Université de Marne-la-Vallée

Licence d'informatique et d'I.U.P.
Examen d'algorithmique

Janvier 2000, durée : 2h

Seule la consultation des supports de cours, travaux dirigés et travaux pratiques est autorisée pendant le contrôle.
Chacune des dix questions compte pour deux points

Tri de suffixes

On considère une table de caractères, y, de dimension n. Le but de cet exercice est de calculer la table Pos de dimension n pour laquelle

$$y[Pos[0]\,..\,n-1] < y[Pos[1]\,..\,n-1] < \ldots < y[Pos[n-1]\,..\,n-1],$$

où $y[i\,..\,n-1] = y[i]y[i+1]\ldots y[n-1]$ désigne le suffixe de y qui est à la position i. La table fournit ainsi les suffixes non vides de y en ordre croissant (pour l'ordre lexicographique).

1. Écrire en C le code de la fonction `strcmp` de prototype
   `int strcmp(const char * s, const char * t);`
2. Écrire en C une fonction d'en-tête `int Partage(int d, int f, int p)` qui, appliquée aux suffixes dont les positions sont $Pos[d]$, $Pos[d+1]$, $\ldots$, $Pos[f]$ et au pivot $y[Pos[p]\,..\,n-1] = z$ $(d \leq p \leq f)$, effectue le « partage » des suffixes considérés selon le pivot (en modifiant Pos) et produit le plus petit indice k $(d \leq k \leq f)$ pour lequel $y[Pos[k]\,..\,n-1] = z$.
3. Écrire un algorithme de classement des suffixes de y qui produit la table Pos en utilisant la fonction de partage du 2. Quelle table Pos obtient-on avec $y =$ `"abracadabra"`?
4. Donner une majoration asymptotique des temps d'exécution de `Partage(d,f,p)` et de l'algorithme de classement. Quel peut être son temps moyen d'exécution en supposant que le temps moyen pour comparer deux suffixes est $O(\log n)$?

Arbre de recherche optimal

L'objectif de cet exercice est d'étudier la construction d'un arbre binaire de recherche qui procure un temps d'accès optimal aux éléments stockés dans ses feuilles lorsque la probabilité d'avoir à rechercher chacun de ces éléments est connue.

Soient $a_0,a_1,\ldots,a_n$ des éléments satisfaisant $a_0 < a_1 < \ldots < a_n$ et de poids respectifs $q_0,q_1,\ldots,q_n$ ($q_0 + q_1 + \ldots + q_n = 1$). Le coût d'un arbre binaire (localement) complet dont les feuilles sont étiquetées par les paires (a_i,q_i) est

$$\sum_{i=0}^{i=n} q_i \times \mathit{niveau}(f_i),$$

où f_i est la feuille d'étiquette (a_i,q_i) et $\mathit{niveau}(f_i)$ son niveau dans l'arbre.

5. Construire un arbre de Huffman $\mathcal{H}$ pour la suite

$$S = ((\mathtt{A},23),(\mathtt{C},8),(\mathtt{G},18),(\mathtt{T},1),(\mathtt{X},16),(\mathtt{Y},1))$$

et donner son coût. Est-il possible de transformer l'arbre $\mathcal{H}$ en un arbre binaire de recherche sur les lettres en échangeant les sous-arbres gauches et droits de certains nœuds ?

6. **Arbre intermédiaire**. On construit un arbre $\mathcal{I}$ en appliquant l'algorithme de Huffman à la suite $((a_i,q_i) \mid 0 \leq i \leq n)$ avec les modifications suivantes : à chaque étape les deux éléments, x_i et x_j avec $i < j$, choisis au sein de la suite par l'algorithme ne doivent pas être séparés par une feuille et la somme de leur poids est minimale. Le parent qui leur est attribué remplace x_i dans la suite et a pour poids la somme des poids ; l'élément x_j est supprimé de la suite.
Construire l'arbre $\mathcal{I}$ correspondant à la suite S de la question 5.

7. **Niveaux**. On suppose que le type de l'arbre $\mathcal{I}$ est donné par la déclaration :
`typedef struct NOEUD {char c; int q; struct * NOEUD g,d;} * arbre;`
Écrire le code C d'une fonction qui produit, à partir de $\mathcal{I}$, la suite $(l_i \mid 0 \leq i \leq n)$ où l_i est le niveau de la feuille d'étiquette (a_i,q_i) dans $\mathcal{I}$.

8. **Arbre de recherche optimal**. On construit un arbre binaire de recherche $\mathcal{A}$ de coût minimal à la façon de l'algorithme de Huffman à partir de la suite $((a_i,q_i,l_i) \mid 0 \leq i \leq n)$ où les niveaux l_i sont les valeurs définies à la question précédente. À chaque étape on choisit les deux premiers éléments consécutifs de la suite qui ont niveau maximal l. On leur crée un parent commun auquel on attribue le niveau $l-1$ et qui prend leur place au sein de la suite.
Construire l'arbre de recherche $\mathcal{A}$ associé à la suite

$$((\mathtt{A},23,2),(\mathtt{C},8,2),(\mathtt{G},18,2),(\mathtt{T},1,4),(\mathtt{X},16,4),(\mathtt{Y},1,3)).$$

9. **Temps d'exécution**. Indiquer quelle structure de données peut être utilisée pour stocker la suite des nœuds pondérés considérés à la question 8 et quel temps d'exécution la structure induit pour la construction de l'arbre $\mathcal{A}$.

10. **Code**. Quels codes binaires obtient-on pour les lettres `A`,`C`,`G`,`T`,`X`,`Y` au moyen de l'arbre $\mathcal{H}$ calculé à la question 5, d'une part, et au moyen de l'arbre $\mathcal{A}$ calculé à la question 8, d'autre part ?