Tri faire-valoir

Alain Cournier

Université de Picardie

Licence 3 informatique

Comprendre le problème

- On souhaite trié dans l'ordre croisant un tableau de valeurs T indicé de l'indice d à l'indice f selon un ordre total ≤.

- La spécification précise à été donnée en TD (cf TD 1)

Principe

Le principe du tri faire-valoir est le suivant :

a) Si T contient 0 ou 1 case : C’est facile

b) Si T contient 2 cases On échange le premier et le dernier élément du tableau à trier s'ils ne sont pas dans le bon ordre.

c) Si le tableau contient plus de trois éléments :
   - on trie (récursivement) les deux premiers tiers du tableau ;
   - on trie (récursivement) les deux derniers tiers du tableau ;
   - on trie (récursivement) à nouveau les deux premiers tiers du tableau.

Exemple

1	2	3	4	5	6	7	8	9	10	11	12
P	a	i	n	t	i	t	b	l	a	c	k

1	2	3	4	5	6	7	8	9	10	11	12
P	a	i	n	t	i	t	b	l	a	c	k
a	b	i	i	n	P	t	t	l	a	c	k

1	2	3	4	5	6	7	8	9	10	11	12
a	b	i	i	n	P	t	t	l	a	c	k
a	b	i	i	a	c	k	l	n	P	t	t

1	2	3	4	5	6	7	8	9	10	11	12
a	b	i	i	a	c	k	l	n	P	t	t
a	a	b	c	i	i	k	l	n	P	t	t

Code

```
Procédure TriFaireValoir(T, d, f)
    Si f=d+1 et T[d] > T[f] alors échanger T[d] et T[f]
    SinonSi ((f – d) + 1) > 2 alors
        Inter = ((f – d) + 1) div 3
        TriFaireValoir(T, d  , f-Inter)
        TriFaireValoir(T, d+Inter, f  )
        TriFaireValoir(T, d  , f-Inter)
    FinSi
```

TriFaireValoir s’arrête

- Informellement, nous voulons établir : pour tout tableau de valeurs T et pour tout couple d’indices du tableau T, (d, f), TriFaireValoir(T, d, f) termine son exécution en un temps fini.

- Problème cette formulation n’est pas adéquate pour une preuve par récurrence.

TriFaireValoir s'arrête

- ArretTriFaireValoir(k) : Pour tout tableau T et tout couple d'indice (d,f), (f-d)+1 = k (k cases) implique que TriFaireValoir(T, d, f) termine son exécution en un temps fini.

- Pour tout k < 2, TriFaireValoir(T, d, f) n'exécute aucune instruction avant de renvoyer T. Donc TriFaireValoir(T, d, f) se termine en un temps fini.
- Pour k = 2 TriFaireValoir(T, d, f) exécute au plus un échange avant de renvoyer T. Donc TriFaireValoir(T, d, f) se termine en un temps fini.

TriFaireValoir s’arrête

On veut démontrer que pour tout entier naturel u :

ArretTriFaireValoir(0) et … et ArretTriFaireValoir(u) et ArretTriFaireValoir(u+1) et ArretTriFaireValoir(u+2) prouve ArretTriFaireValoir(u+3)

Soit u un entier naturel

Remarque 1 : Puisque u est un entier naturel $(f-d)+1 = u+3 \geq 3$.

Remarque 2 : $1 \leq ((f-d)+1) \text{ div } 3 = Inter \leq f-d$

Remarque 3 : TriFaireValoir(T, d, f-Inter) se termine.

Preuve : $f-Inter < f$ (car $1 \leq Inter$) donc $((f-Inter)-d)+1 < (f-d)+1 = u+3$ donc ArretTriFaireValoir $(((f-Inter)-d)+1)$ est vrai

TriFaireValoir s'arrête

Remarque 4 : TriFaireValoir(T, d+Inter, f) se termine.

