Algorithmes de tri (2)

Ch. Lasou, N.E. Oussous, E. Wegrzynowski

Licence ST-A, USTL - API1

19 février 2007

1 Tri par fusion
- Principe
- Algorithmes
- Coût

2 Comparaison

Principe du tri par fusion

Pour trier le tableau t

$t[1..8]=$ | T | I | M | O | L | E | O | N |

Principe du tri par fusion

Pour trier le tableau t

$$t[1..8] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}\boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$$

- on coupe le tableau en deux parties de même taille

$$t[1..4] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}} \text{ et } t[5..8] = \boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$$

Principe du tri par fusion

Pour trier le tableau t

$$t[1..8] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}\boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$$

- on coupe le tableau en deux parties de même taille

  $t[1..4] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}$ et $t[5..8] = \boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$

- on trie chacune des deux moitiés

  $t[1..4] = \boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}\boxed{\texttt{T}}$, $t[5..8] = \boxed{\texttt{E}}\boxed{\texttt{L}}\boxed{\texttt{N}}\boxed{\texttt{O}}$

Principe du tri par fusion

Pour trier le tableau t

$$t[1..8] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}\boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$$

- on coupe le tableau en deux parties de même taille

  $t[1..4] = \boxed{\texttt{T}}\boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}$ et $t[5..8] = \boxed{\texttt{L}}\boxed{\texttt{E}}\boxed{\texttt{O}}\boxed{\texttt{N}}$
- on trie chacune des deux moitiés

  $t[1..4] = \boxed{\texttt{I}}\boxed{\texttt{M}}\boxed{\texttt{O}}\boxed{\texttt{T}}$, $t[5..8] = \boxed{\texttt{E}}\boxed{\texttt{L}}\boxed{\texttt{N}}\boxed{\texttt{O}}$
- puis on fusionne ces deux tableaux triés

  $t[1..8] = \boxed{\texttt{E}}\boxed{\texttt{I}}\boxed{\texttt{L}}\boxed{\texttt{M}}\boxed{\texttt{N}}\boxed{\texttt{O}}\boxed{\texttt{O}}\boxed{\texttt{T}}$

Les besoins

Il suffit donc de

Les besoins

Il suffit donc de

- savoir fusionner deux tranches triées $t[a..b]$ et $t[b+1..c]$ de façon à obtenir une tranche $t[a..c]$ triée.

Les besoins

Il suffit donc de

- savoir fusionner deux tranches triées $t[a..b]$ et $t[b+1..c]$ de façon à obtenir une tranche $t[a..c]$ triée.
- Spécification de la procédure de fusion
  fusion(t,a,b,c)
  Données : un tableau t, trois indices a, b, c
  CU : les tranches $t[a..b]$ et $t[b+1..c]$ sont triées
  But : rendre triée la tranche $t[a..c]$

Algorithme récursif

Algorithme récursif

- l'algorithme de tri par fusion est récursif

Algorithme récursif

- l'algorithme de tri par fusion est récursif
- la récursion dépendant des indices délimitant les tranches

Algorithme récursif

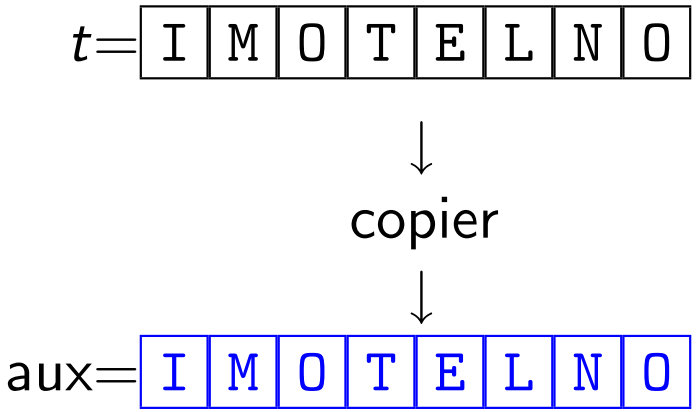
- l'algorithme de tri par fusion est récursif
- la récursion dépendant des indices délimitant les tranches
- spécification de la procédure
  tri_fusion_rec (t,a,b)
  Données : un tableau t, deux indices a et b
  But : trier la tranche $t[a..b]$

Algorithme de la fusion

fusion(t,a,b,c)
Données : un tableau t, trois indices a, b, c
CU : les tranches $t[a..b]$ et $t[b+1..c]$ sont triées
But : rendre triée la tranche $t[a..c]$

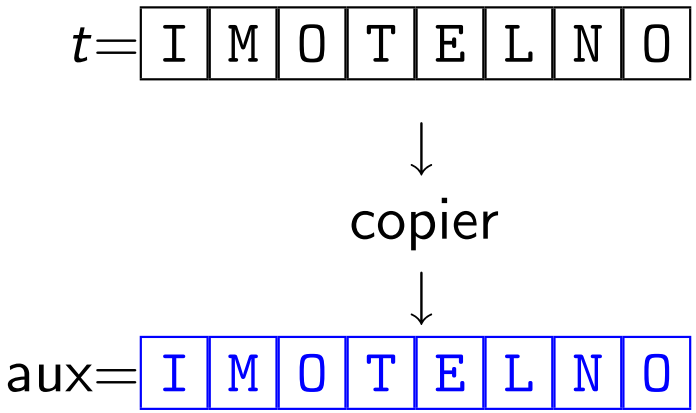
```
copier t[a..c] dans aux[a..c]

i := a   j := succ(b)
pour k := a à c faire
    si (i ≤ b) et (j ≤ c) alors
        si aux[i] ≤ aux[j] alors
            t[k] := aux[i] inc(i)
        sinon
            t[k] := aux[j] inc(j)
    sinon si i ≤ b alors
        t[k] := aux[i] inc(i)
    sinon
        t[k] := aux[j] inc(j)
    fin si
fin pour
```

Fusion en action



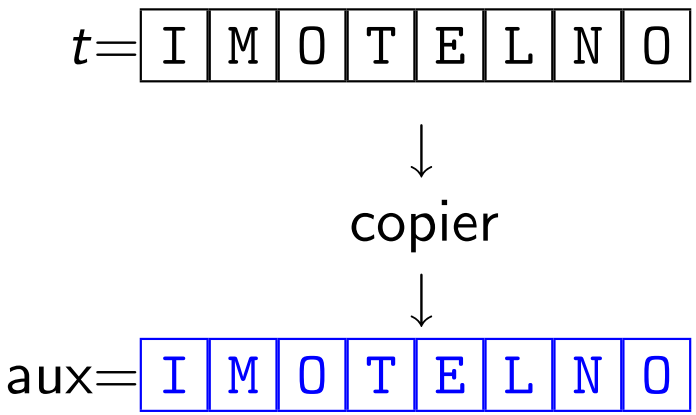
k	i j	t

Fusion en action



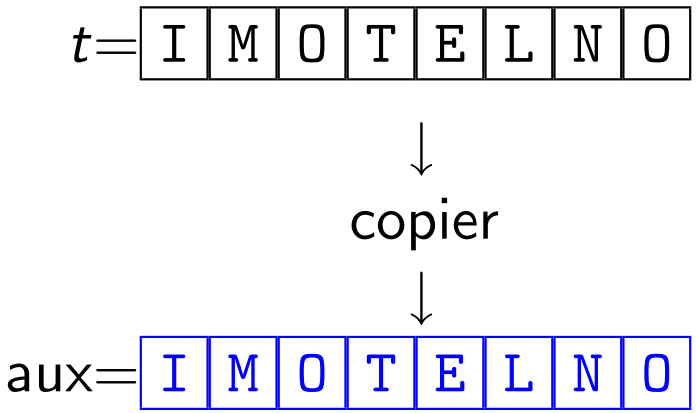
k	i	j	t
	1	5	I M O T E L N O

Fusion en action



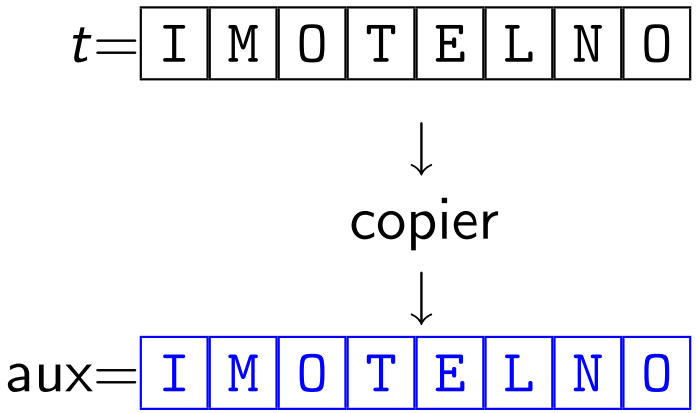
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O

Fusion en action



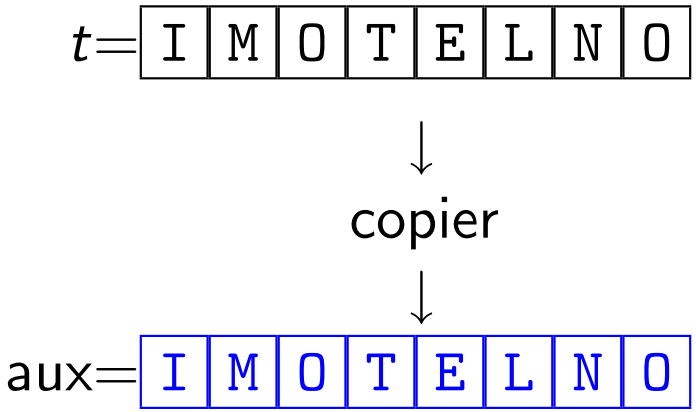
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O

Fusion en action



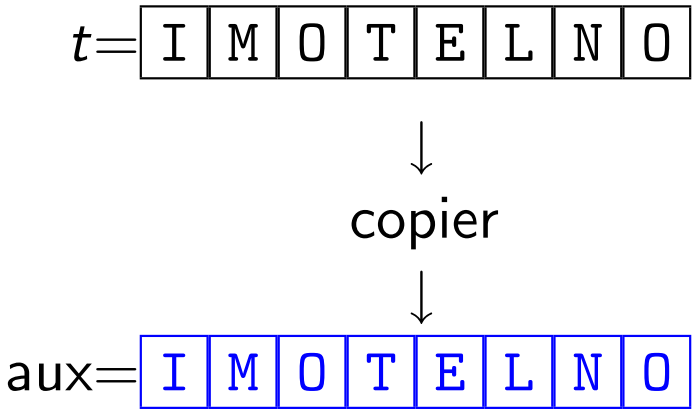
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O

