

CLASSEMENT

UMLV ©

Liste

$L = (7, 3, 1, 4, 8, 3)$

↓ classement, tri

$L = (1, 3, 3, 4, 7, 8)$

Liste classée (en ordre croissant)

Tri interne

éléments en table, liste chaînée. En mémoire centrale

Tri externe

éléments en fichiers

Opérations élémentaires

comparaison de deux éléments

échange

sélection de places

51

TRIS INTERNES

UMLV ©

Elémentaires

Sélection

Insertion [Shell] $O(n^2)$

Bulles

Dichotomiques

Tri rapide

Fusion $O(n \log n)$

Par structures

Arbre (équilibré)

Tas $O(n \log n)$

Rangement

Tri lexicographique

Partitionnement - rangement ~ linéaires, $O(n)$

52

CLES

UMLV ©

Guy	1m75	60 k
Anne	1m70	55 k
Lou	1m75	57 k
Luc	1m72	61 k

CLE = TAILLE

Anne	1m70	55 k
Luc	1m72	61 k
Guy	1m75	60 k
Lou	1m75	57 k

CLE = POIDS / TAILLE

Anne	1m70	55 k
Lou	1m75	57 k
Guy	1m75	60 k
Luc	1m72	61 k

CLE = (TAILLE, POIDS)

Anne	1m70	55 k
Luc	1m72	61 k
Lou	1m75	57 k
Guy	1m75	60 k

53

TRI en TABLE

UMLV ©

$L = (e_1, e_2, \dots, e_n)$ en table, accès direct à e_i
Clé : Élément \rightarrow Ensemble muni de l'ordre \leq

Problème

Calculer p , permutation de $\{1, 2, \dots, n\}$
telle que $Clé(e_{p(1)}) \leq Clé(e_{p(2)}) \leq \dots \leq Clé(e_{p(n)})$

Rang

$p^{-1}(i)$ est le rang de l'élément i dans la suite classée

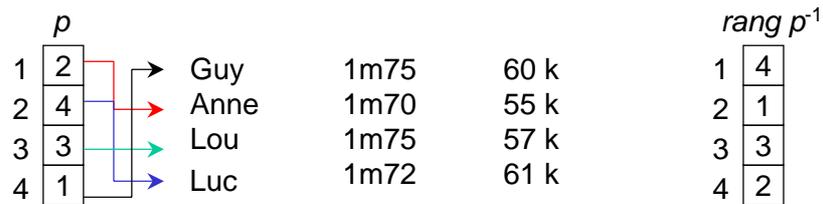
Stabilité

p est stable, chaque fois que $Clé(e_{p(i)}) = Clé(e_{p(k)})$:
 $i < k$ équivalent $p(i) < p(k)$
[le tri n'échange pas les éléments de même clé]

54

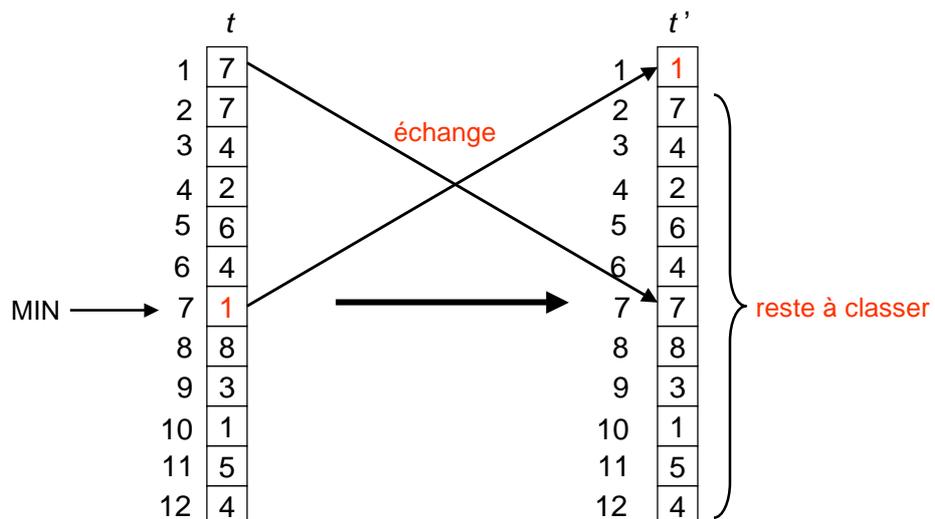
1	Guy	1m75	60 k
2	Anne	1m70	55 k
3	Lou	1m75	57 k
4	Luc	1m72	61 k

