Algorithmique

Magistère STIC

Examen du 15 décembre 2005

durée 3 heures

Les notes de cours et de TD sont autorisées.

Les exercices sont indépendants.

Toutes les réponses devront être correctement justifiées. La rigueur des raisonnements, la clarté des explications, et la qualité de la présentation influeront sensiblement sur la note.

1 Insertion à la racine d'un ABR

On s'intéresse dans cette partie à des arbres binaires de recherche (ABR). Dans certains cas, les éléments que l'on a le plus de chances de rechercher sont ceux qui ont été insérés le plus récemment. Dans ce cas l'algorithme classique d'insertion aux feuilles n'est pas très adapté.

Soit t un ABR et x une clé. Lorsqu'on insère x dans t à la racine, la profondeur d'un nœud augmente d'au plus 1 et bien sûr la clé x se retrouve à la racine du nouvel ABR t'. Donc l'avant dernier élément inséré est un fils de la racine, et l'avant avant dernier élément inséré est un fils ou un petit-fils de la racine.

a) Dessiner les arbres obtenus après chaque insertion à partir de l'arbre vide par insertions à la racine des clés 7, 4, 9, 5, 6, 1, 3, 2, 8.

b) Écrire un algorithme d'insertion à la racine dont la complexité en temps est proportionnelle à la hauteur de l'arbre.
Donner le ou les invariants (de boucle ou de récursivité) et prouver la correction de l'algorithme. Justifier aussi sa complexité.

c) Ordonner les entiers de 1 à 7 de telle façon que la création d'un ABR par insertions successives à la racine de ces entiers dans cet ordre donne l'ABR parfait (toutes les branches ont la même longueur).

d) Donner un algorithme qui, étant donné un entier k, affiche les entiers de 1 à $2^k - 1$ dans un ordre permettant d'obtenir un arbre parfait par insertion à la racine de ces entiers à partir d'un arbre vide.
Prouver la correction de l'algorithme et calculer sa complexité.

e) Comparer l'arbre obtenu à partir d'un arbre vide par insertion à la racine d'une suite d'entiers 2 à 2 distincts et l'arbre obtenu à partir de l'arbre vide par insertion aux feuilles de cette suite d'entiers dans l'ordre inverse.

2 Recherche de motif par duels

On note $t = t[1]\cdots t[n]$ le texte de longueur n et $x = x[1]\cdots x[m]$ le motif de longueur m. On notera $t[i\cdot\cdot j] = t[i]\cdots t[j]$ le facteur de t commençant à la position i et se terminant à la position j. Par convention, ce facteur est vide si $j < 1$ ou $i > n$ ou $j < i$.

On rappelle qu'un *bord* d'un mot w est un préfixe strict de w qui est aussi suffixe de w. On note $\mathrm{Bord}(w)$ le plus long bord de w.

Le motif x étant fixé, on note $b(j) = |\mathrm{Bord}(x[1\cdot\cdot j])|$ la longueur du plus long bord de $x[1\cdot\cdot j]$. On rappelle que la fonction $b : \{1,\ldots,m\} \to \{0,\ldots,m-1\}$ peut se calculer en temps linaire. Plus précisément, on peut calculer en temps $\mathcal{O}(m)$ un tableau B défini par $B[i] = b(i)$ pour $1 \le i \le m$.

Le motif x étant toujours fixé, on définit la fonction $p : \{1,\ldots,m\} \to \{0,\ldots,m-1\}$ par

$$p(i) = \max\{k \mid 0 \le k \le m-i \text{ et } x[i+1\cdot\cdot i+k] \text{ est préfixe de } x\}.$$

a) Calculer les fonctions b et p pour le motif $x = abcabaabcababc$.

Les questions **(b)** à **(i)** visent à élaborer un algorithme qui calcule le tableau représentant la fonction p en temps linéaire.

b) Montrer que pour $1 \le j \le m$ on a $b(j) \le p(j - b(j))$.

c) Montrer que pour $1 \le i \le m$ on a $p(i) \le b(i + p(i))$.

