

Éléments d'Algorithmique

CMTD3: Complexité des Tris par sélection et par insertion

Christine Tasson

Université de Paris, IRIF



INSTITUT
DE RECHERCHE
EN INFORMATIQUE
FONDAMENTALE



Algorithmes de tris

Données : Tableau T contenant des éléments que l'on peut ordonner (entiers, paires d'entiers, chaînes de caractères, ...)

Algorithme : Plusieurs possibilités (Sélection, Insertion, Fusion, ...)

Sortie : Un tableau **ordonné** et qui contient exactement les **mêmes éléments** que ceux de T

Efficacité des algorithmes de tri

Des **simulations** de différents tris peuvent être visualisées à l'adresse suivante :

[Simulations des Algorithmes de Tris](#)

Ces simulations donnent une indication sur l'efficacité des tris. Nous allons maintenant calculer leur **complexité théorique**.

Dans ce cours : Complexité des Tris

La complexité d'un algorithme est le nombre d'opérations élémentaires pour résoudre le problème.

Opérations élémentaires pour un tableau t

- Affectations : $t[i] = 5$
- Comparaisons : $t[i] < t[j]$

Comparaison de deux algorithmes

- Tri par sélection
- Tri par insertion

Tri par sélection

Tri par sélection - Principe

Le **tri par sélection** consiste à échanger le minimum des éléments non triés pour le placer à la suite des éléments triés.

10, **1**, 5, 19, 3, 3

1, 10, 5, 19, **3**, 3

1, 3, 5, 19, 10, **3**

1, 3, 3, 19, 10, **5**

1, 3, 3, 5, **10**, 19

1, 3, 3, 5, 10, 19

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION( $T$ )  
2:    $n \leftarrow$  longueur de  $T$   
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire  
4:     trouver l'indice  $\min$  du plus  
5:       petit élément parmi  $T[i, \dots, n-1]$   
6:       échanger  $T[i]$  et  $T[\min]$ 
```

Nombres d'opérations élémentaires = nombre de comparaisons

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION( $T$ )  
2:    $n \leftarrow$  longueur de  $T$   
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire  
4:     trouver l'indice  $\min$  du plus  
       petit élément parmi  $T[i, \dots, n-1]$   
5:     échanger  $T[i]$  et  $T[\min]$ 
```

Nombres d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du \min dans un tableau de longueur $n-i$.

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION( $T$ )  
2:    $n \leftarrow$  longueur de  $T$   
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire  
4:     trouver l'indice  $\min$  du plus  
5:       petit élément parmi  $T[i, \dots, n-1]$   
6:       échanger  $T[i]$  et  $T[\min]$ 
```

Nombres d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du \min dans un tableau de longueur $n-i$. Nécessite $n-i-1$ comparaisons

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION(T)
2:    $n \leftarrow$  longueur de T
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire
4:     trouver l'indice min du plus
       petit élément parmi  $T[i, \dots, n-1]$ 
5:     échanger  $T[i]$  et  $T[\text{min}]$ 
```

$$\sum_{i=0}^{n-2} (n - i - 1)$$

Nombres d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du min dans un tableau de longueur $n-i$. Nécessite $n-i-1$ comparaisons

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION(T)
2:    $n \leftarrow$  longueur de T
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire
4:     trouver l'indice min du plus
       petit élément parmi  $T[i, \dots, n-1]$ 
5:     échanger  $T[i]$  et  $T[\text{min}]$ 
```

Nombres d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du min dans un tableau de longueur $n-i$. Nécessite $n-i-1$ comparaisons

$$\sum_{i=0}^{n-2} (n - i - 1) = (n - 1) + \dots + 1 = \sum_{\ell=1}^{n-1} \ell$$

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION(T)
2:    $n \leftarrow$  longueur de T
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire
4:     trouver l'indice min du plus
       petit élément parmi  $T[i, \dots, n-1]$ 
5:     échanger  $T[i]$  et  $T[\text{min}]$ 
```

Nombres d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du min dans un tableau de longueur $n-i$. Nécessite $n-i-1$ comparaisons

$$\sum_{i=0}^{n-2} (n-i-1) = (n-1) + \dots + 1 = \sum_{\ell=1}^{n-1} \ell = \frac{n(n-1)}{2}$$