Classement par rapport à la clé (TAILLE, POIDS)



Problème équivalent à :
trier (1, 2, ..., n) suivant Clé o e

Tri par sélection



Recherche du minimum par balayage séquentiel

Ou organisation des éléments en « tas » (voir « file de priorité »)

Tri par sélection

UMLV ©

```

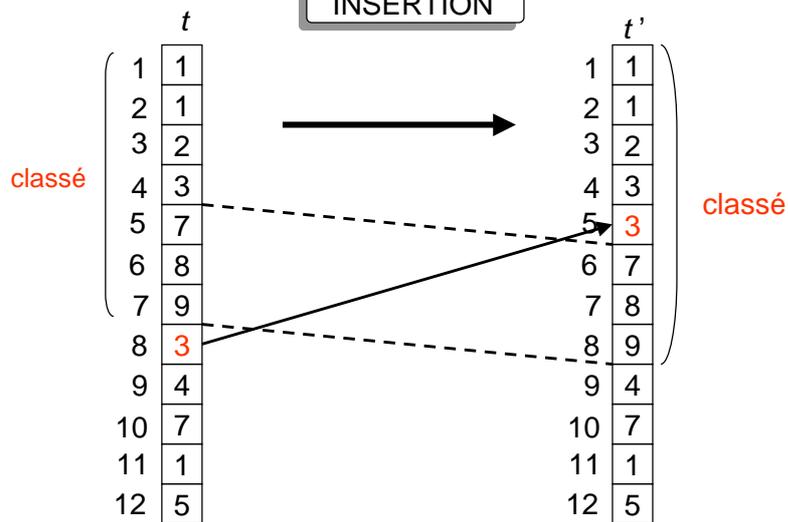
fonction TRI_PAR_SELECTION (t table [1...n] ) : table ;
début
  pour i ← 1 à n-1 faire
  { min ← i;
    pour j ← i+1 à n faire
      si t[j] < t[min] alors min ← j;
      temp ← t [i] ;
      t[i] ← t[min] ;
      t[min] ← temp ;
    } retour (t) ;
fin .
  
```

Complexité : espace $O(1)$ $\left\{ \begin{array}{l} O(n^2) \text{ comparaisons} \\ n-1 \text{ échanges} \end{array} \right.$
 temps $O(n^2)$

57

INSERTION

UMLV ©



Point d'insertion
 - recherche séquentielle
 - recherche dichotomique

58

TRI PAR INSERTION

UMLV ©

```

fonction TRI_PAR_INSERTION ( t table [1...n] ) : table ;
début t[0] ← -∞ ;
    pour i ← 2 à n faire
        { k ← i - 1 ; temp ← t[i] ;
          tant que temp < t[k] faire
              { t[k + 1] ← t[k] ; k ← k - 1 ; }
              t[k + 1] ← temp ;
          }
    retour ( t ) ;
fin .
    
```

Complexité : espace $O(1)$ { $O(n^2)$ comparaisons
 temps $O(n^2)$ { $O(n^2)$ affectations d'éléments

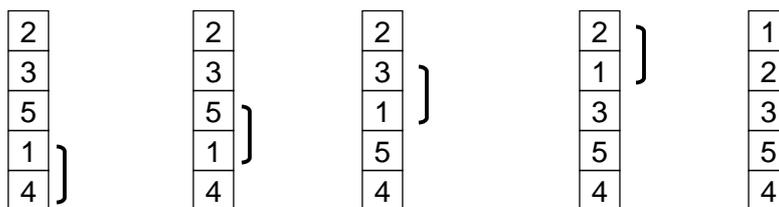
Insertion dichotomique : $O(n \log n)$ comparaisons

59

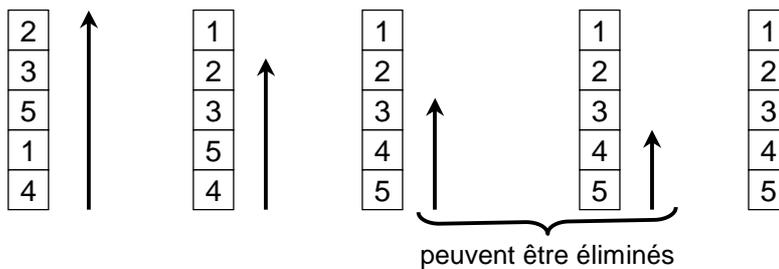
Tri des Bulles

UMLV ©

Un balayage



Suite des balayages



60

TRI DES BULLES

UMLV ©

fonction TRI_des_BULLES (*t* table [1...*n*]) : table ;

début

$i \leftarrow 1$;

tant que $i \leq n - 1$ **faire**

{ *dernier_échange* $\leftarrow n$;

pour $k \leftarrow n$ à $i + 1$ **pas - 1 faire**

si $t[k - 1] > t[k]$ **alors**

 { *temp* $\leftarrow t[k - 1]$;

