

Les tris

Sandrine Vial
`sandrine.vial@uvsq.fr`

Septembre 2020

Les tris par comparaison

Données

- ▶ Collection de N valeurs du même type rangées dans un tableau T
- ▶ Un opérateur de comparaison (\leq, \geq)

But

Ré-ordonner les valeurs de T de telle sorte que :

$$T[i] \leq T[i + 1], \forall i \in \{1 \dots N - 1\}$$

Quelques algorithmes de tris

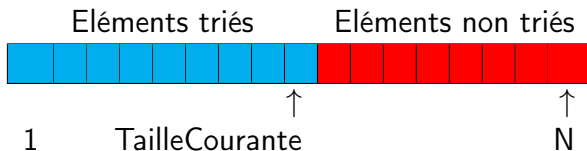
- ▶ Le tri par insertion
- ▶ Le tri à bulles (par permutation)
- ▶ Le tri fusion
- ▶ Le tri rapide (Quicksort)

Le tri par insertion

Principe Général

A tout moment le tableau T est séparé en 2 parties :

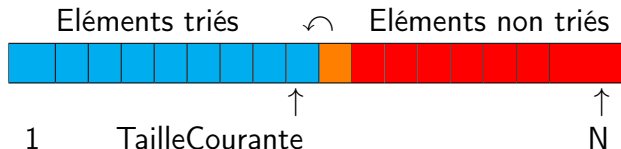
- ▶ $T[1] \dots T[TailleCourante]$: Partie déjà triée du tableau
- ▶ $T[TailleCourante + 1] \dots T[N]$: Partie non triée du tableau.



Le tri par insertion (2)

Une Etape

- ▶ Prendre un élément non encore trié ;
- ▶ L'insérer à sa place dans l'ensemble des éléments triés.



Le tri par insertion (3)

Algorithme 1 Tri par insertion

TriInsertion(T : tableau d'entiers, N : entier)

▷ *Variables Locales*

$TC, i, p, temp$: entiers

Debut

pour TC de 1 à $N - 1$ faire

$temp \leftarrow T[TC + 1]$

$p \leftarrow 1$

tant que $T[p] < temp$ faire

$p \leftarrow p + 1$

fin tant que

Chercher la position p

pour i de TC en décroissant à p faire

$T[i+1] \leftarrow T[i]$

fin pour

$T[p] \leftarrow temp$

fin pour

Fin

Décaler les éléments

Tri par insertion(4)

Complexité pour N éléments

- ▶ Le corps de la boucle est exécuté $N - 1$ fois
- ▶ Une itération :
 - ▶ Recherche de la position : p
 - ▶ Décalage des éléments : $TC - p$
 - ▶ Total : TC
- ▶ Au total :

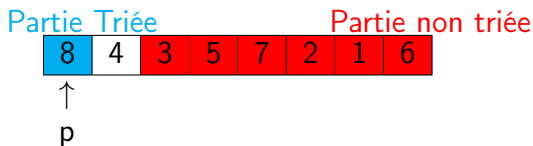
$$\sum_{i=1}^{N-1} i = \frac{N(N-1)}{2}$$

La complexité du tri par insertion est en $O(N^2)$.