Tri par sélection : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARSÉLECTION(T)
2:    $n \leftarrow$  longueur de T
3:   pour  $i \leftarrow 0$  à  $n - 2$  faire
4:     trouver l'indice min du plus
       petit élément parmi  $T[i, \dots, n-1]$ 
5:     échanger  $T[i]$  et  $T[\text{min}]$ 
```

Nombre d'opérations élémentaires = nombre de comparaisons

- Trouver l'indice du min dans un tableau de longueur $n-i$. Nécessite $n-i-1$ comparaisons

$$\sum_{i=0}^{n-2} (n-i-1) = (n-1) + \dots + 1 = \sum_{\ell=1}^{n-1} \ell = \frac{n(n-1)}{2}$$

Avec un nombre d'opérations élémentaires proportionnel à n^2 , la complexité du tri par sélection est **quadratique** et ne dépend pas du tableau de taille n passé en argument.

Tri par insertion

Tri par insertion - Principe

Le tri par insertion consiste à

- garder le début du tableau trié
- y insérer successivement, à leur place, les éléments non-triés.

10, **1**, 5, 19, 3, 3

1, 10, 5, 19, 3, 3

1, 10, **5**, 19, 3, 3

1, **5**, 10, 19, 3, 3

1, 5, 10, **19**, 3, 3

1, 5, 10, 19, **3**, 3

1, 5, 10, **3**, 19, 3

1, 5, **3**, 10, 19, 3

1, **3**, 5, 10, 19, 3

1, 3, 5, 10, 19, **3**

1, 3, 5, 10, **3**, 19

1, 3, 5, **3**, 10, 19

1, 3, **3**, 5, 10, 19

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:    $j \leftarrow i$ 
3:   tant que  $j > 0$  et  $T[j] < T[j - 1]$  faire
4:     échanger  $T[j - 1]$  et  $T[j]$ 
5:      $j \leftarrow j - 1$ 
6: fonction TRIPARINSERTION(T)
7:    $n \leftarrow$  longueur de T
8:   pour  $i \leftarrow 1$  à  $n - 1$  faire
9:     TRIPARTIEL(T, i)
```

Nombres d'opérations élémentaires = nombre de comparaisons

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:    $j \leftarrow i$ 
3:   tant que  $j > 0$  et  $T[j] < T[j - 1]$  faire
4:     échanger  $T[j - 1]$  et  $T[j]$ 
5:      $j \leftarrow j - 1$ 
6: fonction TRIPARINSERTION(T)
7:    $n \leftarrow$  longueur de T
8:   pour  $i \leftarrow 1$  à  $n - 1$  faire
9:     TRIPARTIEL(T, i)
```

Nombres d'opérations élémentaires = nombre de comparaisons
Appliquer TRIPARTIEL(T, i)

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:    $j \leftarrow i$ 
3:   tant que  $j > 0$  et  $T[j] < T[j - 1]$  faire
4:     échanger  $T[j - 1]$  et  $T[j]$ 
5:      $j \leftarrow j - 1$ 
6: fonction TRIPARINSERTION(T)
7:    $n \leftarrow$  longueur de T
8:   pour  $i \leftarrow 1$  à  $n - 1$  faire
9:     TRIPARTIEL(T, i)
```

Nombres d'opérations élémentaires = nombre de comparaisons

Appliquer TRIPARTIEL(T, i)

- Si $T[i - 1] < T[i]$, alors 1 comparaison (meilleur des cas)

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:    $j \leftarrow i$ 
3:   tant que  $j > 0$  et  $T[j] < T[j - 1]$  faire
4:     échanger  $T[j - 1]$  et  $T[j]$ 
5:      $j \leftarrow j - 1$ 
6: fonction TRIPARINSERTION(T)
7:    $n \leftarrow$  longueur de T
8:   pour  $i \leftarrow 1$  à  $n - 1$  faire
9:     TRIPARTIEL(T, i)
```

Nombres d'opérations élémentaires = nombre de comparaisons

Appliquer TRIPARTIEL(T, i)

- Si $T[i - 1] < T[i]$, alors **1 comparaison** (meilleur des cas)
- Si $T[i] < T[0]$, alors **i comparaisons** (pire des cas)

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:   j ← i
3:   tant que j > 0 et T[j] < T[j - 1] faire
4:     échanger T[j - 1] et T[j]
5:     j ← j - 1
6: fonction TRIPARINSERTION(T)
7:   n ← longueur de T
8:   pour i ← 1 à n - 1 faire
9:     TRIPARTIEL(T, i)
```

Pour la boucle, on somme :

- Dans le **pire des cas**, la complexité est **quadratique** : $\sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}$

Nombres d'opérations élémentaires = nombre de comparaisons

Appliquer TRIPARTIEL(T, i)

- Si $T[i - 1] < T[i]$, alors **1 comparaison** (meilleur des cas)
- Si $T[i] < T[0]$, alors **i comparaisons** (pire des cas)

Tri par insertion : Pseudocode/Complexité

Entrée : tableau T

```
1: fonction TRIPARTIEL(T, i)
2:    $j \leftarrow i$ 
3:   tant que  $j > 0$  et  $T[j] < T[j - 1]$  faire
4:     échanger  $T[j - 1]$  et  $T[j]$ 
5:      $j \leftarrow j - 1$ 
6: fonction TRIPARINSERTION(T)
7:    $n \leftarrow$  longueur de T
8:   pour  $i \leftarrow 1$  à  $n - 1$  faire
9:     TRIPARTIEL(T, i)
```

Nombres d'opérations élémentaires = nombre de comparaisons

Appliquer TRIPARTIEL(T, i)

- Si $T[i - 1] < T[i]$, alors **1 comparaison** (meilleur des cas)
- Si $T[i] < T[0]$, alors **i comparaisons** (pire des cas)

Pour la boucle, on somme :

- Dans le **pire des cas**, la complexité est **quadratique** : $\sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}$
- Dans le **meilleur des cas**, la complexité est **linéaire**.

En résumé

Complexité des algorithmes de tris

- Le tri par sélection est quadratique
- Le tri par insertion est quadratique dans le pire des cas, mais linéaire dans le meilleur des cas
- Il existe des algorithmes de tri (tri fusion) en $n \log(n)$
- On ne peut pas résoudre le problème du tri avec une meilleur complexité que $n \log(n)$.