

Exercice de style

Michel Habib

7 mars 2011

1 Un problème d'arborescence

On considère une arborescence A finie. Les feuilles de A sont étiquetées par des éléments d'un ensemble X . A est représentée par la fonction père (tout élément sauf la racine admet un père unique), et l'on suppose en outre que chaque noeud de l'arborescence possède une variable qui contient le nombre de fils. On suppose qu'un pointeur permet d'associer à chaque élément de X la feuille qui lui correspond dans A .

1. Etant donné $S \subseteq X$ écrire un algorithme qui vérifie qu'il existe un noeud $a_s \in A$ tel que S corresponde à l'ensemble des feuilles de la sous-arborescence de A engendrée par a_s . Peut-on écrire un algorithme en $O(|S|)$?

2 Une solution à base de parcours

Il existe sûrement plusieurs solutions ayant la même complexité $O(|S|)$. Nous présentons ici une solution inspirée d'un parcours de graphe.

Nous ferons l'hypothèse que chaque sommet de l'arborescence x possède un identifiant unique ($numero(x) \in [1, |A|]$). Ainsi nous pouvons créer des tableaux de taille $|A|$ afin d'y mettre des informations sur les sommets de l'arborescence.

Données : Une arborescence A et F un ensemble de feuilles de A

Résultat : Un sommet de A

$OUVERTS \leftarrow \{F\}$

$FERMES \leftarrow \emptyset$

$Listaux \leftarrow \emptyset$

$dernier \leftarrow NIL$

tant que $OUVERTS \neq \emptyset$ **faire**

$z \leftarrow Choix(OUVERTS)$
 $Ajout(z, FERMES)$
 $Explorer(z)$
 $Retrait(z, OUVERTS)$
 $dernier \leftarrow z$

Explorer(z)

$p \leftarrow parent(z)$

si $p \neq NIL$ **alors**

si $Tag(numero(p)) = -1$ **alors**
 $Tag(numero(p)) \leftarrow degre(numero(p))$
 $Ajout(numero(p), Listaux);$
 $Tag(numero(p)) \leftarrow Tag(numero(p)) - 1$
 si $Tag(numero(p)) = 0$ **alors**
 $Ajout(p, OUVERTS)$

sinon

 la réponse est l'arborescence A en entier STOP

Si l'on veut remettre A dans la situation initiale, il suffit de remettre le tableau Tag à sa situation initiale par les instructions suivantes en $O(|Listaux|)$.

tant que $Listaux \neq \emptyset$ **faire**

$x \leftarrow Premier(Listaux)$
 $Retrait(x, Listaux); Tag(x) \leftarrow -1$

En outre si l'on veut savoir si la réponse est OUI ou NON, il suffit de modifier la procédure précédente comme suit :

```

racine ← 0
tant que Listaux ≠ ∅ faire
  | x ← Premier(Listaux)
  | Retrait(x, Listaux)
  | si Tag(x) ≠ 0 alors
  |   | racine ← racine + 1
  |   | Tag(x) ← -1
  | si racine > 1 alors
  |   | Alors réponse NON
  | sinon
  |   | réponse OUI, dernier

```

En cas de succès, à la fin de la procédure la variable **dernier** pointe sur le nœud de l'arborescence qui est la racine du sous-arbre dont l'ensemble des feuilles est F . En effet ce parcours construit une extension linéaire de l'ensemble des sommets visités de l'arborescence (cf. cours) et donc s'il existe un maximum unique c'est le dernier élément.

Théorème 1 *Si tout sommet de A est soit une feuille soit un nœud ayant au moins deux fils alors $|Listaux| \leq 2|F|$.*

Corollaire 1 *La complexité temporelle de l'algorithme précédent est donc $O(|F|)$.*

3 La question des certificats

En cas de réponse NON, on peut fournir la liste des sommets x tels que $Tag(x) > 0$.

En cas de réponse OUI, si l'on veut vérifier la réponse, soit l'arborescence est munie de pointeurs vers le bas et dans ce cas il suffit de démarrer un parcours d'arborescence classique (en profondeur par exemple) à partir du sommet fourni **dernier** et vérifier que l'on atteint bien toutes les feuilles.

Sinon, il faut construire une arborescence inversée auxiliaire lors du parcours.

4 Commentaires

Ce problème est issu d'un algorithme incrémental de reconnaissance des cographes [CPS85]. À chaque étape de l'algorithme on considère $S = N(x)$, les sommets étant visités un seule fois mais dans un ordre quelconque, l'arborescence A étant modifiée en fonction du résultat de la recherche.

Références

- [CPS85] Derek G. Corneil, Yehoshua Perl, and Lorna K. Stewart. A linear recognition algorithm for cographs. *SIAM J. Comput.*, 14(4) :926–934, 1985.