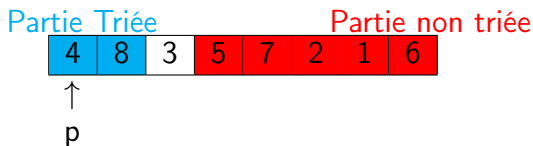
Un exemple de tri par insertion

8	4	3	5	7	2	1	6
---	---	---	---	---	---	---	---

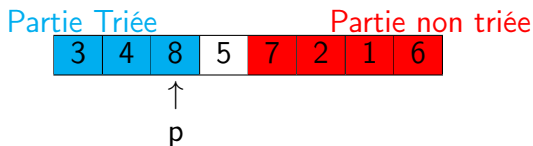
Un exemple de tri par insertion



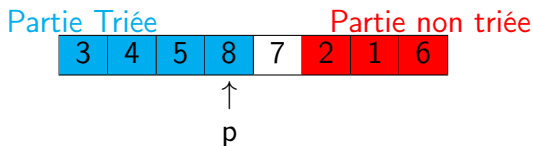
Un exemple de tri par insertion



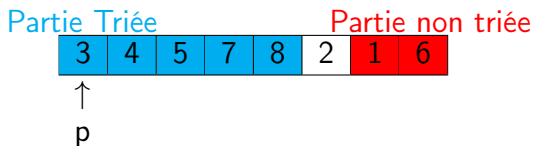
Un exemple de tri par insertion



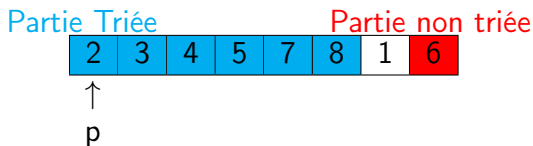
Un exemple de tri par insertion



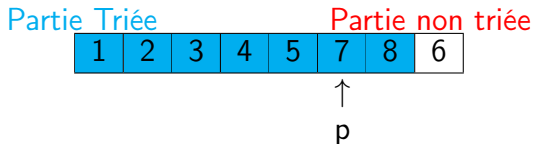
Un exemple de tri par insertion



Un exemple de tri par insertion



Un exemple de tri par insertion



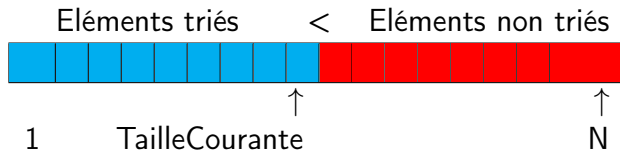
Un exemple de tri par insertion

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

Tri par permutation

Principe général

- ▶ Si deux éléments voisins ne sont pas ordonnés correctement, on les échange.
 - ▶ Deux parties dans le tableau :
 - ▶ Une partie avec des éléments triés
 - ▶ Une partie avec des éléments non triés
- de telle sorte que les éléments de la partie triée sont inférieurs aux éléments de la partie non triée.



Tri par permutation (2)

Algorithme 2 Tri par permutation

TriPermutation(T : tableau d'entiers, N : entier)

▷ *Variables Locales*

i, TC : entiers

Debut

pour TC de 2 à N faire

pour i de N en décroissant à TC faire

si $T[i-1] > T[i]$ faire

$T[i-1] \leftrightarrow T[i]$

fin si

fin pour

fin pour

Fin

Tri par permutation (3)

Complexité pour N éléments

- ▶ Boucle externe : $N - 2$ fois
- ▶ Boucle interne : $N - TC$ fois
- ▶ Total : $\frac{(N-1)(N-2)}{2}$

La complexité du tri par permutation est en $O(N^2)$.