$t[k - 1] \leftarrow t[k]$;

$t[k] \leftarrow temp$;

dernier_échange $\leftarrow k$;

 }

$i \leftarrow dernier_échange$;

}

retour (*t*) ;

fin.

Complexité : espace $O(1)$

 temps $O(n^2)$ comparaisons et échanges

61

PARTAGE / FUSION

UMLV ©

Si liste assez longue

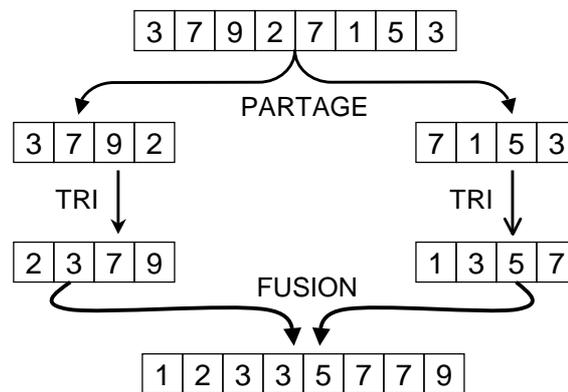


Schéma pour : TRI RAPIDE
 TRI par FUSION

62

Schéma correct si

- correct sur listes courtes
- partage correct
- fusion correcte

et surtout si

- $(L_1, L_2) = \text{PARTAGE}(L)$
 $\Rightarrow |L_1| < |L|$ et $|L_2| < |L|$

$T_{\text{MAX}} = O(n \log n)$ si $|L_1| \sim |L_2| \sim |L|/2$ et
 PARTAGE et FUSION linéaires

Soit $n = |L| = 2^k$ (après ajout éventuel d'éléments fictifs)

$$T(n) = \gamma n + 2 T(n/2) \text{ si } n > 1$$

$$T(1) = \beta$$

$$\begin{aligned} \frac{T(2^k)}{2^k} &= \gamma + \frac{T(2^{k-1})}{2^{k-1}} = \gamma + \gamma + \frac{T(2^{k-2})}{2^{k-2}} \\ &= \gamma k + \beta \end{aligned}$$

$$T(n) = \gamma n \log n + \beta n = O(n \log n)$$

TRI RAPIDE

UMLV ©

Partage avec *pivot* = 3

2	4	1	7	3	2	3	6
---	---	---	---	---	---	---	---

2	2	1	7	3	4	3	6
---	---	---	---	---	---	---	---

< 3				≥ 3			
2	2	1	3	3	4	7	6
TRI				TRI			
1	2	2	3	3	4	6	7

Suite du tri

65

UMLV ©

fonction TRI_RAPIDE (*t* table [1...*n*]) : table ;

début

appliquer TR (1, *n*) à *t* ;

retour (*t*) ;

fin .

procédure TR (*i*, *j*) ;

/* classe la partie *t* [*i*..*j*] de *t* */

début

si *i* < *j* **alors**

{ *p* ← choix (*i*, *j*) ;

k ← PARTAGE (*i*, *j*, *p*) ;

TR (*i*, *k* - 1) ; TR (*k* + 1, *j*) ;

}

fin .

choix(*i*, *j*) : choix aléatoire d'un entier entre *i* et *j*.

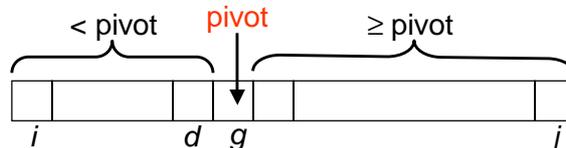
66

```

fonction PARTAGE ( i, j, p );
/* partage t suivant le pivot t[ p ] */
début
  g ← i ; d ← j ; échanger ( t[ p ], t[ j ] ) ; pivot = t[ j ] ;
  répéter
    tant que t[ g ] < pivot faire g ← g + 1 ;
    tant que d ≥ g et t[ d ] ≥ pivot faire d ← d - 1 ;
    si g < d alors
      { échanger ( t[ g ], t[ d ] ) ;
        g ← g + 1 ; d ← d - 1 ;
      }
  jusqu' à ce que g > d ;
  échanger ( t[ g ], t[ j ] ) ;
  retour ( g ) ;

```

fin.



