Informatique fondamentale

Tris

R.Gosswiller

1. Recherche d'élément
2. Tri à bulles
3. Tri rapide
4. Tri fusion

Recherche d'élément

Recherche d'élément

Principe

Une grande quantité de problèmes d'informatique sont assimilables à une recherche d'éléments dans un tableau.

Exemples

Gestion d'inventaire, Publicité ciblée, Big Data, Anti-fraude...

Recherche d'élément

Principe

Chercher un élément est plus rapide dans un tableau trié que dans un tableau désordonné
Parcours complet : C(n)
Dichotomie : C(log(n))

Recherche d'élément

Principe

Trier un tableau de plusieurs centaines de milliers de ligne est un problème qui peut vite se compliquer.

Exemples

Contrainte en temps, en mémoire, données fragmentées, ...

Recherche d'élément

Principe

Différentes contraintes aboutissent à différents algorithmes.
Il n'existe pas de méthode de tri 'parfaite'.

Tri à bulles

Tri à bulles

Principe

Trier un tableau en faisant remonter successivement chaque valeur la plus haute

Tri à bulles

Algorithme

```
1  Parcourir i allant de la derniere a la premiere case
2
3      Pour j allant de la case 0 a la case i
4
5          Si T[j+1] < T[j]
6
7              permuter(T[j+1],T[j])
```

Tri à bulles

Cas d'usage

Listes avec peu d'éléments non-triés
Ajout d'un élément dans une liste déjà triée

Complexité

$o(n^2)$
.. avec un des deux n qui dépends du nombre d'éléments à trier

Tri rapide

Quicksort

Principe

Fonctionnement par pivot : placer tous les éléments inférieurs à gauche et supérieurs à droite

Quicksort

Algorithme

```
1   part(tableau tab, entier f, entier l, entier piv)
2       permuter tab[piv] et tab[l]
3       j := f
4       pour i de f a l - 1
5           si tab[i] <= tab[l] alors
6               permuter tab[i] et tab[j]
7               j := j + 1
8       permuter tab[dernier] et tab[j]
9       renvoyer j
10
11   qsort(tableau T, entier f, entier l)
12      si f < l alors
13          piv := choix_pivot(T, f, l)
14          piv := part(T, f, l, piv)
15          qsort(T, f, piv-1)
16          qsort(T, piv+1, l)
```

Quicksort

Cas d'usage

Contraintes de mémoire et de temps
Option de repli

Complexité

$o(n^2)$, $o(n * log(n))$ en cas moyen

Tri fusion

Tri fusion

Principe

Tout tableau trié peut être vu comme la fusion de 2 sous-tableaux triés
Chaque sous-tableau peut être vu comme la fusion de deux sous-tableaux triés
Tout tableau à un seul élément est automatiquement trié

Tri fusion

Algorithme

```
1  triFusion(T)
2    si n<= 1: renvoyer T
3
4    renvoyer fusion(triFusion(T[0...n/2],triFusion(T[n/2....n-1]))
5
6  fusion(Ta,Tb):
7    Si Ta est vide : renvoyer Tb
8    Si Tb est vide : renvoyer Ta
9    Si Ta[0] <= Tb[0] : renvoyer A[0] + fusion(Ta[1...],Tb)
10   Sinon renvoyer B[0] + fusion(Ta,Tb[1...])
```

Tri fusion

Cas d'usage

Fusion de deux grandes listes triées - Multi-thread
Besoin de gagner du temps plutôt que de la mémoire

Complexité

$o(n * log(n))$