8	4	3	5	7	2	1	6
---	---	---	---	---	---	---	---

TC



8	4	3	5	7	2	1	6
---	---	---	---	---	---	---	---

TC



8	4	3	5	7	2	1	6
---	---	---	---	---	---	---	---

TC



8	4	3	5	7	1	2	6
---	---	---	---	---	---	---	---

TC



8	4	3	5	1	7	2	6
---	---	---	---	---	---	---	---

TC



8	4	3	1	5	7	2	6
---	---	---	---	---	---	---	---

TC



8	4	1	3	5	7	2	6
---	---	---	---	---	---	---	---

TC



8	1	4	3	5	7	2	6
---	---	---	---	---	---	---	---

TC



1	8	4	3	5	7	2	6
---	---	---	---	---	---	---	---

TC



1	8	4	3	5	7	2	6
---	---	---	---	---	---	---	---

TC



1	8	4	3	5	2	7	6
---	---	---	---	---	---	---	---

TC



1	8	4	3	2	5	7	6
---	---	---	---	---	---	---	---

TC



1	8	4	2	3	5	7	6
---	---	---	---	---	---	---	---

TC



1	8	2	4	3	5	7	6
---	---	---	---	---	---	---	---

TC



1	2	8	4	3	5	7	6
---	---	---	---	---	---	---	---

TC



1	2	8	4	3	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	8	4	3	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	8	4	3	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	8	3	4	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	3	8	4	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	3	8	4	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	3	8	4	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	3	8	4	5	6	7
---	---	---	---	---	---	---	---

TC



1	2	3	4	8	5	6	7
---	---	---	---	---	---	---	---

TC

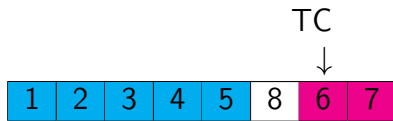


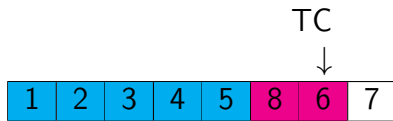
1	2	3	4	8	5	6	7
---	---	---	---	---	---	---	---

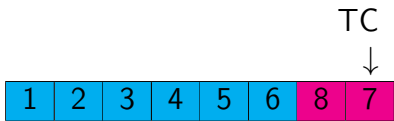
TC

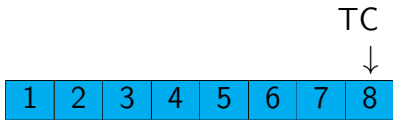


1	2	3	4	8	5	6	7
---	---	---	---	---	---	---	---









Tri Fusion

- ▶ Machine à trier des cartes perforées en 1938 ;
- ▶ 1er algo de tri fusion écrit par Von Neumann pour l'EDVAC en 1945 ;
- ▶ Basé sur le paradigme
«Diviser pour régner»

Diviser Pour Régner

- ▶ Séparer le problème en plusieurs sous-problèmes similaires au problème initial.
- ▶ 3 étapes :
 - ▶ **Diviser** le problème en un certain nombre de sous-problèmes
 - ▶ **Régner** sur les sous-problèmes en les résolvant
 - ▶ **Combiner** les solutions des sous-problèmes en une solution unique au problème initial.

Tri Fusion (2)

3 étapes :

- ▶ **Diviser** le tableau de N éléments à trier en 2 sous-tableaux de $\frac{N}{2}$ éléments.
- ▶ **Régner** :
 - ▶ Tout tableau de longueur 1 est trié.
 - ▶ Trier les 2 sous-tableaux récursivement à l'aide du Tri Fusion
- ▶ **Combiner** : **Action Principale : la Fusion**
 - ▶ Fusionner les 2 sous-tableaux triés pour produire un tableau trié.

Tri Fusion (3)

Algorithme 3 Tri Fusion

TriFusion(T : tableau d'entiers, p : entier, r : entier)

▷ p et r sont les indices entre lesquels on veut trier le tableau. On suppose $p \leq r$.

Debut

si $p < r$ **faire**

$q \leftarrow \lfloor \frac{p+r}{2} \rfloor$

TriFusion(T, p, q)

TriFusion($T, q+1, r$)

Fusion(T, p, q, r)

fin si

Fin

Tri Fusion (4)

Algorithme 4 Tri Fusion

Fusion(T : tableau d'entiers, p : entier, q : entier, r : entier)

▷ *Entrées* : T : tableau d'entiers. p , q et r : indices entre lesquels on veut trier le tableau avec $p \leq q \leq r$.

▷ *Sortie* : T : tableau trié entre les indices p et r .

