Algorithmes de tri

Dr Khadim DRAME
kdrame@univ-zig.sn

Département d'Informatique
UFR des Sciences et Technologies
Université Assane Seck de Ziguinchor

2 décembre 2021

Plan

Introduction

- Un **algorithme de tri** est une suite finie d'opérations consistant à ordonner un ensemble d'éléments selon un ordre donné.
- Le tri est l'un des domaine de l'algorithmique les plus étudiés.
- Le tri est l'une des tâches les plus fréquentes effectuées par les ordinateurs.
- Les algorithmes de tri sont utilisés pour mettre en évidence certains concepts
  - complexité d'un algorithme,
  - algorithme de recherche,
  - etc.

Introduction

- Le tri a une grande utilité dans la pratique.
- Il est préalable à la résolution efficace de certains problèmes.
- Exemples
  - recherche d'un élément dans un tableau trié (dichotomie) ;
  - recherche du $i^{ème}$ élément le plus grand ;
  - résultats d'un moteur de recherche ;
  - calcul de la médiane d'une séquence de données ;

Plan

Tri par sélection

- Le principe du tri par sélection est simple
  - recherche du plus petit élément ;
  - permutation avec le premier élément ;
  - tri du reste du tableau avec le même principe.
- Il n'est pas efficace sur des tableaux de grande taille.

Tri par sélection

tableau initial	8	3	9	5	**2**
1^{er} passage	2	**3**	9	5	8
$2^{ème}$ passage	2	3	9	**5**	8
$3^{ème}$ passage	2	3	5	9	**8**
$4^{ème}$ passage	2	3	5	8	9

Tri par sélection

```
1  procedure tri_selection(var T: tab; N: integer);
2  var i, j, imin: integer;
3    temp : integer; {même type que éléments du tableau}
4  begin
5    for i:=1 to N-1 do
6    begin
7      imin:=i;
8      for j:=i+1 to N do
9        if T[j] < T[imin] then
10         imin:=j;
11     if i <> imin then
12     begin {échanger T[i] et T[imin]}
13       temp:=T[i];
14       T[i]:=T[imin];
15       T[imin]:=temp;
16     end;
17   end;
18 end;
```

Tri à bulles (bubble sort)

- Le tri à bulles est l'un des algorithmes de tri les plus simples.
- Le principe du tri à bulles est simple
  - passage sur chaque élément et comparaison avec son suivant ;
  - permutation des deux éléments si nécessaire ;
  - répétition du processus jusqu'à ce que tous les éléments soient ordonnés.
- Il n'est pas efficace sur des tableaux de grande taille.

Tri à bulles

tableau initial	8	3	9	5	2
1^{er} passage	3	8	5	2	**9**
$2^{ème}$ passage	3	5	2	**8**	**9**
$3^{ème}$ passage	3	2	**5**	**8**	**9**
$4^{ème}$ passage	2	**3**	**5**	**8**	**9**

Tri à bulles

```
1  procedure tri_bulles(var T: tab; N: integer);
2  var
3    i, j: integer;
4    temp : integer; {même type que éléments du tableau}
5  begin
6    for i:=N downto 2 do
7      for j:=1 to i-1 do
8        if T[j+1] < T[j] then
9        begin {échanger T[j] et T[j+1]}
10         temp:=T[j];
11         T[j]:=T[j+1];
12         T[j+1]:=temp;
13       end;
14 end;
```

Tri par insertion (insertion sort)

- Le tri par insertion fait partie des algorithmes de tri les plus simples.
- Le principe du tri par insertion est très intuitif
  - on passe sur chaque élément et l'insère à la bonne place ;
  - pour cela, l'élément à insérer est comparé aux éléments déjà triés sur la gauche.
- Ses performances sont bonnes sur de petits tableaux, moins bonnes sur des tableaux de grande taille.