Pour $1 \le i \le m$ on définit $E(i) = \{j \mid i < j \le m \text{ et } j - b(j) = i\}$ et

$$f(i) = \begin{cases} 0 & \text{si } E(i) = \emptyset \\ \max\{b(j) \mid j \in E(i)\} & \text{sinon.} \end{cases}$$

d) Calculer la fonction f pour le motif $x = abcabaabcababc$. Pour $1 \le i \le m$, montrer que

- $f(i) \le p(i)$ et
- $E(i) \ne \emptyset$ si et seulement si $f(i) > 0$.

e) Montrer que si $j \in E(i)$ et $j \le j' \le i + p(i)$ alors $j' \in E(i)$. En déduire que $E(i) \ne \emptyset$ implique $f(i) = p(i)$.

f) Montrer que si $p(i') > 0$ alors il existe i tel que $E(i) \ne \emptyset$ et $i \le i' < i' + p(i') \le i + p(i)$.

g) Montrer que si $i < i'$, $E(i) \ne \emptyset$ et $E(i') \ne \emptyset$ alors $i + p(i) < i' + p(i')$.

h) Montrer que si $i < i' < i + p(i)$ et $E(i) \ne \emptyset$ et $E(i'') = \emptyset$ pour tout $i < i'' \le i'$ alors $p(i') = \min(p(i' - i),\, p(i) - (i' - i))$.

i) Déduire des questions précédentes un algorithme qui calcule en temps $\mathcal{O}(m)$ le tableau P définissant la fonction p. On justifiera la correction de l'algorithme à l'aide des questions précédentes et on prouvera qu'il s'exécute bien en temps linéaire.

Nous passons maintenant à l'algorithme de recherche de motif par duels. On définit

$$q(i) = \begin{cases} 1 + p(i) & \text{si } i + 1 + p(i) \leq m \\ 0 & \text{sinon.} \end{cases}$$

j) Calculer les fonctions p et q pour le motif $x = bababba$.

On veut trouver les positions $i \in \{0, \ldots, n-m\}$ du texte telles que $x = t[i+1 \cdot\cdot i+m]$. Noter que dans un tel alignement le motif commence à la position $i+1$, ce qui justifie l'intervalle considéré pour i.

Soient $i,i' \in \{0, \ldots, n-m\}$ deux positions du texte. On dit qu'il y a *duel* entre i et i' si $0 < |i' - i| < m$ et $q(|i' - i|) \neq 0$. Sinon les positions i et i' sont dites *compatibles*. Lorsqu'il y a duel entre $i < i'$ alors i' gagne le duel si $x[q(i' - i)] = t[i' + q(i' - i)]$, sinon c'est i qui gagne le duel.

k) Soient $i < i'$ deux positions en duel. Montrer que si i gagne le duel alors $x \neq t[i' + 1 \cdot\cdot i' + m]$ et que si i' gagne le duel alors $x \neq t[i + 1 \cdot\cdot i + m]$.

l) Soient $i < i' < i''$ des positions du texte telles que i et i' d'une part et i' et i'' d'autre part sont compatibles. Montrer que i et i'' sont compatibles.

On considère l'algorithme suivant :

```
s <- pile vide
pour i <- n-m à 0 faire
  i1 <- i
  tant que s.nonVide() faire
    i2 <- s.dépiler()
    si i1 et i2 sont compatibles alors
      s.empiler(i2)
      sortir de la boucle tant que
    finsi
    i1 <- le gagnant du duel entre i1 et i2
  fin tant que
  s.empiler(i1)
fin pour
marquer toutes les positions contenues dans la pile s
```

m) Exécuter cet algorithme sur le texte $t = abcbababbababbabac$ et le motif $x = bababba$.

n) Montrer que les positions marquées sont deux à deux compatibles et que si $x = t[i + 1 \cdot\cdot i + m]$ alors la position i est marquée.

Une position $1 \leq j \leq n$ du texte est *bonne* s'il existe une position marquée i telle que $i < j \leq i + m$ et $t[j] = x[j - i]$.

o) Donner un algorithme qui construit en temps $\mathcal{O}(n)$ un tableau de booléens $g[1 \cdot\cdot n]$ tel que $g[j] = V$ si et seulement si la position j est bonne.

p) Montrer que $x = t[i + 1 \cdot\cdot i + m]$ si et seulement si i est une position marquée et $g[i + 1 \cdot\cdot i + m]$ ne contient que des V. En déduire un algorithme de recherche de motif qui fonctionne en temps linéaire.