▷ *Pré-condition* : T tableau trié entre les indices p et q et T trié entre les indices $q+1$ et r

▷ *Variables locales* : i, j, k : entiers et B : tableau d'entiers

Debut

$i \leftarrow p$; $k \leftarrow p$; $j \leftarrow q + 1$;

tant que ($i \leq q$ et $j \leq r$) **faire**

si $T[i] < T[j]$ **faire**

$B[k] \leftarrow T[i]$

$i \leftarrow i + 1$

sinon

$B[k] \leftarrow T[j]$

$j \leftarrow j + 1$

fin si

$k \leftarrow k + 1$

fin tant que

tant que $i \leq q$ **faire**

$B[k] \leftarrow T[i]$

$i \leftarrow i + 1$

$k \leftarrow k + 1$

fin tant que

tant que $j \leq r$ **faire**

$B[k] \leftarrow T[j]$

$j \leftarrow j + 1$

$k \leftarrow k + 1$

fin tant que

$T \leftarrow B$

Fin

Tri Fusion (5)

Complexité pour N éléments

- Intuitivement il faut résoudre :

$$Tri(N) = 2 \times Tri\left(\frac{N}{2}\right) + \Theta(N)$$

- $\Theta(N)$: complexité de la fusion

La complexité du tri fusion est en $\Theta(N \log_2 N)$.

8	4	3	5	7	2	1	6
---	---	---	---	---	---	---	---

8 4 3 5



8 4 3 5 7 2 1 6

8 4 3 5 7 2 1 6



8 4 3 5



8 4

8 4 3 5 7 2 1 6



8 4 3 5



8 4



8

8 4 3 5 7 2 1 6



8 4 3 5



8 4



8



4

8 4 3 5 7 2 1 6



8 4 3 5



4 8



8



4

8 4 3 5 7 2 1 6



8 4 3 5



4 8



3 5



8



4

8 4 3 5 7 2 1 6



8 4 3 5



3 5

4 8



8

4

3

8 4 3 5 7 2 1 6



8 4 3 5



3 5

4 8



8

4

3

5

8 4 3 5 7 2 1 6



8 4 3 5



4 8

3 5



8

4

3

5

8 4 3 5 7 2 1 6



3 4 5 8



4 8

3 5



8

4

3

5

8 4 3 5 7 2 1 6



3 4 5 8



7 2 1 6



4 8

3 5



8

4

3

5

8 4 3 5 7 2 1 6

3 4 5 8

4 8

8

4

3

5

3 5

7 2

7 2 1 6

8 4 3 5 7 2 1 6

3 4 5 8

4 8

8

4

3

3 5

5

7

7 2

7 2 1 6

8 4 3 5 7 2 1 6

3 4 5 8

4 8

8

4

3

3 5

5

7

7 2

2

7 2 1 6

8 4 3 5 7 2 1 6

3 4 5 8

7 2 1 6

4 8

3 5

2 7

8

4

3

5

7

2

8 4 3 5 7 2 1 6

3 4 5 8

7 2 1 6

4 8

3 5

2 7

1 6

8

4

3

5

7

2

8 4 3 5 7 2 1 6

3 4 5 8

7 2 1 6

4 8

3 5

2 7

1 6

8

4

3

5

7

2

1

8 4 3 5 7 2 1 6

3 4 5 8

7 2 1 6

4 8

3 5

2 7

1 6

8

4

3

5

7

2

1

6

8 4 3 5 7 2 1 6

3 4 5 8

7 2 1 6

4 8

3 5

2 7

1 6

8

4

3

5

7

2

1

6

8 4 3 5 7 2 1 6

3 4 5 8

1 2 6 7

4 8

3 5

2 7

1 6

8

4

3

5

7

2

1

6

1 2 3 4 5 6 7 8

3 4 5 8

4 8

8

4

3

3 5

5

2 7

7

2

1 2 6 7

1 6

1

6

Tri Rapide

- ▶ Proposé par Hoare en 1962.
- ▶ Basé sur le paradigme «diviser pour régner» :
 - ▶ **Diviser** : le Tableau $T[p..r]$ est divisé en 2 sous-tableaux non vides. Trouver q de telle sorte que chaque élément de $T[p..q]$ soit inférieur à chaque élément de $T[q + 1..r]$.
 - ▶ **Régner** : 2 sous-tableaux sont triés grâce à la récursité.
 - ▶ **Combiner** : rien à faire.