Preuve : d+Inter > d (car 1 ≤ Inter) donc ((f-(d+Inter))+1 < (f-d)+1 = u+3 donc ArretTriFaireValoir ((f-(d+Inter))+1) est vrai

Remarque 5 : Les trois appels récursifs se terminent

Conséquence : Si (f-d)+1 = u+3, TriFaireValoir(T, d, f) se termine.

TriFaireValoir : Bon Résultat

- Informellement, nous voulons établir : pour tout tableau de valeurs T et pour tout couple d'indices du tableau T, (d, f), TriFaireValoir(T, d, f) renvoie un tableau trié entre les deux indices d et f.

- Problème cette formulation n'est pas adéquate pour une preuve par récurrence.

TriFaireValoir : Bon Résultat

- BonResTriFaireValoir(k) : Pour tout tableau T et tout couple d'indice (d,f), (f-d)+1 = k (k cases) implique que TriFaireValoir(T, d, f) renvoie un tableau T trié entre les indices d et f.

- Pour tout k < 2, TriFaireValoir(T, d, f) n'exécute aucune instruction avant de renvoyer T. Or un tableau ayant au plus une case est toujours trié. Donc TriFaireValoir(T, d, f) renvoie un tableau T trié entre les indices d et f.
- Pour k = 2 TriFaireValoir(T, d, f) Si les 2 éléments sont en désordre on échange avant de renvoyer T. Donc TriFaireValoir(T, d, f) renvoie un tableau T trié entre les indices d et f.

TriFaireValoir : Bon Résultat

On veut démontrer que pour tout entier naturel u :

BonResTriFaireValoir(0) et … et BonResTriFaireValoir(u) et BonResTriFaireValoir(u+1) et BonResTriFaireValoir(u+2) prouve BonResTriFaireValoir(u+3)

Soit u un entier naturel

Remarque 1 : Puisque u est un entier naturel $(f\text{-}d)+1 = u+3 \geq 3$.

Remarque 2 : $1 \leq ((f\text{-}d)+1)$ div $3 = Inter \leq f\text{-}d$

Remarque 3 : TriFaireValoir(T, d, f-Inter) renvoie un tableau trié entre les indices d et f-inter.

Preuve : $f\text{-}Inter < f$ (car $1 \leq Inter$) donc $((f\text{-}Inter)\text{-}d)+1 < (f\text{-}d)+1 = u+3$ donc BonResTriFaireValoir $(((f\text{-}Inter)\text{-}d)+1)$ est vrai.

TriFaireValoir : Bon Résultat

Remarque 4 : BonResTriFaireValoir(T, d, f-Inter) renvoie un tableau trié entre les indices d et f-inter.

Preuve : d+Inter > d (car 1 ≤ Inter) donc ((f-(d-Inter))+1 < (f-d)+1 = u+3 donc BonResTriFaireValoir ((f-(d-Inter))+1) est vrai

Remarque 5 : Les trois appels récursifs renvoient le résultat attendu.

Il manque un argument pour lier les 3 résultats partiels.

Exemple : Rappel

1	2	3	4	5	6	7	8	9	10	11	12
P	a	i	n	t	i	t	b	l	a	c	k

1	2	3	4	5	6	7	8	9	10	11	12
P	a	i	n	t	i	t	b	l	a	c	k
a	b	i	i	n	P	t	t	l	a	c	k

1	2	3	4	5	6	7	8	9	10	11	12
a	b	i	i	n	P	t	t	l	a	c	k
a	b	i	i	a	c	k	l	n	P	t	t

1	2	3	4	5	6	7	8	9	10	11	12
a	b	i	i	a	c	k	l	n	P	t	t
a	a	b	c	i	i	k	l	n	P	t	t

TriFaireValoir : Bon Résultat

Remarque 6 : Il faut impérativement que la taille de la zone qui subit les 3 tris récursifs soit plus grande que ou égale à la taille des zones bleues.