Tri par insertion

tableau initial	8	**3**	9	5	2
$1^{ère}$ insertion	3	8	**9**	5	2
$2^{ème}$ insertion	3	8	9	**5**	2
$3^{ème}$ insertion	3	5	8	9	**2**
$4^{ème}$ insertion	2	3	5	8	9

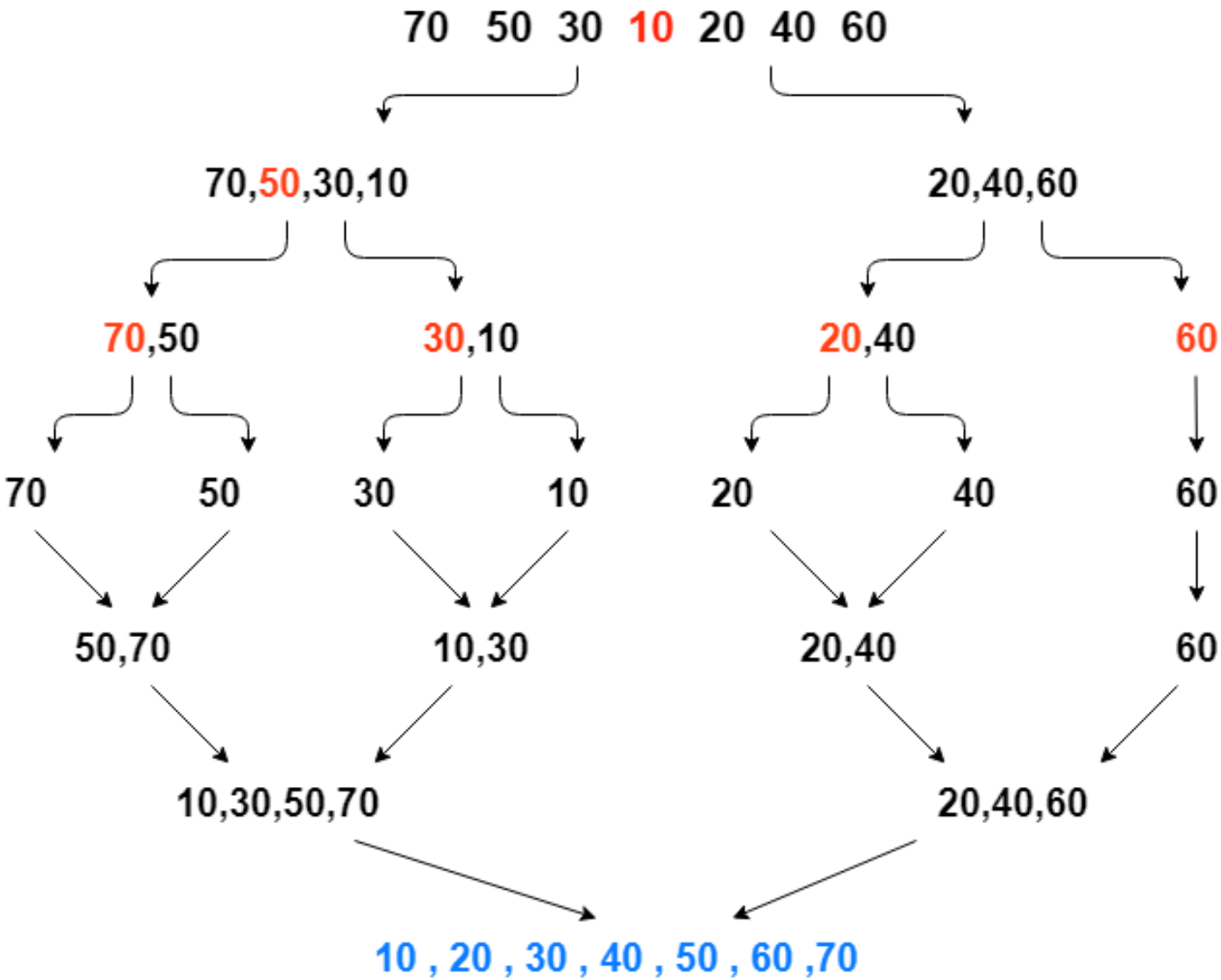
Tri par insertion

```
1  procedure tri_insertion(var T: tab; N: integer);
2  var
3    i, j: integer;
4    X : integer; {même type que éléments du tableau}
5  begin
6    for i:=2 to N do
7    begin
8      X:=T[i];
9      j:=i;
10     while (j > 1) and (T[j-1] > X) do
11     begin
12       T[j]:=T[j-1];
13       j:=j-1;
14     end;
15     T[j]:=X;
16   end;
17 end;
```