Fusion en action



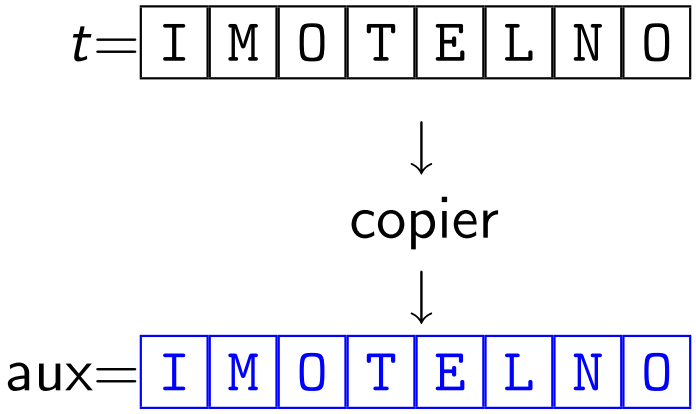
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O
4	3	7	E I L M E L N O

Fusion en action



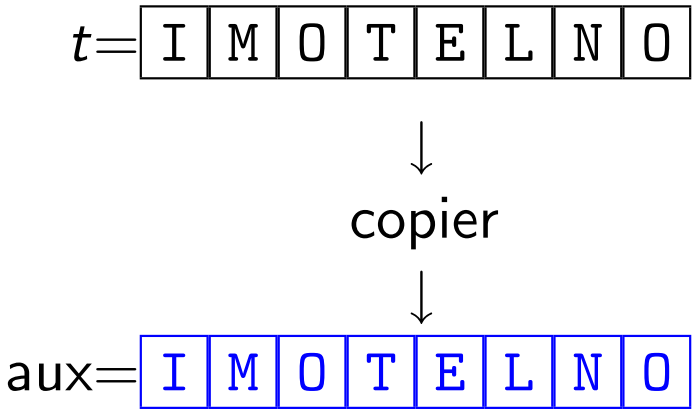
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O
4	3	7	E I L M E L N O
5	3	8	E I L M N L N O

Fusion en action



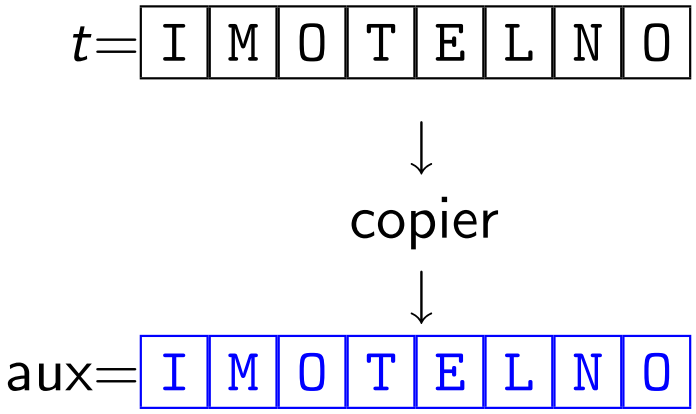
k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O
4	3	7	E I L M E L N O
5	3	8	E I L M N L N O
6	4	8	E I L M N O N O

Fusion en action



k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O
4	3	7	E I L M E L N O
5	3	8	E I L M N L N O
6	4	8	E I L M N O N O
7		9	E I L M N O O O

Fusion en action



k	i	j	t
	1	5	I M O T E L N O
1	1	6	E M O T E L N O
2	2	6	E I O T E L N O
3	2	7	E I L T E L N O
4	3	7	E I L M E L N O
5	3	8	E I L M N L N O
6	4	8	E I L M N O N O
7		9	E I L M N O O O
8	4		E I L M N O O T

Coût de la fusion

Pour fusionner $t[a..b]$ de taille n_1 avec $t[b+1..c]$ de taille n_2 ($n = n_1 + n_2$)

Coût de la fusion

Pour fusionner $t[a..b]$ de taille n_1 avec $t[b+1..c]$ de taille n_2 ($n = n_1 + n_2$)

- Dans tous les cas le nombre d'affectations est :

$$a_n = 2n$$

Coût de la fusion

Pour fusionner $t[a..b]$ de taille n_1 avec $t[b+1..c]$ de taille n_2 ($n = n_1 + n_2$)

- Dans tous les cas le nombre d'affectations est :

$$a_n = 2n$$

- Nombre de comparaisons

Coût de la fusion

Pour fusionner $t[a..b]$ de taille n_1 avec $t[b+1..c]$ de taille n_2 ($n = n_1 + n_2$)

- Dans tous les cas le nombre d'affectations est :

$$a_n = 2n$$

- Nombre de comparaisons
  - Meilleur des cas : tous les éléments de la tranche la plus petite sont inférieurs ou égaux à ceux de l'autre tranche

$$c_n = \min(n_1, n_2)$$

Coût de la fusion

Pour fusionner $t[a..b]$ de taille n_1 avec $t[b+1..c]$ de taille n_2 ($n = n_1 + n_2$)

- Dans tous les cas le nombre d'affectations est :

$$a_n = 2n$$

- Nombre de comparaisons
  - Meilleur des cas : tous les éléments de la tranche la plus petite sont inférieurs ou égaux à ceux de l'autre tranche

$$c_n = \min(n_1, n_2)$$

  - Pire des cas : les deux éléments les plus grands de la tranche $t[a..c]$ se trouvent dans deux tranches différentes

$$c_n = n - 1$$

Algorithme du tri

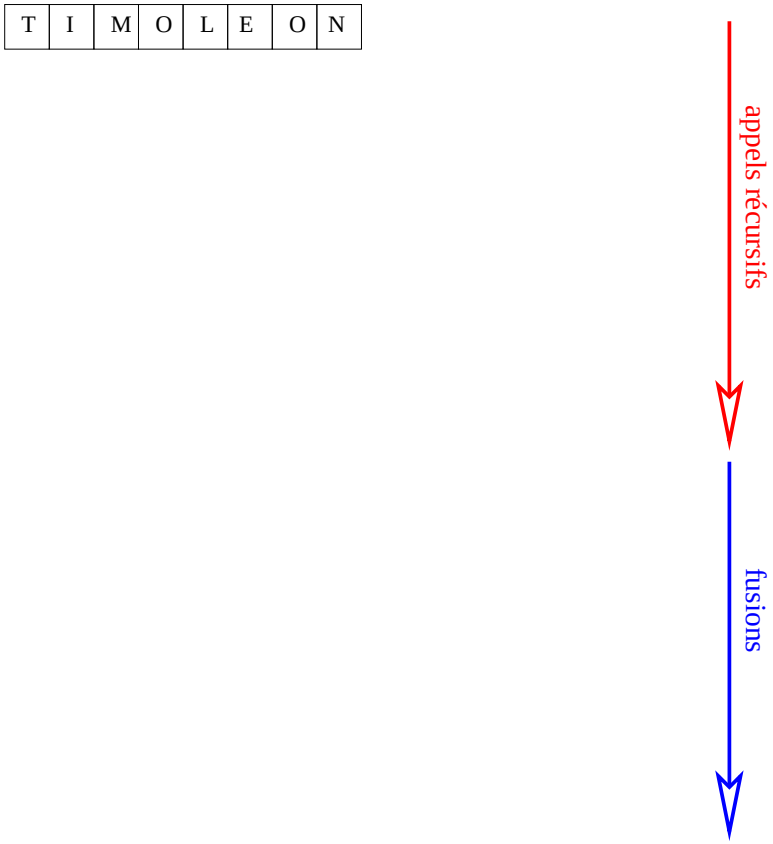
tri_fusion_rec (t,a,b)

Données : un tableau t, deux indices a et b

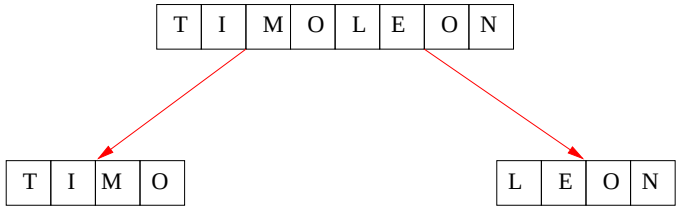
But : trier la tranche $t[a..b]$

```
si a<b alors
  m := milieu(a,b)
  tri_fusion_rec(t,a,m)
  tri_fusion_rec(t,succ(m),b)
  fusion(t,a,m,b)
fin si
```