Preuve : Dans le pire des cas aucun des plus grands élément n'est dans les 2 derniers tiers du tableau. Il faut donc que la zone qui a subi les deux premiers tris (second tier) soit de taille supérieure ou égale à la taille du troisième tier. Pour que cette zone puisse elle-même accueillir les éléments maximaux à l'issu du premier tri, il faut que la zone qui a subi les deux premiers tris (second tier) soit de taille supérieure ou égale à la taille du premier tier.

Explication

1	2	3	4	5	6	7
W	O	R	K	I	N	G

1	2	3	4	5	6	7
W	O	R	K	I	N	G
K	O	R	W	I	N	G

1	2	3	4	5	6	7
K	O	R	W	I	N	G
K	O	R	G	I	N	W

1	2	3	4	5	6	7
K	O	R	G	I	N	W
G	K	O	R	I	N	W

Explication

1	2	3	4	5	6	7
W	O	R	K	I	N	G

1	2	3	4	5	6	7
W	O	R	K	I	N	G
I	K	O	R	W	N	G

1	2	3	4	5	6	7
I	K	O	R	W	N	G
I	K	G	N	O	R	W

1	2	3	4	5	6	7
I	K	G	N	O	R	W
G	I	K	N	O	R	W

TriFaireValoir : Bon Résultat

- Puisque nous trions récursivement les 2/3 (arrondi à l’entier supérieur) du tableau la zone du tableau qui subit les 3 tris est de taille supérieure ou égale à celles qui n’en subissent qu’un ou deux.

- En conséquences : BonResTriFaireValoir(u+3) est vrai

TriFaireValoir : Complexité

Equations de récurrence (n nb de cases):

CoutTFV (0) = CoutTFD (1) = 6

CoutTFV (2) = 8

Pour $n \geq 3$:

CoutTFV (n) = 10 + 3 CoutTFV((2/3) n)

Pas d'intuition : On remplace

TriFaireValoir : Complexité

Equations de récurrence (n nb de cases):

Pour $n \geq 3$:

$CoutTFV(n) = 10 + 3\ CoutTFV((2/3)\ n)$

$CoutTFV(n) = 10 + 3\ (10 + 3\ CoutTFV((2/3)^2\ n))$

$CoutTFV(n) = 10(1+3) + 3^2\ CoutTFV((2/3)^2\ n))$

Hypothèse :

$CoutTFV(n) = 10(1+3+\ldots+ 3^{i-1}) + 3^i\ CoutTFV((2/3)^i\ n))$

On vérifie par récurrence.

TriFaireValoir : Complexité

Equations de récurrence (n nb de cases):

Pour $n \geq 3$: Les log sont en base 3/2

CoutTFV (n) = $10(1+3+\ldots+3^{i-1}) + 3^{i}$ CoutTFV($(2/3)^{i}$ n))

CoutTFV (n) = $10\ (1+3+\ldots+3^{\log n})$

CoutTFV (n) = $10\ (3^{1+\log n}-1)/(3-1)$

CoutTFV (n) Appartient à $\Theta\ (n^{1/\log 3/2})$ soit $\Theta\ (n^{2,71})$

Code

```
Procédure Mystère (A)
Donnée/Résultat A un tableau de valeurs indicé par des entiers
Variable u, v : deux indices de T; Pu, Pv : Piles d’entiers; t : Entier
Pu ← PileVide(); Pv← PileVide();
Empiler(Pu, PremierIndice(T)); Empiler(Pv, DernierIndice(T))
TantQue Non (TestPileVide(Pv)) faire
    u ← Sommet(Pu); Dépiler(Pu); v ← Sommet(Pv); Dépiler(Pv);
    Si u<v et T[u] > T[v] alors échanger T[u] et T[v] finsi
    Si (u – v) > 1 alors
        t ← (2*((v – u) + 1)+1) div 3
        Empiler(Pu, u); Empiler(Pu,1+v-t); Empiler(Pu, u);
        Empiler(Pv, u+t-1); Empiler(Pv,v); Empiler(Pv, u+t-1);
    FinSi
FinTq
```