherché soit trouvé ou bien que la fin du tableau soit atteinte et le rang (indice) de l'élément est alors retourné.

14

Algorithme de recherche séquentielle

```
entier Fonction RechSéq(T : Tableau d'entiers, N : entier, ValRech : entier)
    Variable i : entier
    Début
        Pour (i de 0 à N-1) Faire
            Si (T[i] = ValRech) Alors
                Retourner i
            FinSi
        FinPour
        Retourner -1
    Fin
```

Description

L'algorithme ci-dessus est une fonction qui prend en entrée, un tableau T d'entiers non triés de N éléments et un entier ***ValRech*** qui est la valeur recherché.
La boucle ***Pour*** consiste à parcourir le tableau en comparant la valeur recherchée avec chaque élément du tableau.
Si la valeur est trouvée, l'instruction ***Retourner i*** interrompt la fonction. Dans ce cas la fonction renvoie la valeur de ***i***, c'est à dire l'indice de l'élément recherché.
Lorsque la valeur recherchée n'est pas trouvée dans le tableau, la fonction renvoie ***-1***.

15

Recherche dichotomique

- L'algorithme de recherche dichotomique est un algorithme efficace et rapide pour la recherche d'un élément dans un tableau **trié**. Le principe de cet algorithme consiste à diviser le tableau en deux parties et à comparer la valeur recherchée avec l'élément situé au milieu du tableau (l'élément médian).

  Si la valeur recherchée correspond à l'élément médian, alors l'algorithme s'arrête et renvoie l'indice de cet élément. Sinon l'algorithme va répéter la recherche uniquement dans la partie qui peut contenir la valeur recherchée.

16

Algorithme de recherche dichotomique

```
entier Fonction RechDich( T: tableau d'entiers, N : entier, ValRech : entier)
    Variables D, F, M : entier
    D ← 1
    F ← N
    Début
        TantQue (D <= F) Faire
            M ← Ent ((D + F) / 2) //Ent : c'est la fonction partie entière, équivalent a :  M ← (D + F) DIV 2
            Si (T[M] = ValRech) Alors
                Retourner M
            Sinon
                Si (T[M] < ValRech) Alors
                    D ← M + 1
                    Si (T[D] = ValRech) Alors
                        Retourner D
                    FinSi
                Sinon
                    F ← M - 1
                    Si (T[F] = ValRech) Alors
                        Retourner F
                    FinSi
                FinSi
            FinSi
        FTQ
    Retourner -1
    Fin
```

17

Algorithme de recherche dichotomique

1	2	3	4	5	6	7	8	9	10	11	12
1	5	6	9	12	15	19	28	35	42	60	80
1	5	6	9	12	15	19	28	35	42	60	80
1	5	6	9	12	15	19	28	35	42	60	80

18

Algorithme de recherche dichotomique

- **Description** :
  L'algorithme ci-dessus est une fonction qui prend en entrée, un tableau T d'entiers trié, la taille de ce tableau N et un entier ***ValRech*** qui est la valeur recherchée.
  Les variables D et F correspondent aux indices respectivement début et fin de l'intervalle de recherche dans le tableau. M c'est l'indice médian de l'intervalle de recherche.
  Le tableau ci-dessous montre le fonctionnement de cet algorithme. La valeur recherchée est 35.

- Chaque ligne indique l'intervalle de recherche dans le tableaux à chaque itération de la boucle ***TantQue***. La cellule grise représente l'élément médian de l'intervalle de recherche.
  La première ligne indique la position initiale avant le démarrage de l'algorithme. Dans la ligne suivante, l'algorithme compare le nombre 15 qu'est l'élément médian à la valeur 35 (15 < 35). Dans ce cas l'algorithme continue la recherche entre les indices M +1 et F.
  Lors de l'itération suivante la valeur recherchée a été trouvée, la fonction s'interrompt et renvoie l'indice de cet élément.
  Cette fonction dispose de quatre sorties possibles : trois à l'intérieur de la boucle, si la valeur est trouvée et une à la fin si elle n'est pas trouvée. Par convention la valeur -1 est retournée si la valeur n'est pas trouvée.

  Remarque :
  Pour utiliser la recherche dichotomique, il faut tout d'abord trier le tableau en utilisant l'un des algorithmes de tri.