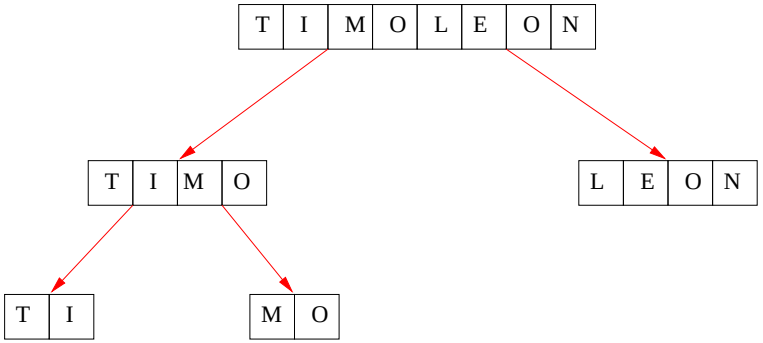
Exemple



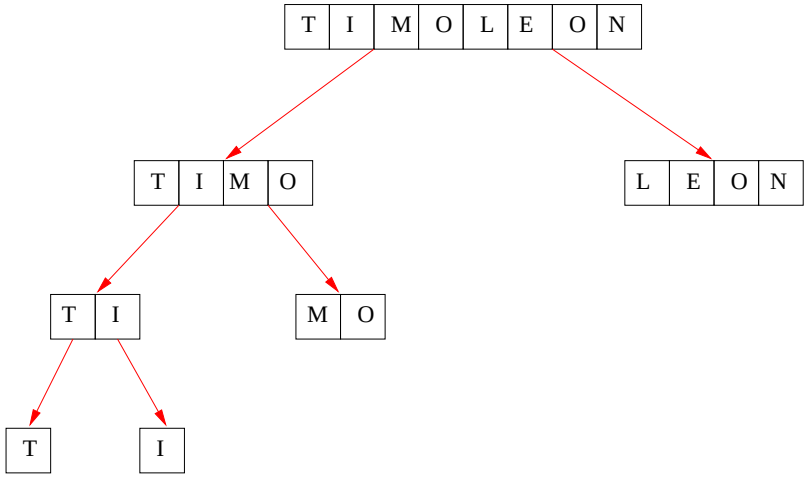
Exemple



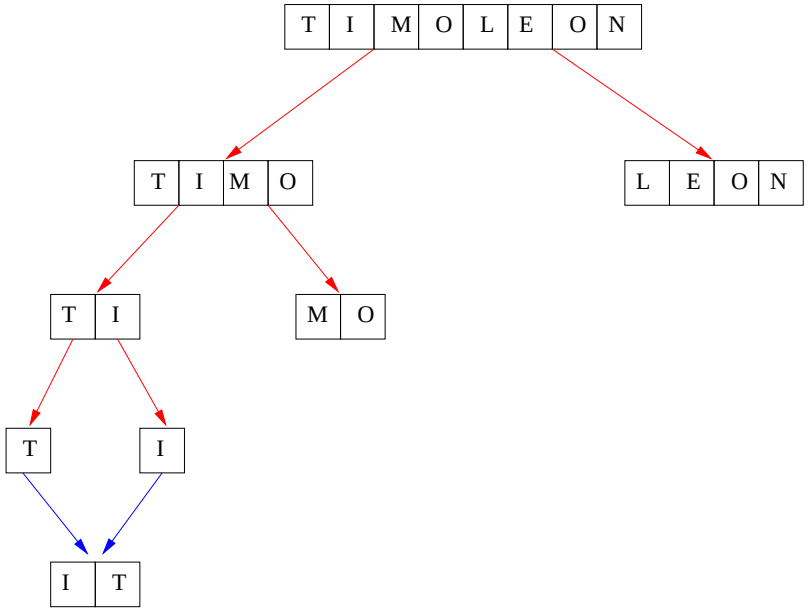
Exemple



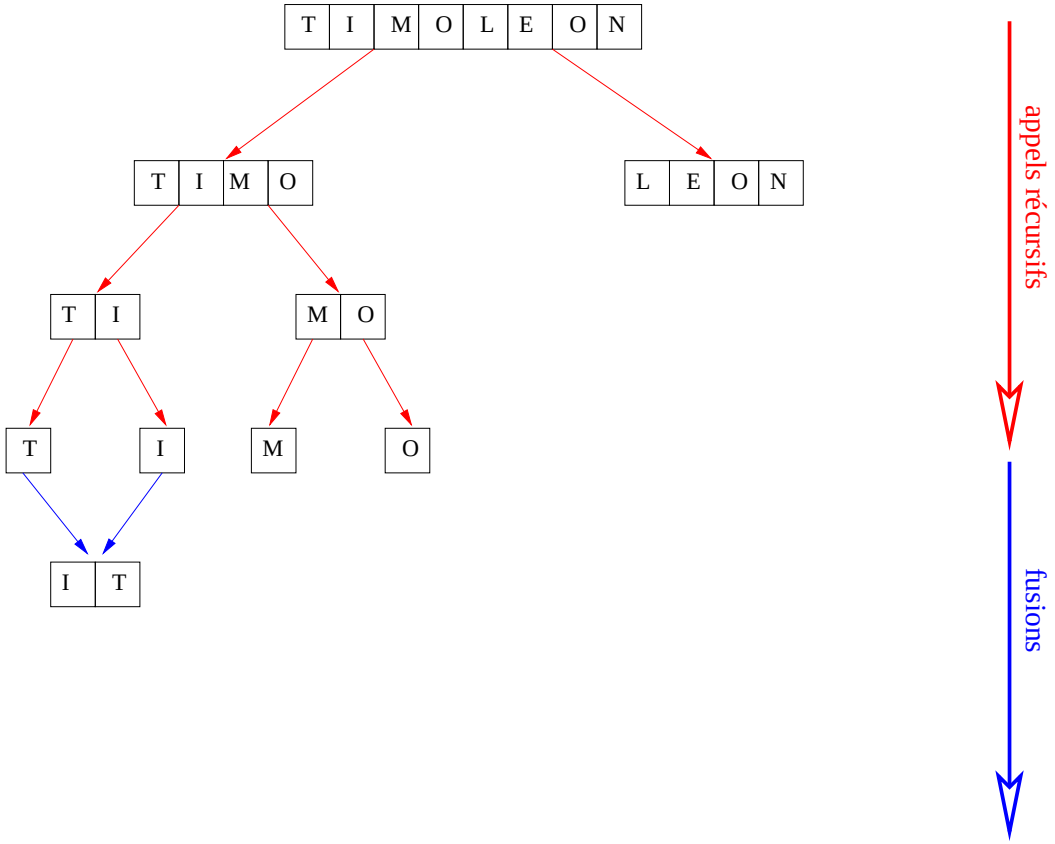
Exemple



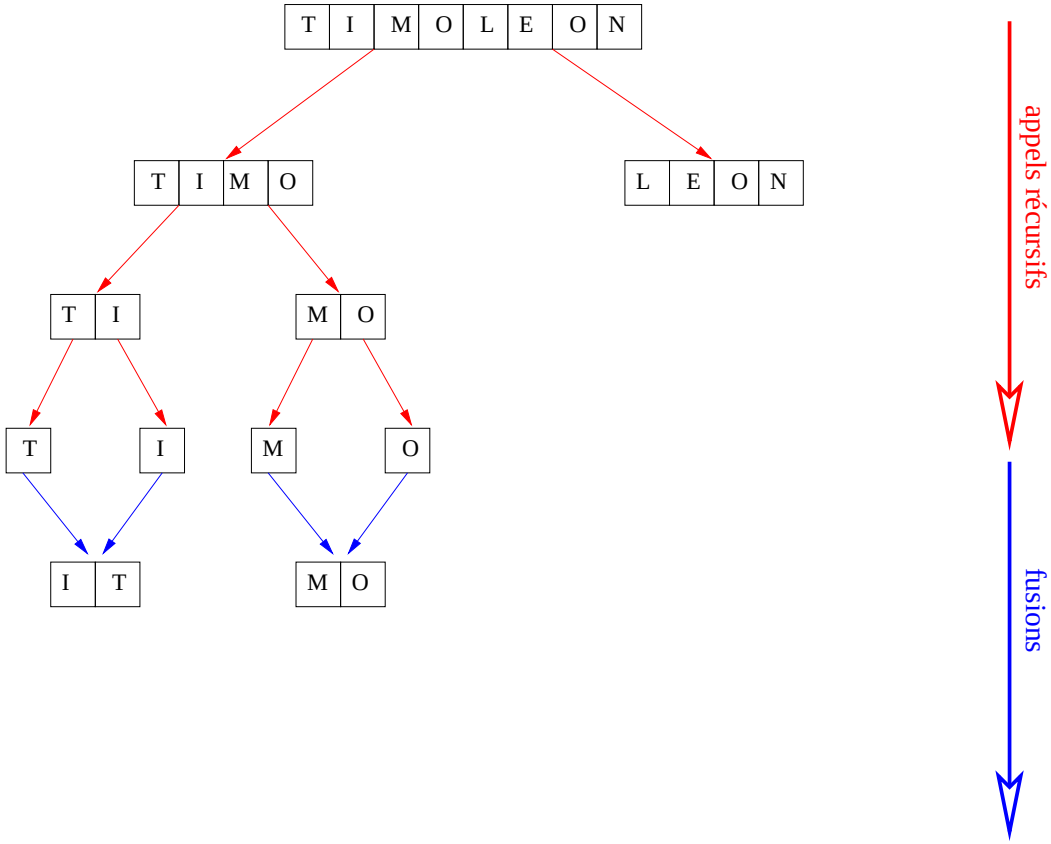
Exemple



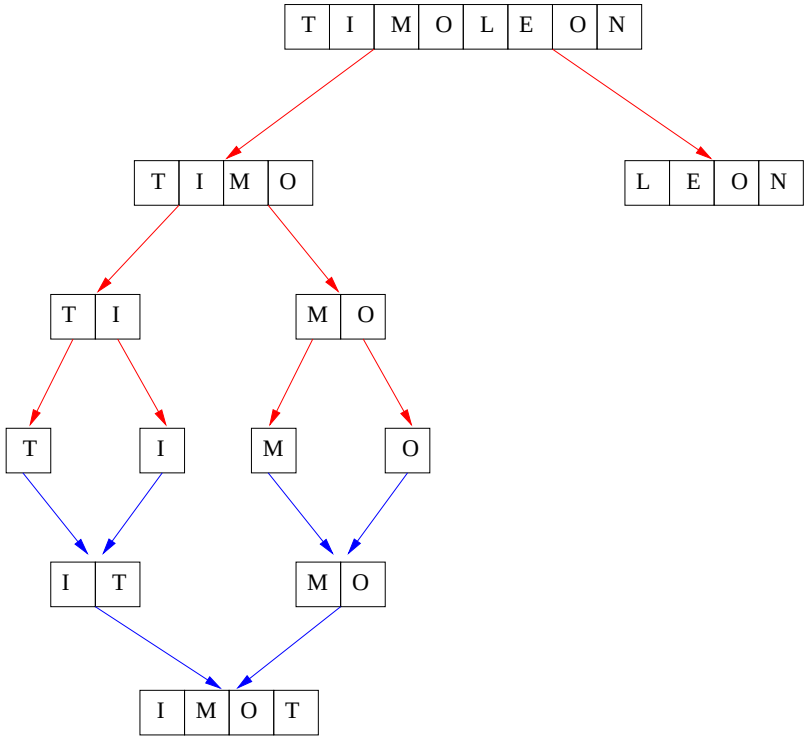
Exemple



Exemple

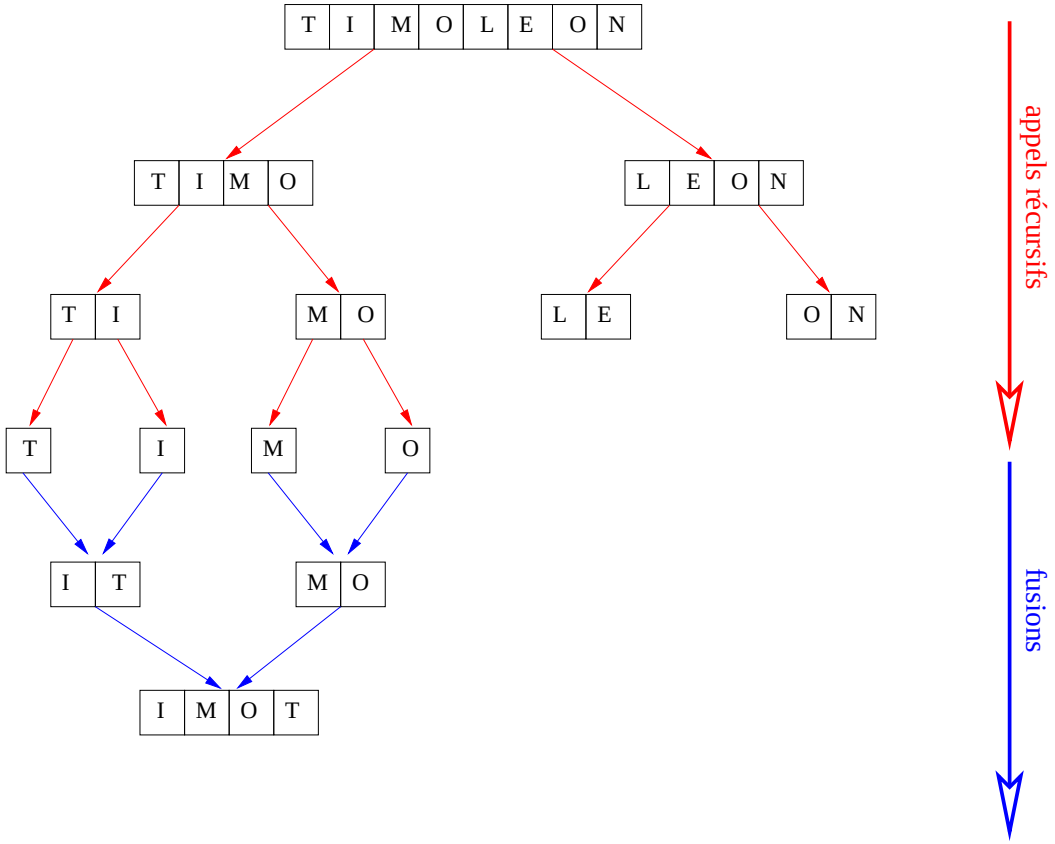

T I M O L E O N
T I M O
L E O N
T I
M O
T
I
M
O
I T
M O
appels récursifs