Tri par fusion (merge sort)

- Le tri par fusion est un peu plus complexe que les algorithmes de tri précédents.
- Le principe du tri par fusion est diviser pour régner
  - division l'ensemble des données à trier en deux parties ;
  - réitération sur les parties (jusqu'à ce qu'elles contiennent un seul élément) ;
  - fusion des parties triées pour constituer le tableau trié.
- Il a de bonnes performances notamment sur des tableaux de grande taille.

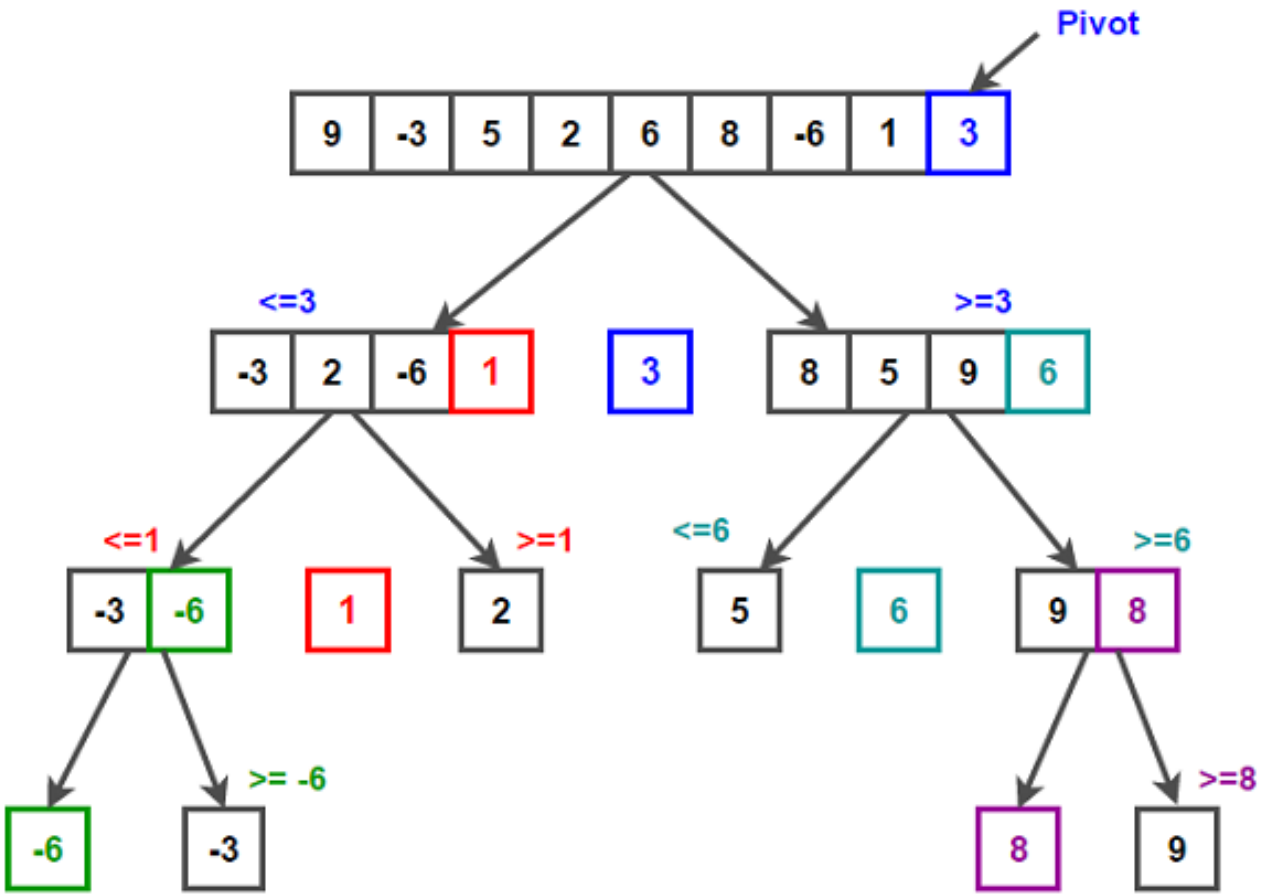
Tri par fusion



Tri rapide (quicksort)

- Le tri rapide est aussi appelé tri de **Hoare** ou tri pivot.
- Il est plus complexe que les algorithmes de tri précédents.
- Le principe du tri rapide est : diviser pour régner
  - choix d'un élément (de l'ensemble des données) comme un pivot ;
  - une fois qu'on a ce pivot
    - placement des éléments inférieurs au pivot d'un côté ;
    - placement des éléments supérieurs au pivot d'un autre côté.
  - réitération sur chaque partie(jusqu'à ce qu'elles contiennent un seul élément).
- C'est l'un des algorithmes de tri les plus efficaces.

Tri rapide



Plan

Exercices d'application

- Exercice 1
  Écrire un programme qui permet de calculer la médiane d'une suite de valeurs réelles.

Exercices d'application

- Exercice 1
  Écrire un programme qui permet de calculer la médiane d'une suite de valeurs réelles.

```
1  program calcul_mediane;
2  var
3    t : array[1..10] of real;
4    i, N : integer;
5    med : real;
6  begin
7    {saisie des données du tableau}
8    N:=length(t);
9    tri_selection(t,N);
