Cours d'algorithmique
Tris rapides - EISTI - ING 1

Ecole Internationale des Sciences du Traitement de l'Information

Tri par fusion

Description

- Diviser les éléments en deux parties (équilibrées)
- Trier chaque partie
- Fusionner

Tableaux dynamiques

Définition

Un tableau dynamique est un tableau dont la taille n'est pas une constante mais une variable entière positive pouvant par exemple être calculée à l'exécution.

Syntaxe

En algorithmique, nous utiliserons la même syntaxe que pour les tableaux classiques, en omettant la taille. Nous ajoutons donc une procédure permettant de fixer la taille quand celle ci est connue

`creertableau(ES tableau t():entier,E n:entier)` :

```
variables
  n:entier
  tableau t():entier
  ...
  lire(n)
  creertableau(t,n)
  ...
```

Tri par fusion

Algorithme

```
procedure fusion(ES tableau t(N):entier,d:
      entier,m:entier,f:entier)
variables
  tableau l():entier
  tableau r():entier
  tl,tr,i,j,k:entier
  tl ← m-d+1
  tr ← f-m
  creertableau(l,tl+1)
  creertableau(r,tr+1)
  pour i ← 1 a tl pas 1
    l(i) ← t(d+i-1)
  fpour
  pour j ← 1 a tr pas 1
    r(j) ← t(m+j)
  fpour
  l(tl+1) ← MAXINT
  r(tr+1) ← MAXINT
  i←1
  j←1
  pour k ←d a f pas 1
    si l(i)≤ r(j) alors
      t(k) ← l(i)
      i ← i+1
    sinon
      t(k) ← r(j)
      j ← j+1
    fsi
  fpour
finprocedure
```

```
procedure trifusion(ES:
      tableau t(N):entier,d:
      entier,f:entier)
variables
  m:entier
  si d < f alors
    m ← (d+f)/2
    trifusion(t,d,m)
    trifusion(t,m+1,f)
    fusion(t,d,m,f)
  fsi
finprocedure
```

Tri par fusion

Complexité

```
procedure trifusion(ES: tableau t(N):entier,d:entier,f:entier)
variables
  m:entier
  si d < f alors
    m ← (d+f)/2
    trifusion(t,d,m)
    trifusion(t,m+1,f)
    fusion(t,d,m,f)
  fsi
finprocedure
```

$$T(n) = 2 * T(n/2) + c * n$$
$$T(2^k) = 2 * T(2^{k-1}) + c * 2^k$$
$$= 2 * (2 * T(2^{k-2}) + c * 2^{k-1}) + c * 2^k$$
$$= 2^2 * T(2^{k-2}) + c * 2 * 2^k$$
$$= \dots$$
$$= 2^k * T(1) + c * k * 2^k$$
$$= n + c * n * log_2 n$$

Complexité : $O(nlog_2 n)$ pour trier n éléments

Invariant du tri par fusion

```
i←1 j←1
pour k ←d a f pas 1
  si l(i)≤ r(j) alors t(k)←l(i) i←i+1
  sinon t(k)←r(j) j←j+1
  fsi
fpour
```

- Invariant :
  $t(d \cdots k-1)$ contient les $(k-d)$ plus petits éléments de $l(1 \cdots tl+1)$ et $r(1 \cdots tr+1)$ en ordre trié et $l(i)$,$r(j)$ sont les plus petits élements de l et r non copiés dans t
- Conservation :
  si $l(i) < r(j)$, $l(i)$ est le plus petit élément non copié dans t. Après la copie, $t(d \cdots k)$ contient les $(k-d+1)$ plus petits éléments
- Terminaison :
  si $k = f+1$, $t(d \cdots k-1) = t(d \cdots f)$ contient les $(k-d) = (f-d+1)$ plus petits éléments de l et r triés. Seuls $l(tl+1)$ et $r(tr+1)$ n'ont pas été copiés dans t.

Comparaison des tris

Tri fusion vs tri insertion

- Ordinateur rapide A (1 milliard d'instruction par seconde)
- Ordinateur lent B (10 millions d'instructions par seconde)
- Tri d'un tableau de 1 million de nombres
- Bon programme de tri insertion ($c_1 = 2$)
- Mauvais programme de tri fusion ($c_2 = 50$)
- Comparer les temps d'exécution des deux ordinateurs pour effectuer le tri
- A partir de quelle taille B est plus rapide que A

Tri rapide

Description

- Choisir un élément pivot du tableau (une méthode de base choisit le premier ou le dernier)
- Diviser le tableau en trois parties : ceux plus petits que le pivot, ceux égaux, ceux plus grands.
- Appliquer la même méthode aux première et troisième parties

Tri rapide

Algorithme

```
fonction partition(ES tableau t(N):entier,d:entier,f:entier):entier
  variables
    x:entier
  x ← t(f)
  i ← d−1
  pour j← d a f−1 pas 1
    si t(j) ≤ x alors
      i ← i+1
      swap(t(i),t(j))
    fsi
  fpour
  swap(t(i+1),t(f))
  retourner i+1
finfonction
```

```
procedure trirapide(ES tableau t(N):entier,d:entier,f:entier)
  variables
    m:entier
  si d<f alors
    m ← partition(t,d,f)
    trirapide(t,d,m−1)
    trirapide(t,m+1,f)
  fsi
finprocedure
```

Invariant de la partition

```
i ← d−1
pour j← d a f−1 pas 1
  si t(j) ≤ x alors
    i ← i+1
    swap(t(i),t(j))
  fsi
fpour
```

- Invariant :
  1. Si $d \leq k \leq i$ alors $t(k) \leq x$
  2. Si $i+1 \leq k \leq j-1$ alors $t(k) > x$
  3. Si $k = f$ alors $t(k) = x$
- Initialisation : $i = d-1$ et $j = d$, cas trivial
- Conservation :
  - Si $t(j) > x$, j est incrémenté : 2. ok pour $t(j-1)$
  - Si $t(j) \leq x$, i est incrémenté et $t(i)$,$t(j)$ sont permutés puis j est incrémenté. Donc $t(i) \leq x$: 1. ok. On a également $t(j-1) > x$ d'après l'invariant.
- Terminaison : $j = f$ cqfd