fusions

Exemple

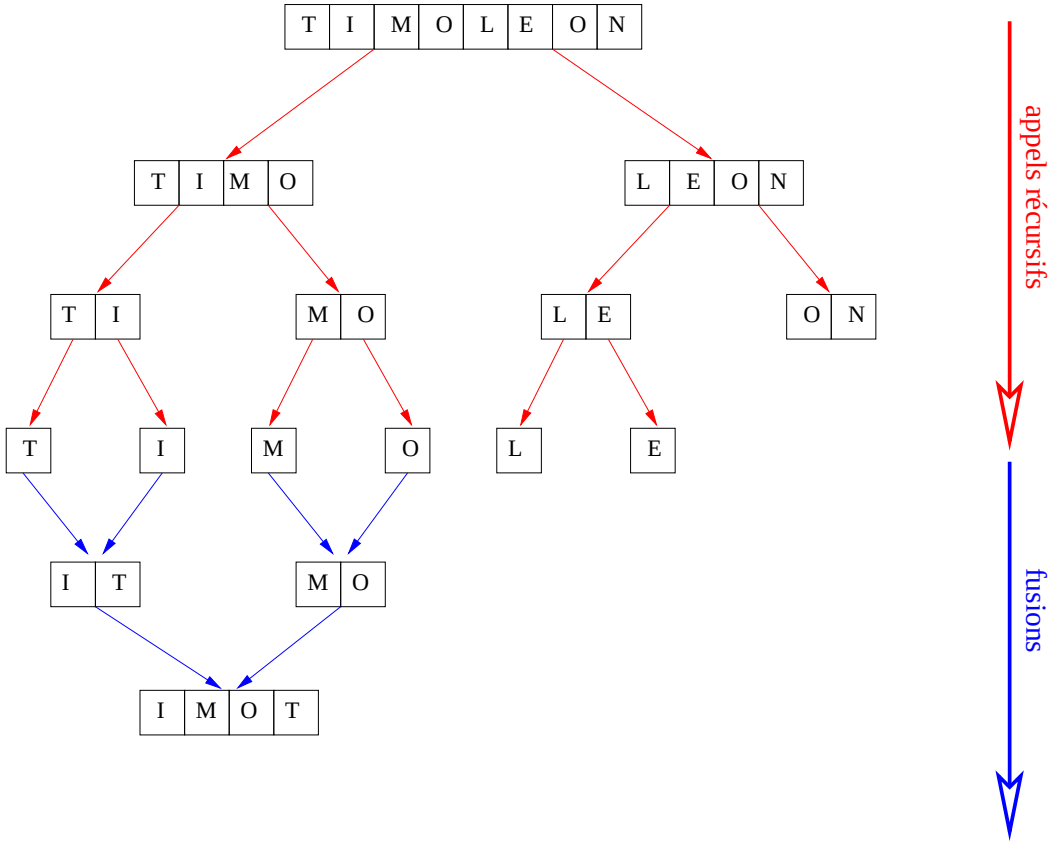

T I M O L E O N
T I M O
L E O N
T I
M O
T
I
M
O
I T
M O
I M O T


appels récursifs
fusions

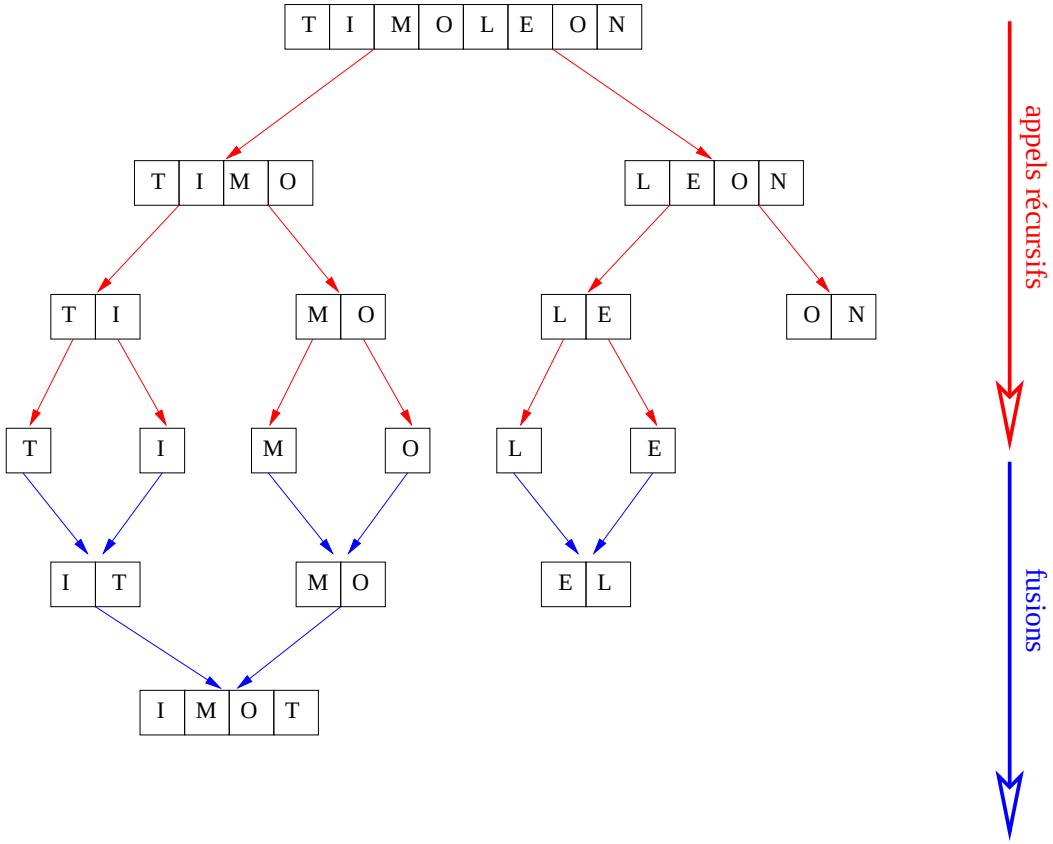
Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
I T
M O
I M O T
appels récursifs
fusions

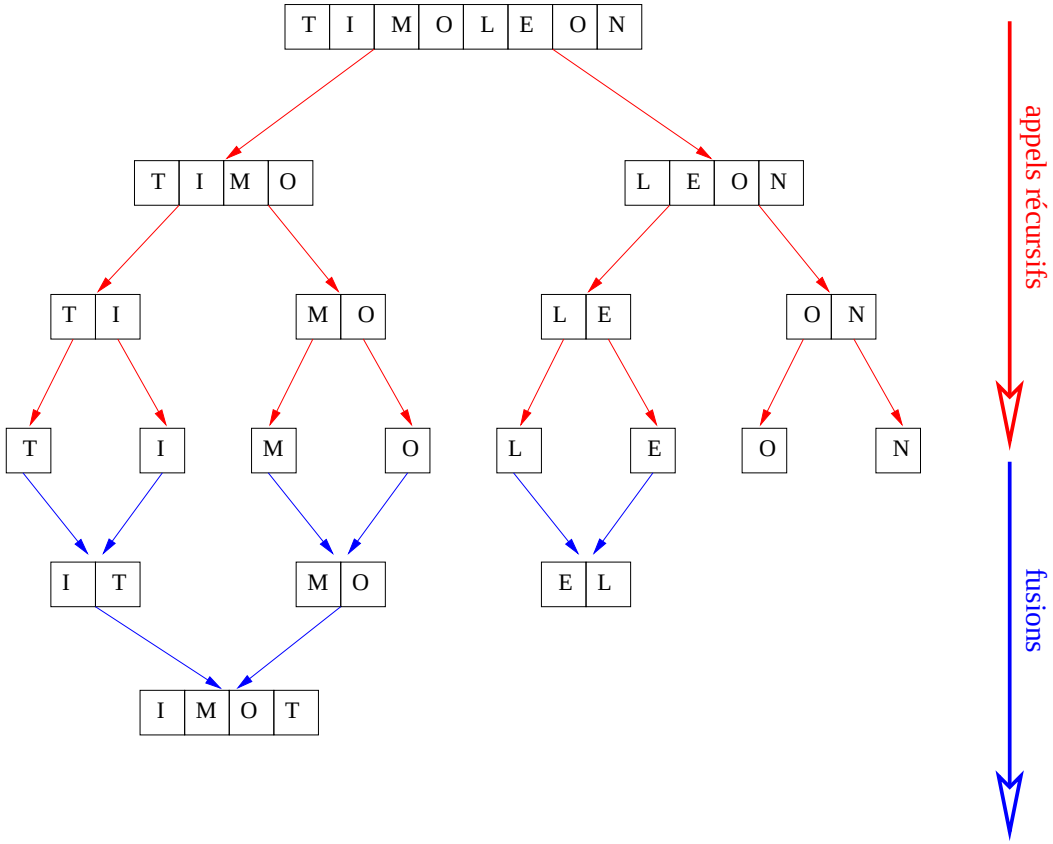
Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
I T
M O
I M O T
appels récursifs
fusions