Complexité du Tri rapide

Temps (PARTAGE (i, j, p)) = $\Theta (j - i + 1)$ ($i < j$)

$T_{\text{MAX}}(\text{TR} (1, n)) = \mathcal{O} (n^2)$

Exemple : table déjà classée et choix successifs
de $p : 1, 2, \dots, n - 1$

$T_{\text{MOY}}(\text{TR} (1, n)) = \mathcal{O} (n \log n)$

$(t[1], \dots, t[n]) = (1, 2, \dots, n)$

toutes les permutations de $t[1], \dots, t[n]$ équiprobables.

Preuve [$T (n) = T_{\text{MOY}}(\text{TR} (1, n))$]

probabilité $1/n$ de d' avoir k parmi $\{ 1, 2, \dots, n \}$

$T (0) = T (1) = \beta$

$$T(n) \leq \gamma n + \frac{1}{n} \sum_{k=1}^n [T(k-1) + T(n-k)] \quad n \geq 2$$

↑
Partage

↑ ↑
Appels récursifs

$$\Rightarrow T(n) \leq (2\beta + 2\gamma) n \log_e n \quad n \geq 2$$

par récurrence sur n .

$$\begin{aligned}
T(n) &\leq \gamma n + \frac{1}{n} \sum_{k=1}^n [T(k-1) + T(n-k)] \quad n \geq 2 \\
&\leq \gamma n + \frac{2}{n} \sum_{k=1}^n T(k-1) \leq \gamma n + \frac{4\beta}{n} + \frac{2}{n} \sum_{i=2}^{n-1} T(i) \\
&\leq \gamma n + \frac{4\beta}{n} + \frac{2(2\beta + 2\gamma)}{n} \sum_{i=2}^{n-1} i \log i \quad \text{par Hyp. Rec.}
\end{aligned}$$

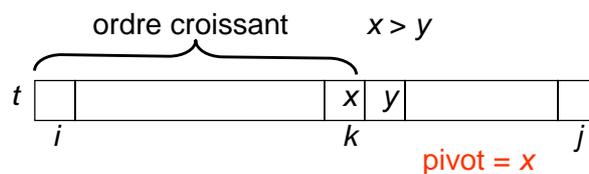
Comme $\sum_{i=2}^{n-1} i \log i \leq \int_2^n x \log x dx \leq \frac{n^2}{2} (\log n - \frac{1}{2})$

On obtient

$$\begin{aligned}
T(n) &\leq \gamma n + \frac{4\beta}{n} + (2\beta + 2\gamma)n \log n - (2\beta + 2\gamma) \frac{n}{2} \\
&\leq (2\beta + 2\gamma)n \log n \quad \text{C.Q.F.D.}
\end{aligned}$$

Autre choix du pivot

Plus de garantie sur le temps d'exécution moyen



fonction choix (i, j) : indice ;
début

pour $k \leftarrow i$ à $j - 1$ **faire**

si $t[k] > t[k + 1]$ **alors**

retour k ;

retour -1 ; /* cas $t[j] \leq t[i+1] \leq \dots \leq t[j]$ */

fin.

```

fonction TRI_RAPIDE_ITERATIF ( t table [1...n] ) : table ;
début
  P ← empiler(Pile_vide, (1,n)) ;
  tantque non vide(P) faire
    (i, j) ← sommet(P) ; dépiler(P) ;
    si i < j alors
      p ← choix (i, j) ;
      k ← PARTAGE (i, j, p) ;
      P ← empiler(P, (k+1, j)) ;
      P ← empiler(P, (i, k-1)) ;
    retour t ;
fin.

```

NOTE : ordre des empilements sans importance.

Hauteur de la pile : $O(n)$

Suppression de la récursivité terminale avec optimisation

```

procédure TR_optimisé (i, j) ;
début
  tant que i < j faire
    { p ← choix (i, j) ;
      k ← PARTAGE (i, j, p) ;
      si k - i < j - k alors
        { TR_optimisé (i, k - 1) ; i ← k + 1 ; }
      sinon
        { TR_optimisé (k + 1, j) ; j ← k - 1 ; }
    }
fin.

```

Hauteur de la pile de récursion : $O(\log n)$