Université d'Aix-Marseille III - Faculté des Sciences et Techniques

I6 – Logique, preuve et complexité - Février 2007

Projet semestriel

Algorithme de tri par Arbres binaires de recherche

Le tri présenté ici s'appuie sur la notion d'arbre binaire de recherche (ABR). Dans cette méthode, nous allons dans un premier temps mémoriser les valeurs à trier dans un ABR. Un ABR est une structure de données qui représente des valeurs dans des éléments appelés nœuds, qui possèdent 0, 1 ou 2 nœuds fils. On distingue la notion de fils gauche et celle de fils droit. Etant donné un nœud, s'il possède un fils gauche (respectivement droit), alors nécessairement, la valeur de celui-ci sera inférieure (respectivement supérieure). On utilisera les déclarations suivantes pour représenter un ABR :

```
typedef struct noeud {   int elt; /* valeur du noeud */
                         struct noeud *fg, *fd; /* pointeurs vers fils */
}triplet;
typedef triplet *ABR;
```

Si l'on prend par exemple l'ensemble d'entiers {10, 14, 6, 9, 13, 18, 16, 7, 22}, il existe plusieurs ABR pour le représenter, mais dans tous les cas, quand il est construit (faites-le à la main), il est simple de lister dans l'ordre croissant les éléments le composant : il suffit de parcourir l'ABR, on considérant d'abord le fils gauche (inférieur), puis le nœud courant, et enfin le fils droit (supérieur). Un tel parcours est naturellement récursif. A partir de là, on dispose d'une méthode de tri.

Algorithme de tri par sélection du plus petit élément

Dans ce problème, nous étudions des arbres binaires noté A_N définis à partir d'un entier $N > 0$ et dont les nœuds sont les entiers $\{1, 2, \ldots N\}$ avec pour racine le nœud 1. Le nœud i est tel que :

- c'est une feuille si $2i > N$
- il possède deux fils $\{2i, 2i+1\}$ si $2i+1 \leq N$
- il ne possède qu'un fils $\{2i\}$ si $2i = N$

L'ensemble des nœuds d'un arbre A_N peut être décomposé entre nœuds de niveau pair, et nœuds de niveau impair comme suit :

- la racine est de niveau pair
- les fils d'un nœud de niveau pair sont de niveau impair
- les fils d'un nœud de niveau impair sont de niveau pair

Question 1. Dessinez l'arbre A_{12} qui correspond donc à $N = 12$.

Question 2. Définissez la fonction C appelée `parite`, qui prend en entrée deux valeurs N et i, qui vérifient $1 \leq i \leq N$ et dont le résultat vaut 1 si le niveau de i dans l'arbre est pair, et 0 sinon. Donnez la

complexité de cette fonction.

Question 3. Dans un tel arbre, l'ensemble des descendants d'un nœud feuille est réduit à cette feuille, l'ensemble des descendants d'un nœud est composé de lui-même ainsi que des descendants de ses fils. Définissez la fonction C appelée `descendants`, qui prend en entrée deux valeurs N et i, qui vérifient $1 \leq i \leq N$ et qui affiche l'ensemble des descendants du nœud i dans l'arbre A_N. Donnez la complexité de cette fonction.

Question 4. La structure dite TA est fondée sur les arbres précédents. Ainsi, un TA noté T_N est donné par une affectation de valeurs de chaque nœud de l'arbre A_N. On utilisera pour cela un tableau d'entiers `t` indicé de 0 à N-1, tel que `t[i-1]` correspondra à la valeur du nœud i de A_N. Un TA doit vérifier les conditions suivantes :

- tout nœud de niveau pair a une valeur inférieure ou égale à celle de ses descendants
- tout nœud de niveau impair a une valeur supérieure ou égale à celle de ses descendants

Donnez un exemple de TA pour $N = 12$. Dans quel nœud se trouve la plus petite valeur d'un TA ? Justifiez votre réponse.

Question 5. Comment peut-on trouver la plus grande valeur d'un TA ? Justifiez votre réponse. Définissez la fonction C appelée `maximum`, qui prend en entrée une valeur N ainsi qu'un tableau `t` représentant un TA, et qui donne la plus grande valeur de ce TA. Donnez la complexité de cette fonction.

Question 6. Pour ajouter une nouvelle valeur x dans un TA, il suffit d'incrémenter N, puis d'affecter à `t[N-1]` la valeur x. Il faut ensuite réorganiser le tableau `t` pour qu'il satisfasse les conditions sur les TA. Cela s'opère par une éventuelle « remontée » de la valeur ajoutée, remontée qui s'opère à l'aide de permutations avec des valeurs figurant dans des nœuds ancêtres. Définissez la fonction C appelée `ajout`, qui prend en entrée des valeurs x et N ainsi qu'un tableau `t` représentant un TA, et qui ajoute la valeur x à ce TA. Donnez la complexité de cette fonction.

Question 7. Définissez la fonction C appelée `supprime`, qui prend en entrée un tableau `t` représentant un TA et qui supprime la plus petite valeur qui est mémorisée.

Question 8. Définissez une fonction de tri de tableau qui exploite les TA.