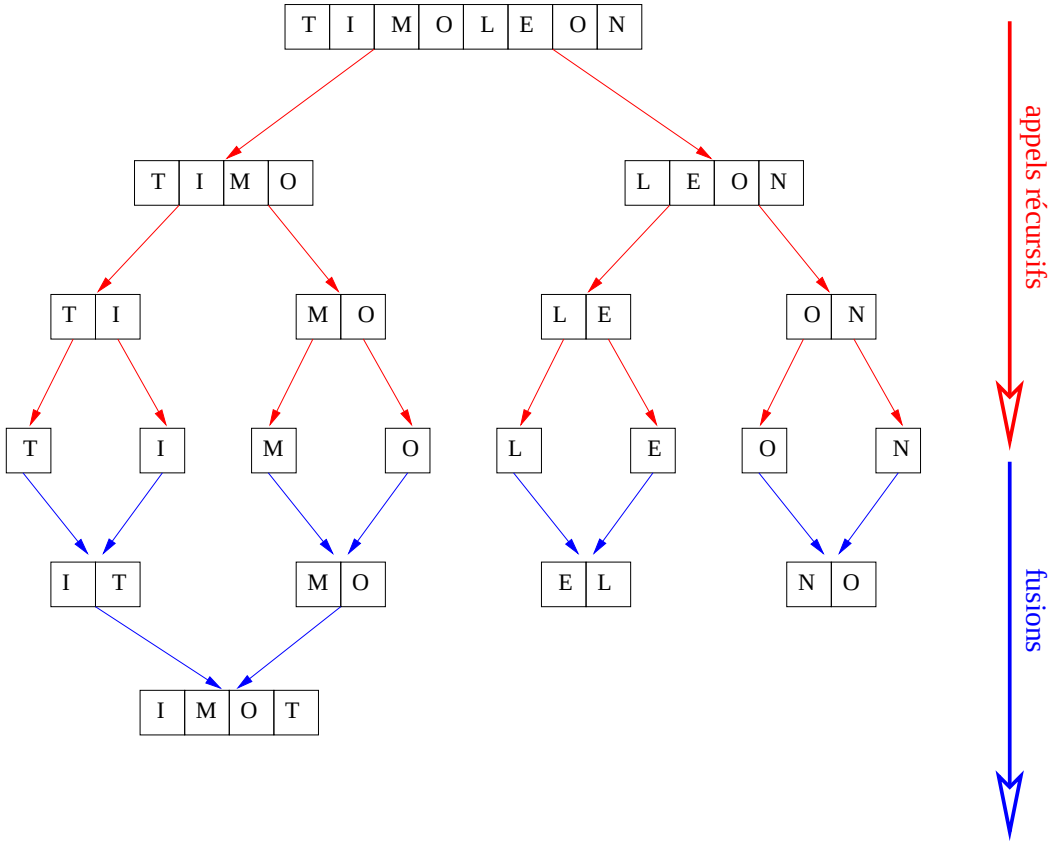
Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
I T
M O
E L
I M O T
appels récursifs
fusions

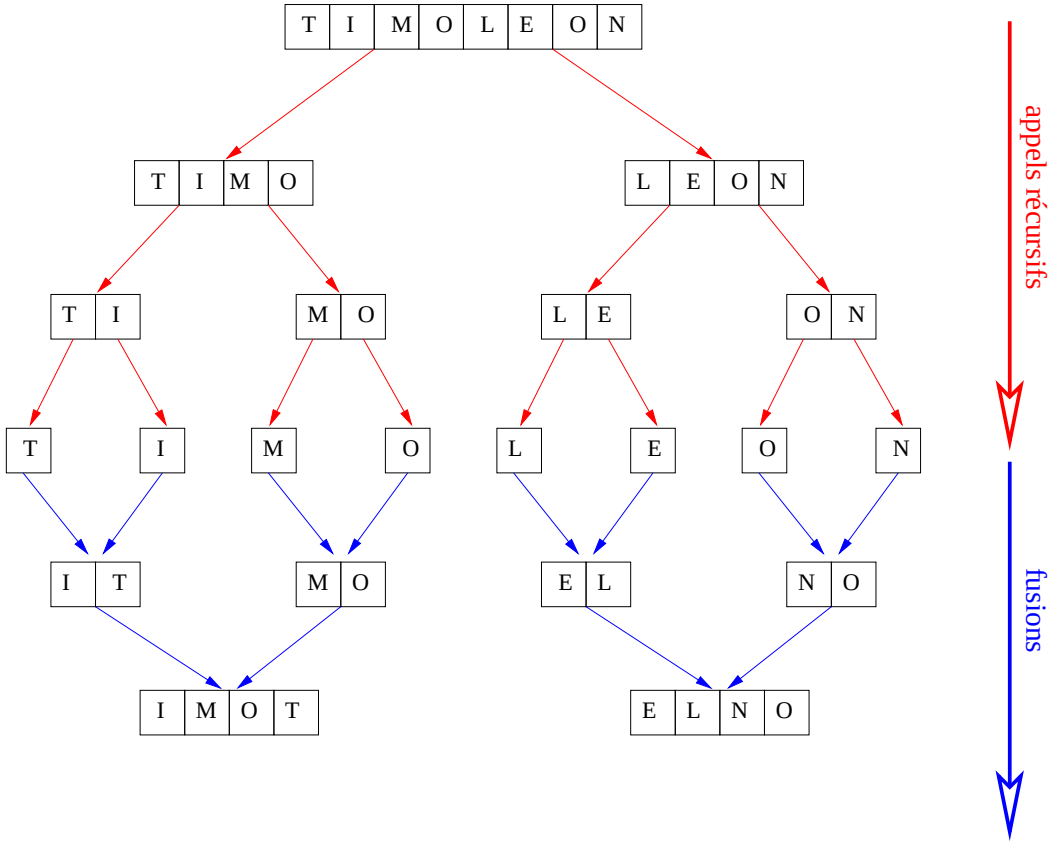
Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
O
N
I T
M O
E L
I M O T
appels récursifs
fusions

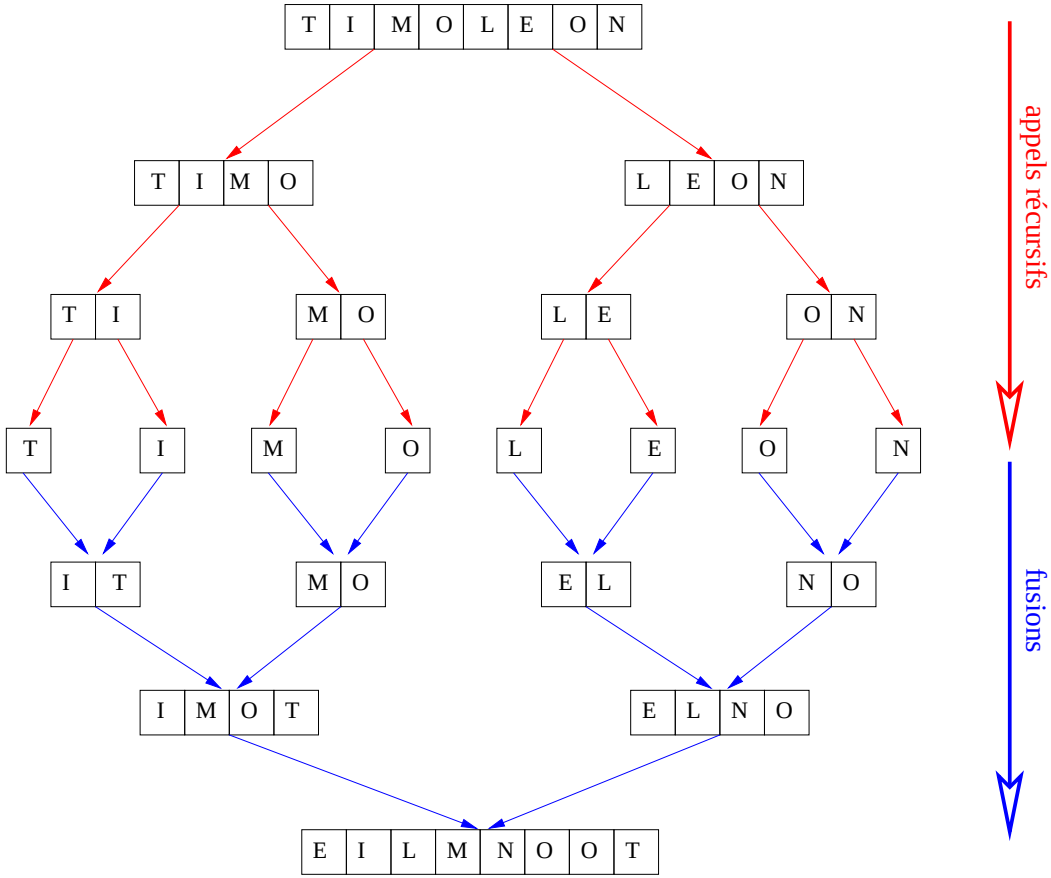
Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
O
N
I T
M O
E L
N O
I M O T
appels récursifs
fusions

Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
O
N
I T
M O
E L
N O
I M O T
E L N O
appels récursifs
fusions

Exemple


T I M O L E O N
T I M O
L E O N
T I
M O
L E
O N
T
I
M
O
L
E
O
N
I T
M O
E L
N O
I M O T
E L N O
E I L M N O O T
appels récursifs
fusions

Coût du tri

Nombre a_n d'affectations pour trier un tableau de taille n

Coût du tri

Nombre a_n d'affectations pour trier un tableau de taille n

- défini de manière récurrente par

$$\begin{aligned} a_1 &= 0 \\ a_n &= a_{\lfloor n/2 \rfloor} + a_{\lceil n/2 \rceil} + 2n \quad \forall n > 1 \end{aligned}$$

Coût du tri

Nombre a_n d'affectations pour trier un tableau de taille n

- défini de manière récurrente par

$$\begin{aligned} a_1 &= 0 \\ a_n &= a_{\lfloor n/2 \rfloor} + a_{\lceil n/2 \rceil} + 2n \quad \forall n > 1 \end{aligned}$$

