

Algorithmique TD4

Magistère Informatique 1ère Année

07 Octobre 2010

Exercice 1 *Récurtivité et Piles*

1. Écrire une fonction récursive pour évaluer une expression préfixe.

$$/ + 8 \times 3 4 - 5 3$$

2. Écrire une version itérative de cette fonction (utiliser les piles).
3. Écrire une version itérative du tri rapide (utiliser les piles) telle que la taille maximale de la pile est inférieure à $\log n$

Exercice 2 *Hauteur moyenne d'un ABR*

1. Calculer la hauteur moyenne d'un ABR construit aléatoirement.
Univers : Permutations de taille n avec distribution uniforme.
Hauteur : Pour une permutation σ , hauteur de l'arbre obtenu par insertions successives de $\sigma(1), \dots, \sigma(n)$.
2. Calculer le coût moyen d'une recherche fructueuse et infructueuse dans un ABR construit aléatoirement.

Exercice 3 *Arbres AVL (Extrait Partiel 2006)*

Un AVL est un arbre binaire de recherche (ABR) tel que pour chaque nœud de l'arbre, la différence de hauteur entre le sous-arbre gauche et le sous-arbre droit est d'au plus 1.

- (a) Soit n le nombre de nœuds et h la hauteur d'un AVL (la hauteur d'un arbre réduit à sa racine est 1). Montrer que $F_{h+2} - 1 \leq n \leq 2^h - 1$ où F_k est le k -ième nombre de Fibonacci avec $F_0 = 0$, $F_1 = 1$ et $F_{k+2} = F_{k+1} + F_k$ pour $k \geq 0$. Montrer que ces bornes sont atteintes.

Montrer que pour $k \geq 0$ on a $F_{k+2} \geq \phi^k$ où $\phi = (1 + \sqrt{5})/2$.

En déduire que $\log_2(n+1) \leq h \leq \log_\phi(n+1)$ et que la recherche dans un AVL se fait au pire en temps $O(\log(n))$.

Pour les algorithmes, un nœud x d'un AVL sera représenté par une structure comportant

- une clé, notée $x \cdot \text{cle}$ d'un type totalement ordonné (par exemple de type entier);
- un sous-arbre gauche, noté $x \cdot g$ et un sous-arbre droit, noté $x \cdot d$;
- la hauteur de l'arbre, notée $x \cdot h$.

L'arbre vide est noté NULL. On notera $h(x)$ la hauteur d'un arbre x avec $h(\text{NULL}) = 0$. On notera $n(x)$ le nombre de nœuds d'un arbre x .

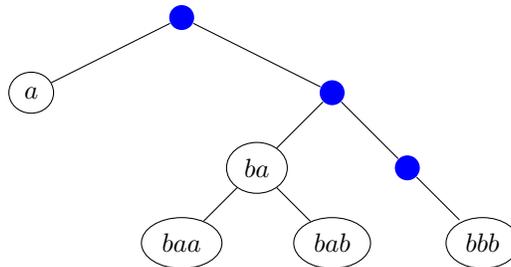
- (b) Le but de cette question est d'écrire une procédure `équilibrer(x)` pour transformer en AVL en temps constant un ABR de racine x en supposant que ses deux sous-arbres sont des AVL et que la différence de hauteur entre les deux sous-arbres est d'au plus 2.

Proposer un rééquilibrage dans le cas où $h(a \cdot g) - h(a \cdot d) = 2$. On pourra distinguer les cas $h(a \cdot g \cdot g) \geq h(a \cdot g \cdot d)$ et $h(a \cdot g \cdot g) < h(a \cdot g \cdot d)$. Illustrer les transformations sur des dessins. Écrire la procédure `équilibrer(x)`.

- (c) Écrire un algorithme pour insérer une clé c dans un AVL x en temps au pire $O(1 + h(x))$. Si x' est l'arbre obtenu, comparer $h(x')$ et $h(x)$. Justifier la correction de l'algorithme (on ne demande pas une preuve de programme). Montrer que cet algorithme engendre au plus un rééquilibrage.
- (d) Écrire un algorithme pour supprimer une clé c dans un AVL x en temps au pire $O(1 + h(x))$. Justifier la correction de l'algorithme (on ne demande pas une preuve de programme). Combien de rééquilibrages peuvent être nécessaires ?
- (e) Écrire un algorithme qui réalise la fusion de deux AVL x et y et d'une clé c en supposant que toutes les clés de x sont strictement inférieures à c et que toutes les clés de y sont strictement supérieures à c . Cet algorithme devra fonctionner en temps $O(1 + |h(x) - h(y)|)$. Justifier.
- (f) Écrire un algorithme qui réalise la scission d'un AVL x en deux AVL y et z contenant respectivement les clés de x inférieures ou égales à c pour y et strictement supérieures à c pour z . Cet algorithme devra fonctionner en temps $O(1 + |h(x)|)$. Justifier.

Exercice 4 Etant donné un alphabet Σ ordonné, on définit l'ordre lexicographique sur Σ^* comme suit : Soit $w = a_0a_1\dots a_p$ et $w' = b_0b_1\dots b_q$, $w < w'$ si et seulement si w est un préfixe de w' ou $\exists j$ tel que $\forall i < j$ $a_i = b_i$ et $a_j < b_j$.

La structure d'arbre de base peut contenir des mots écrits sur un alphabet à deux lettres. Par exemple l'arbre ci-dessous contient les mots bab , ba , baa , bbb et a . Soit \mathcal{L} un langage fini sur un alphabet à deux lettres dont la somme des longueurs des mots vaut n . Montrer qu'on peut utiliser la structure d'arbre de base pour trier le langage lexicographiquement en temps $O(n)$.



Exercice 5 Arbres Binaires de recherches Balisés

Dans un ABR balisé, les clés ne se trouvent qu'aux feuilles et les noeuds internes ont tous exactement 2 fils et contiennent des balises qui permettent d'orienter la recherche. Si x est un noeud interne, alors,

$$\{c(x.g)\} < x.\text{balisé} \leq c(x.d)$$

1. Montrer que $n(a) = 2f(a) - 1$ si a est un ABR balisé.
2. Ecrire les procédures d'insertion et de suppression dans un ABR balisé.
3. Ecrire une procédure pour transformer en temps linéaire un ABR balisé en un ABR classique ayant la même structure.
4. Ecrire la procédure inverse.