Université de Marne-la-Vallée

Licence d'informatique et d'I.U.P.
Examen de Structures de données

Janvier 2004, durée : 2h

Seule la consultation des supports de cours, travaux dirigés
et travaux pratiques est autorisée pendant le contrôle.

1 Arbres de Fibonacci

On note $\mathcal{F} = (\mathcal{F}_k \mid k \in \mathbb{N})$ la suite des arbres de Fibonacci dont les nœuds sont des entiers positifs consécutifs. La suite est définie par : $\mathcal{F}_0 = ()$, $\mathcal{F}_1 = (1, (), ())$, et $\mathcal{F}_k = (F_{k+1}, \mathcal{F}_{k-1}, \mathcal{F}'_{k-2})$ pour $k > 1$, où F_k désigne le nombre de Fibonacci d'indice k ($F_0 = 0$, $F_1 = 1$, et $F_k = F_{k-1} + F_{k-2}$ pour $k > 1$) et où $\mathcal{F}'_{k-2}$ est l'arbre obtenu à partir de l'arbre $\mathcal{F}_{k-2}$ en ajoutant F_{k+1} à chacun de ses nœuds.

1. **Hauteur**. Montrer que $\mathcal{F}_k$ a pour hauteur $k-1$ et qu'il est un arbre AVL (équilibré en hauteur).

2. **Taille**. Montrer que $\mathcal{F}_{k-1}$ possède $F_{k+1} - 1$ nœuds et en déduire que $\mathcal{F}_k$ est un arbre binaire de recherche. Montrer tout arbre AVL de hauteur $k-1$ possède au moins $F_{k+2} - 1$ nœuds.

3. **Ajout**. Dessiner l'arbre AVL obtenu par ajout de la valeur 0 à l'arbre $\mathcal{F}_5$. Combien et quels types de rotation sont nécessaires pour équilibrer l'arbre ?

4. **Retrait**. Dessiner l'arbre AVL obtenu par suppression du nœud maximal de $\mathcal{F}_5$ (nœud qui est le plus à droite dans l'arbre). Combien et quels types de rotation sont nécessaires pour équilibrer l'arbre ?

Classement des suffixes

Le but de ce problème est d'écrire un algorithme de classement des suffixes d'un mot.

Le mot y de longueur n considéré dans la suite est un tableau de caractères aussi noté $y[0]y[1]\cdots y[n-1]$ ou simplement $y[0\,..\,n-1]$. Les suffixes (non vides) de y sont les n

mots $y[0\,..\,n-1]$, $y[1\,..\,n-1]$, $y[2\,..\,n-1]$, …, $y[n-1]$. Pour $k=0$, $k=1$ ou $k=2$, on note S_k l'ensemble des suffixes de y commençant aux positions congrues à k modulo 3 :

$$S_k = \{y[i\,..\,n-1] \mid i \text{ modulo } 3 = k\}.$$

5. **Classement lexicographique**. Rappeler un algorithme de tri lexicographique de mots et sa complexité. Quel est son temps d'exécution lorsqu'il est appliqué aux suffixes du mot y de longueur n ?

6. **Triplets**. On note

   $$T_1 = \{y[i\,..\,i+2] \mid 0 \leq i < n \text{ et } i \text{ modulo } 3 = 1\} = \{y[1\,..\,3], y[4\,..\,6], y[7\,..\,9], \ldots\},$$

   l'ensemble des segments de longueur 3 de y commençant aux positions congrues à 1 modulo 3, et

   $$T_2 = \{y[i\,..\,i+2] \mid 0 \leq i < n \text{ et } i \text{ modulo } 3 = 2\} = \{y[2\,..\,4], y[5\,..\,7], y[8\,..\,10], \ldots\},$$

   l'ensemble de ceux commençant aux positions congrues à 2 modulo 3. (On pourra supposer que $y[n]$ et $y[n+1]$ sont définis et égaux à un caractère strictement plus petit dans l'ordre alphabétique que tous les autres caractères de y.)
   Indiquer en quel temps on peut classer (en ordre alphabétique) $T_1 \cup T_2$ par l'algorithme de tri lexicographique de la question 5.

7. **Récursion**. Pour chaque segment $y[i\,..\,i+2]$ de $T_1 \cup T_2$, on note t_i son rang dans la liste classée des segments de $T_1 \cup T_2$. On considère le mot

   $$z = t_1 t_4 t_7 \cdots t_2 t_5 t_8 \cdots.$$

   Quelle est la longueur de z ? Montrer que classer les suffixes de z est équivalent à classer les mots de $S_1 \cup S_2$.

8. **Récurrence**. Soit T une fonction entière qui satisfait les égalités : $T(0) = 1$ et $T(n) = T(2n/3) + n$ pour $n > 0$. Expliciter $T(n)$ en fonction de n.

9. **Autres suffixes**. On note s_i le rang du suffixe de $S_1 \cup S_2$ qui commence à la position i sur y. (s_i résulte du calcul de la question 7.) Montrer que l'on peut comparer deux mots de S_0 au moyen d'une comparaison de caractères et d'une comparaison de rangs de suffixes de $S_1 \cup S_2$. En déduire que le classement de S_0 peut être réalisé en temps linéaire, $O(n)$.

10. **Fusion**. Montrer que l'on peut comparer un mot de S_0 et un mot de $S_1 \cup S_2$ au moyen d'une ou deux comparaisons de caractères et d'une comparaison de rangs de suffixes de $S_1 \cup S_2$. En déduire que la fusion de la liste classée des mots de S_0 avec la liste classée des mots de $S_1 \cup S_2$ peut se faire en temps linéaire, $O(n)$.