- lorsque $n = 2^p$ est une puissance de deux, on trouve

$$a_n = 2n \log_2 (n)$$

Coût du tri

Nombre a_n d'affectations pour trier un tableau de taille n

- défini de manière récurrente par

$$\begin{aligned} a_1 &= 0 \\ a_n &= a_{\lfloor n/2 \rfloor} + a_{\lceil n/2 \rceil} + 2n \quad \forall n > 1 \end{aligned}$$

- lorsque $n = 2^p$ est une puissance de deux, on trouve

$$a_n = 2n \log_2(n)$$

- dans le cas général

$$a_n = \theta(n \log_2(n))$$

Coût du tri

Nombre c_n de comparaisons pour trier un tableau de taille n

Coût du tri

Nombre c_n de comparaisons pour trier un tableau de taille n

- Relation de récurrence

$$\begin{aligned} c_1 &= 0 \\ c_n &= c_{\lfloor n/2 \rfloor} + c_{\lceil n/2 \rceil} + c'_n \quad \forall n > 1 \end{aligned}$$

où c'_n = nbre de comparaisons dans la fusion

Coût du tri

Nombre c_n de comparaisons pour trier un tableau de taille n

- Relation de récurrence

$$\begin{aligned} c_1 &= 0 \\ c_n &= c_{\lfloor n/2 \rfloor} + c_{\lceil n/2 \rceil} + c'_n \quad \forall n > 1 \end{aligned}$$

où c'_n = nbre de comparaisons dans la fusion

- Dans le meilleur des cas : $c'_n = \lfloor n/2 \rfloor$

Coût du tri

Nombre c_n de comparaisons pour trier un tableau de taille n

- Relation de récurrence

$$\begin{aligned} c_1 &= 0 \\ c_n &= c_{\lfloor n/2 \rfloor} + c_{\lceil n/2 \rceil} + c'_n \quad \forall n > 1 \end{aligned}$$

où c'_n = nbre de comparaisons dans la fusion

- Dans le meilleur des cas : $c'_n = \lfloor n/2 \rfloor$
- Dans le pire des cas : $c'_n = n - 1$

Coût du tri

Nombre c_n de comparaisons pour trier un tableau de taille n

- Relation de récurrence

$$\begin{aligned} c_1 &= 0 \\ c_n &= c_{\lfloor n/2 \rfloor} + c_{\lceil n/2 \rceil} + c'_n \quad \forall n > 1 \end{aligned}$$

où c'_n = nbre de comparaisons dans la fusion

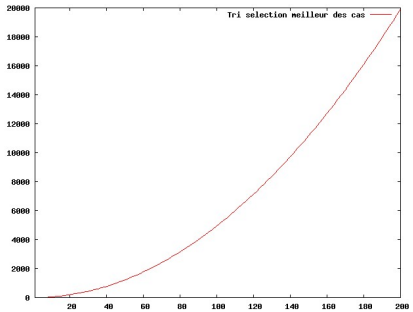
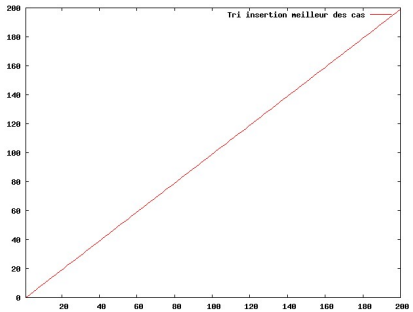
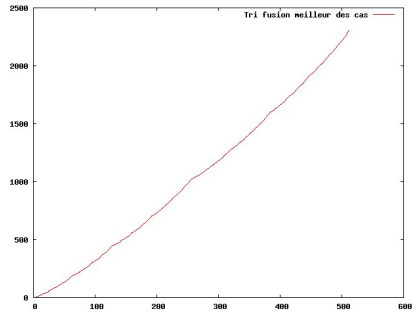
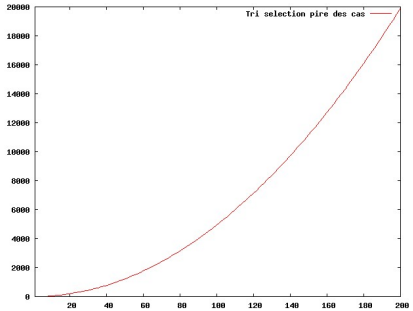
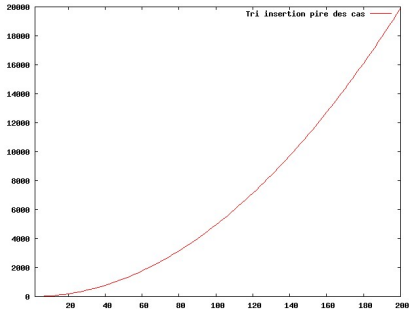
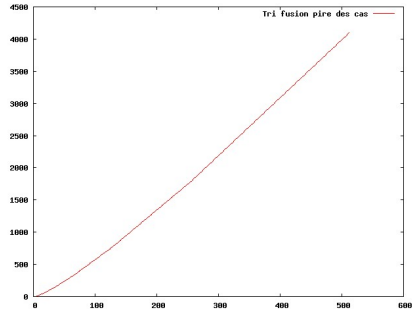
- Dans le meilleur des cas : $c'_n = \lfloor n/2 \rfloor$
- Dans le pire des cas : $c'_n = n - 1$
- Dans tous les cas

$$c_n = \theta(n \log_2(n))$$

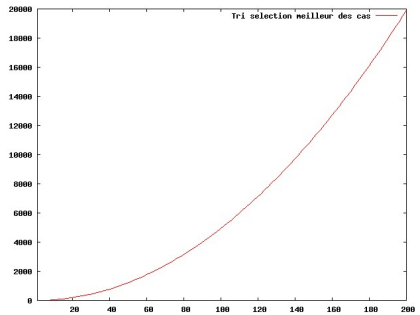
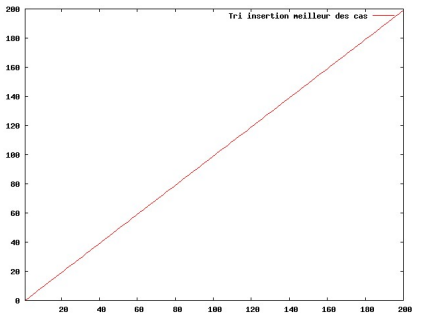
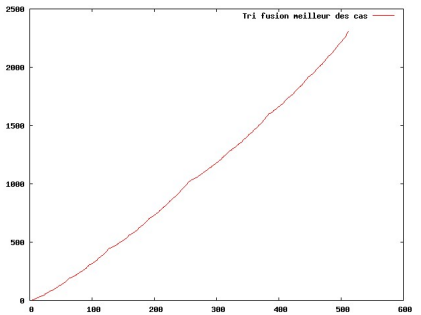
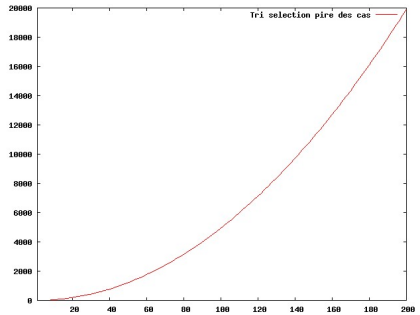
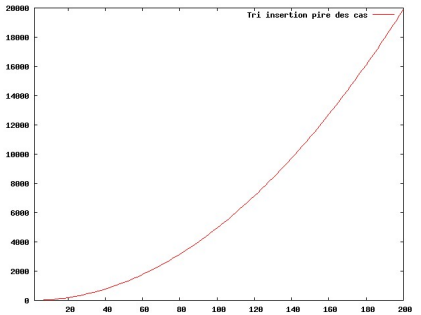
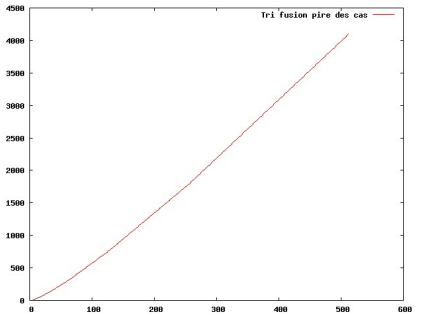
Ordre de grandeur du nombre de comparaisons

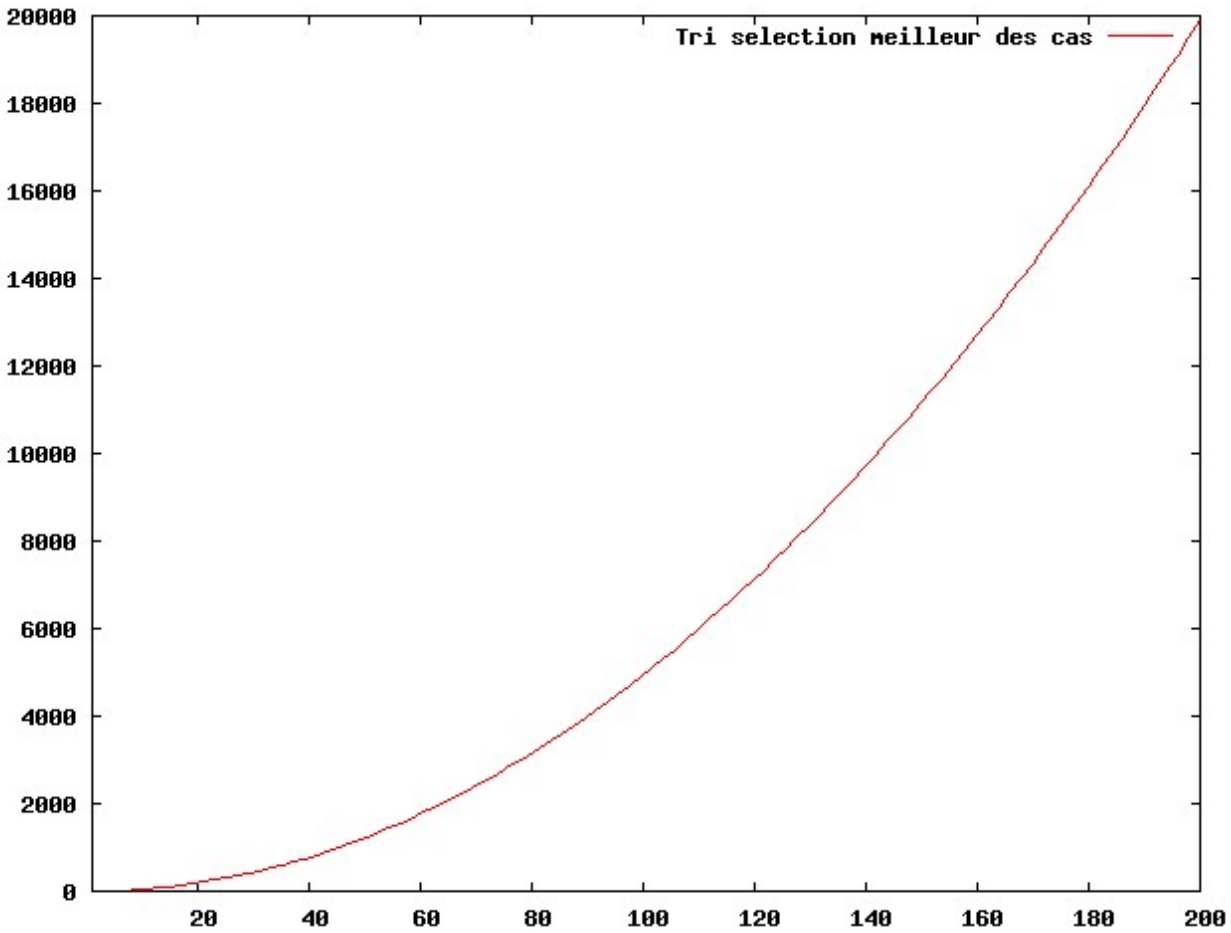
	Tri par sélection	Tri par insertion	Tri par fusion
Meilleur des cas	$\sim \frac{n^2}{2}$	$\sim n$	$\theta(n \log_2 n)$
Pire des cas	$\sim \frac{n^2}{2}$	$\sim \frac{n^2}{2}$	$\theta(n \log_2 n)$