Tri Rapide (2)

Algorithme 5 Tri Rapide

TriRapide(T : tableau d'entiers, p : entier, r : entier)

▷ p et r sont les indices entre lesquels on veut trier le tableau. On suppose $p \leq r$.

Debut

 si $p < r$ faire

$q \leftarrow \text{partitionner}(T, p, r)$

 TriRapide(T, p, q)

 TriRapide($T, q+1, r$)

 fin si

Fin

Tri Rapide (3)

Algorithme 6 Partitionner

Partitionner(T : tableau d'entiers, p : entier, r : entier)

▷ p et r sont les indices entre lesquels on veut trier le tableau. On suppose $p \leq r$.

▷ Variables locales : $i, j, pivot$: entiers

Debut

$i \leftarrow p$; $j \leftarrow r$; $pivot \leftarrow T[p]$;

tant que ($i < j$) faire

 tant que ($T[i] < pivot$) faire $i \leftarrow i + 1$ fin tant que

 tant que ($T[j] > pivot$) faire $j \leftarrow j - 1$ fin tant que

 si ($i < j$) faire

$T[i] \leftrightarrow T[j]$

$i \leftarrow i + 1$

$j \leftarrow j - 1$

 fin si

fin tant que

retourner j

Fin

Tri Rapide (4)

Complexité pour N éléments

- ▶ Partitionner N éléments coûte $\Theta(N)$.
- ▶ Temps d'exécution dépend de l'équilibre ou non du partitionnement :
 - ▶ S'il est équilibré : **aussi rapide que le tri fusion**
 - ▶ S'il est déséquilibré : **aussi lent que le tri par insertion**

Tri Rapide (5)

Partitionnement dans le pire cas

- ▶ 2 sous-tableaux de 1 élément et $N - 1$ éléments.
- ▶ Trier un élément coûte $\Theta(1)$
- ▶ Supposons que ce partitionnement déséquilibré intervienne à chaque étape.
- ▶ Résolution de :

$$Tri(N) = Tri(N - 1) + \Theta(N)$$

Tri Rapide (6)

Partitionnement dans le pire cas

$$Tri(N) = Tri(N - 1) + \Theta(N)$$

$$= \sum_{k=1}^N \Theta(k)$$

$$= \Theta\left(\sum_{k=1}^N k\right)$$

$$= \Theta(N^2)$$

- ▶ Ce partitionnement apparaît quand le tableau est trié !
- ▶ Dans ce cas-là le tri par insertion est linéaire !

Tri Rapide (7)

Partitionnement dans le meilleur cas

- ▶ Le partitionnement est équilibré
- ▶ Il faut résoudre :

$$Tri(N) = 2 \times Tri\left(\frac{N}{2}\right) + \Theta(N)$$

- ▶ Solution : $Tri(N) = \Theta(N \log_2 N)$

Optimalité des tris par comparaisons

- ▶ Un tri par comparaison a une complexité en $\Omega(N \log_2 N)$
- ▶ Les tris qui ont une complexité en $\theta(N \log_2 N)$ sont **optimaux**

Synthèse pour N éléments

Algorithme	Pire cas	En moyenne	Meilleur cas
Insertion	$O(N^2)$	$O(N^2)$	$O(N)$
Permutation	$O(N^2)$	$O(N^2)$	$O(N^2)$
Fusion	$O(N \log_2 N)$	$O(N \log_2 N)$	$O(N \log_2 N)$
Rapide	$O(N^2)$	$O(N \log_2 N)$	$O(N \log_2 N)$