10   if N mod 2 == 0 then
11     med:=(t[N div 2] + t[N div 2 + 1])/2
12   else
13     med:=t[N div 2 + 1];
14 writeln('La médiane est : ', med);
15 end.
```

Exercices d'application

- Exercice 2

  Écrire un programme qui permet de fusionner deux tableaux triés pour former un tableau trié.

Exercices d'application

```
1  program fusion_tableaux_tries;
2  var
3    t1, t2 : array[1..10] of real;
4    t : array[1..20] of real;
5    i1, i2, i, N1, N2 : integer;
6  begin
7    {saisie des données des tableaux}
8    i:=1;
9    i1:=1; N1:=10;{length(t1);}
10   i2:=1; N2:=10;{length(t2);}
11   while (i1 <= N1) and (i2 <= N2) do
12   begin
13     if t1[i1] <  t2[i2] then
14     begin
15       t[i]:=t1[i1];
16       i1:=i1+1;
17     end
18
```

Exercices d'application

```
1   else {suite du code}
2     begin
3       t[i]:=t2[i2];
4       i2:=i2+1;
5     end;
6     i:=i+1;
7   end; {fin de la boucle while}
8   while i1 <= N1 do {copier le reste de t1}
9   begin
10    t[i]:=t1[i1];
11    i1:=i1+1;
12    i:=i+1;
13  end;
14  while i2 <= N2 do {copier le reste de t2}
15  begin
16    t[i]:=t2[i2];
17    i2:=i2+1;
18    i:=i+1;
19  end;
```

Résumé

- Le tri est une opération communément utilisée dans la pratique.
- Le tri est fondamental dans l'étude de certains concepts de l'informatique (complexité).
- Plusieurs algorithmes de tri sont proposés : tris par sélection, à bulles, par insertion, par fusion, rapide...
- Des algorithmes basées sur des idées différentes.
- Des algorithmes ayant des complexités différentes.