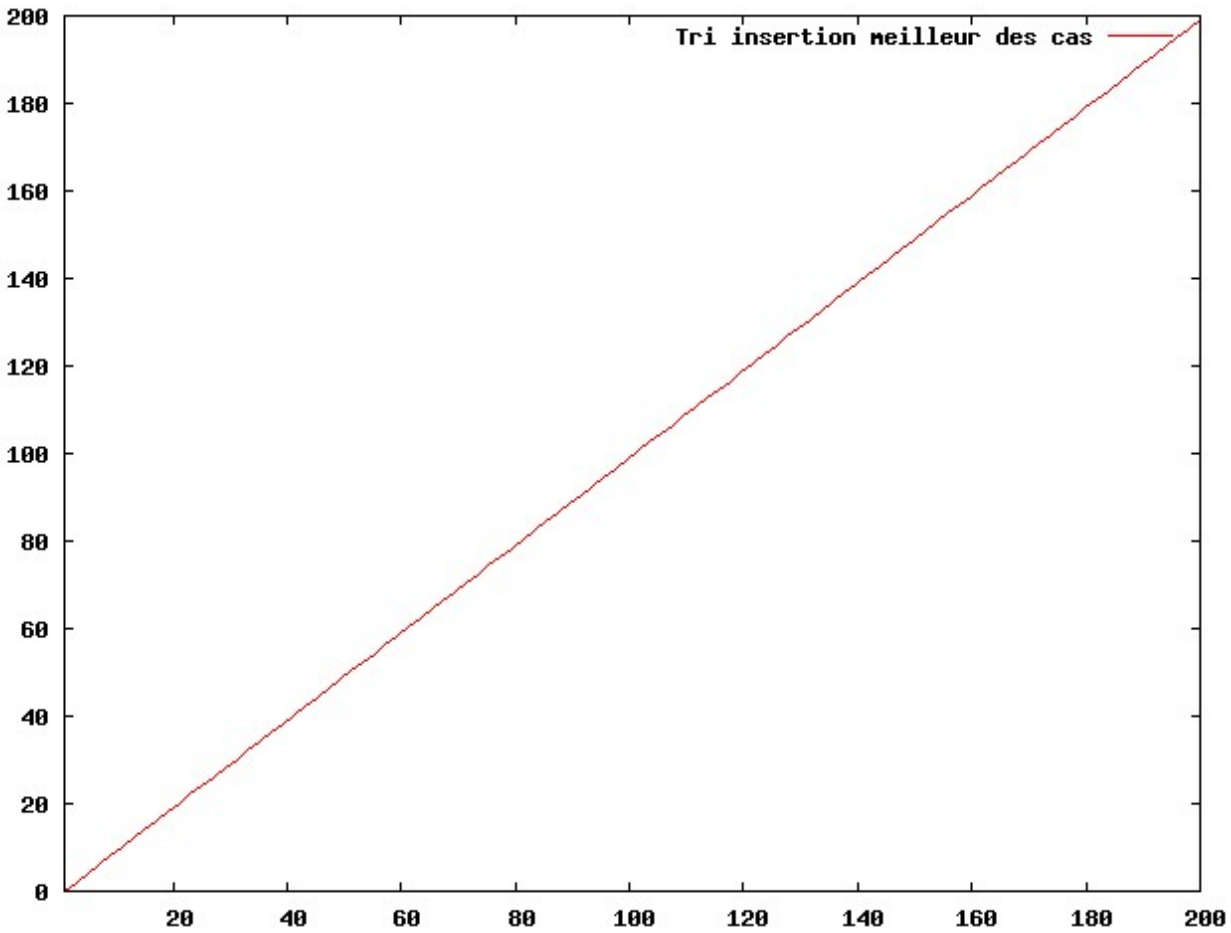
Nombre de comparaisons pour trier des tableaux de taille ≤ 200

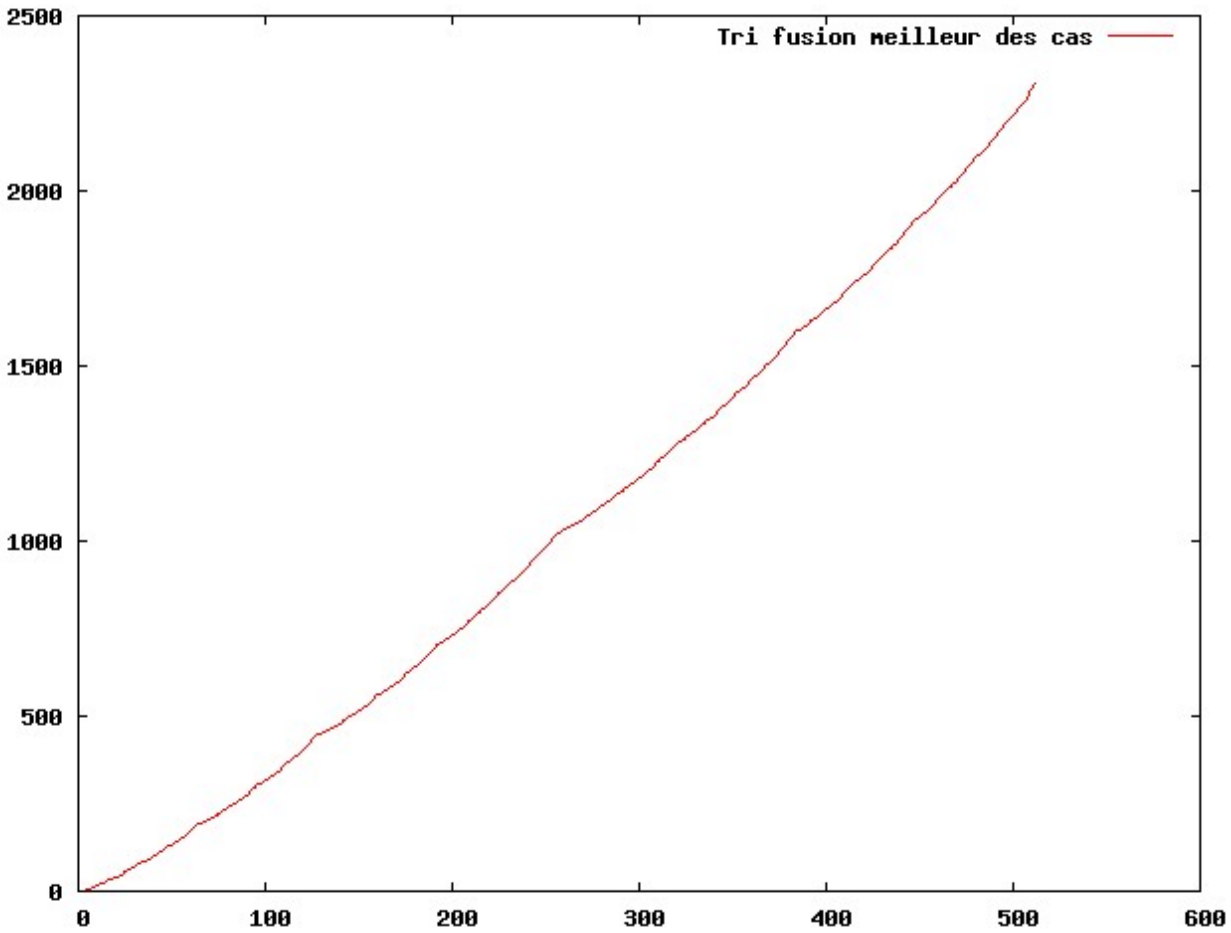
	Tri par sélection	Tri par insertion	Tri par fusion
Meilleur des cas	zoom 	zoom 	zoom 
Pire des cas	zoom 	zoom 	zoom 

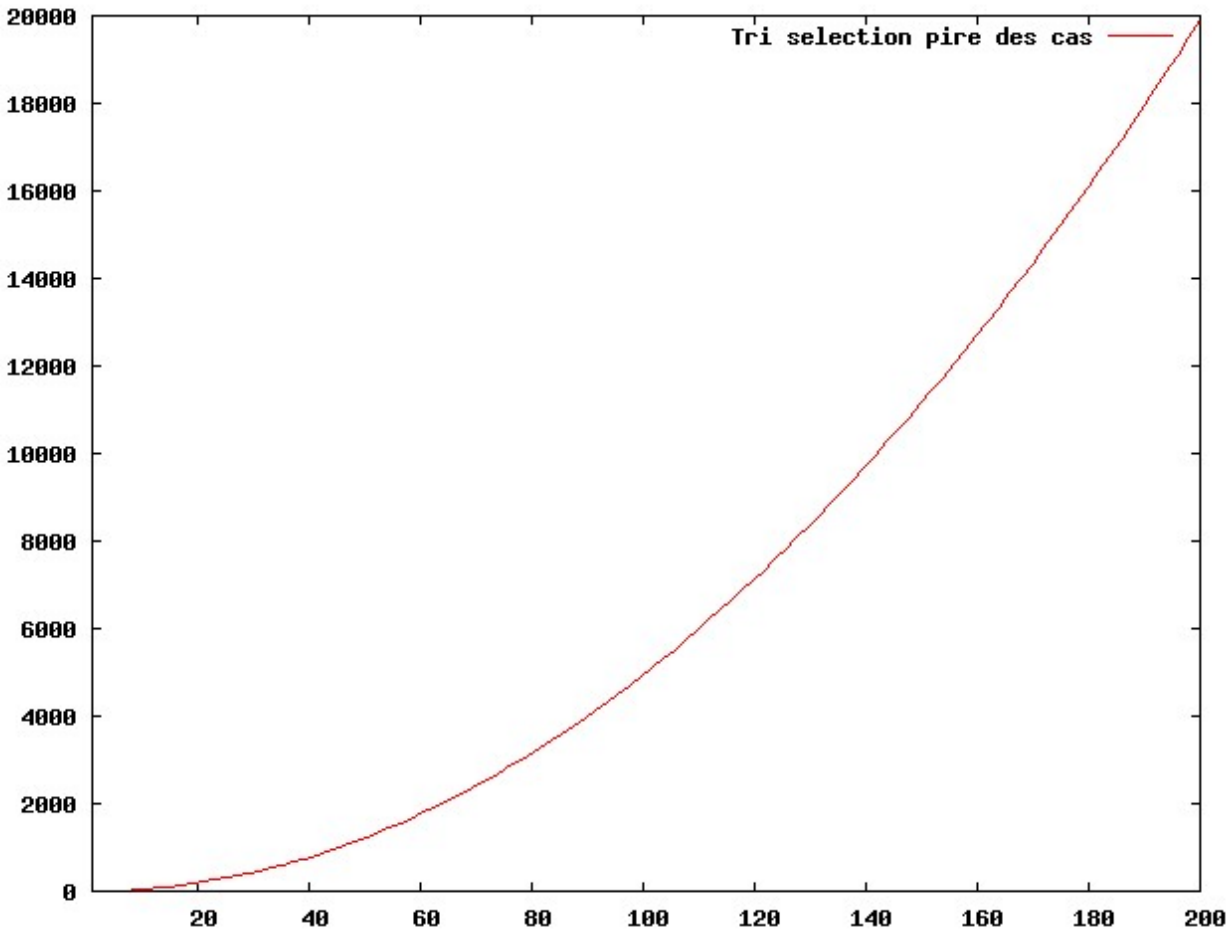
Nombre de comparaisons pour trier des tableaux de taille ≤ 200

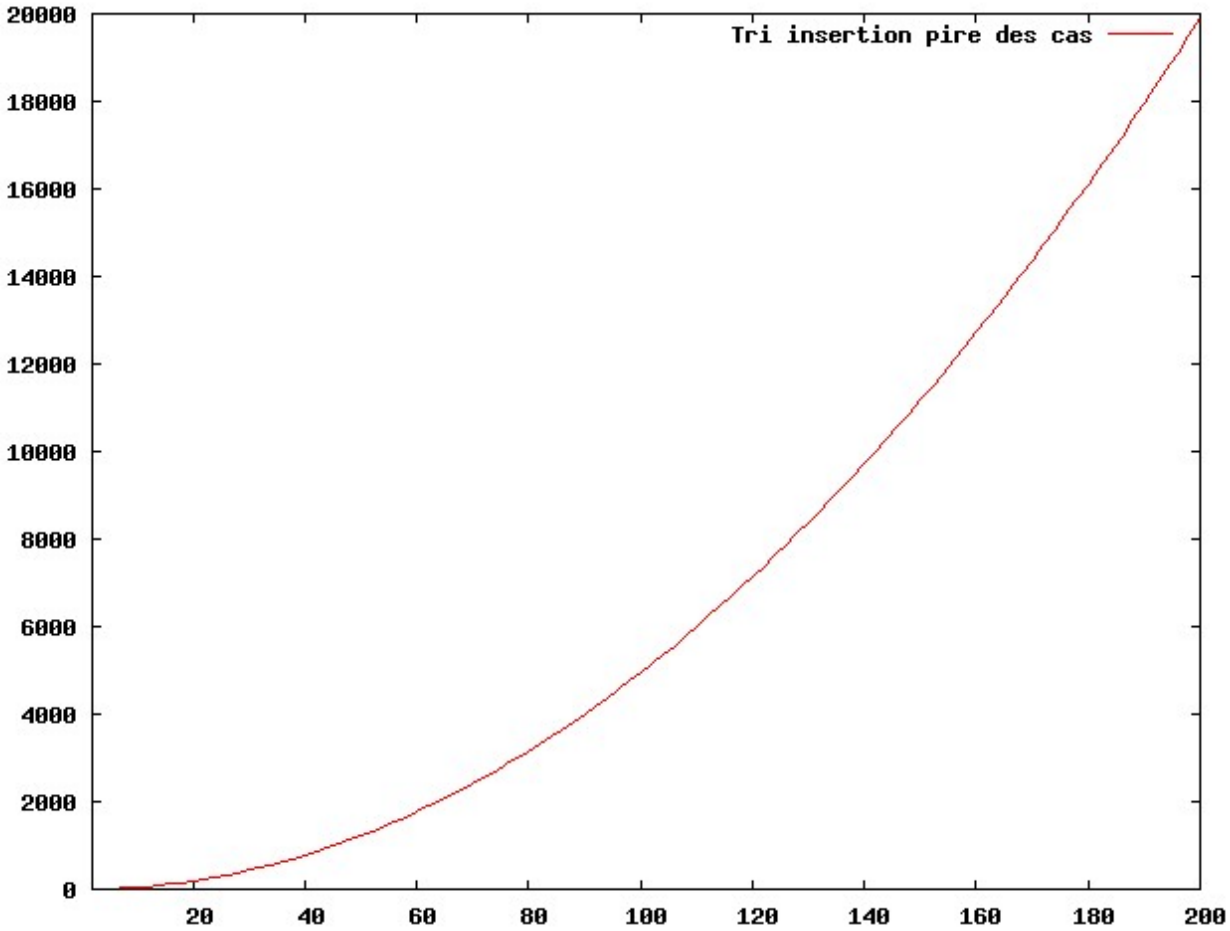
	Tri par sélection	Tri par insertion	Tri par fusion
Meilleur des cas	zoom 	zoom 	zoom 
Pire des cas	zoom 	zoom 	zoom 

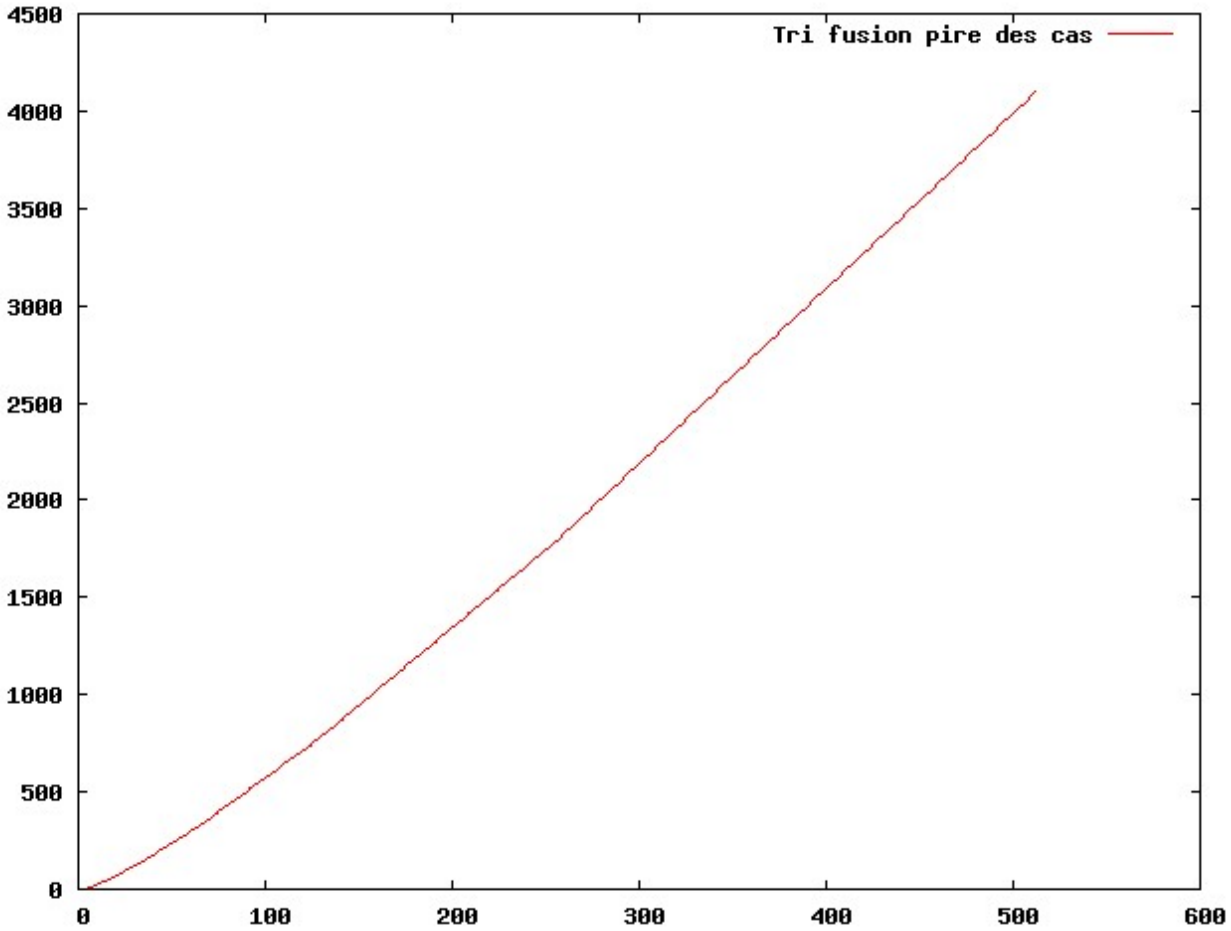

Tri selection meilleur des cas
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
20
40
60
80
100
120
140
160
180
200


Tri insertion meilleur des cas
200
180
160
140
120
100
80
60
40
20
0
20
40
60
80
100
120
140
160
180
200


Tri fusion meilleur des cas
2500
2000
1500
1000
500
0
0
100
200
300
400
500
600


Tri selection pire des cas
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
20
40
60
80
100
120
140
160
180
200


Tri insertion pire des cas
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
20
40
60
80
100
120
140
160
180
200


Tri fusion pire des cas
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
100
200
300
400
500
600