Devoir d'Algorithmique

Magistère Informatique 1ère Année

Octobre 2009

Première partie
Arbres Évasés

On travaille sur des arbres binaires de recherche, étant donné un arbre T on note $c(T)$ l'ensemble de ses clés. On va apppliquer sur ces arbres une opération $EVASER$. Étant donné une clé i et un arbre T, $EVASER(i, T)$ réarrange T en un arbre T' contenant le même ensemble de clés que T de la façon suivante :

- Si T contient i, i est à la racine de T'.
- Sinon la racine de T' est soit la plus petite clé de T supérieure à i, soit la plus grande clé de T inférieure à i

On suppose en particulier qu'il est possible d'appeler $EVASER$ sur deux clés fictives $+\infty$ et $-\infty$ qui n'apparaissent pas dans les arbres et qui sont respectivement strictement supérieure et strictement inférieure à toutes les autres clés.

Question 1 On considère les deux opérations classiques suivantes :

$INSERER(i, T)$	Si l'arbre T ne contient pas la clé i, insère i dans T.
$SUPPRIMER(i, T)$	Si l'arbre T contient pas clé i, supprime i de T.

Montrer qu'il est possible d'effectuer ces opérations au moyen d'un nombre constant d'appels à $EVASER$ et d'opérations en temps constant.

Question 2 On considère maintenant les deux opérations suivantes :

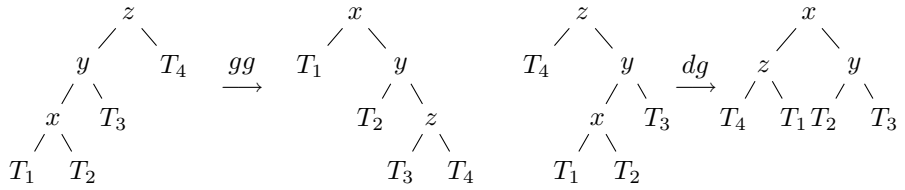
$CONCATENER(T_1, T_2)$	Si pour toute clé i_1 dans T_1 et toute clé i_2 dans T_2 $i_1 < i_2$, construit un arbre contenant l'union des clés de T_1 et T_2
$SCINDER(i, T)$	Construit deux arbres T_1 et T_2 tels que : $- c(T) = c(T_1) \cup c(T_2)$ $- c(T_1) \cap c(T_2) = \emptyset$ $- \forall i_1 \in c(T_1), \forall i_2 \in c(T_2), i_1 \leq i \leq i_2$

Montrer qu'il est possible d'effectuer ces opérations au moyen d'un nombre constant d'appels à $EVASER$ et d'opérations en temps constant.

On décrit maintenant comment $EVASER(x, T)$ est réalisée. On commence par localiser le sommet x contenant la clé i dans T. Ce sommet est ensuite remonté à la racine en utilisant des doubles rotations et éventuellement une rotation simple :

1. Si le père de x est la racine y, on effectue une rotation simple autour de (x, y).

FIG. 1 – Doubles Rotations



2. Sinon on note y le père de x et z celui de y (voir Fig I).
   (a) Si x et y sont tous deux fils gauche, on effectue la double rotation gauche-gauche correspondante.
   (b) Si x est fils gauche et y fils droit, on effectue la double rotation droite-gauche correspondante.
   (c) Les autres cas sont symétriques.

   x est amené à la racine via une séquence de telles rotations.

On cherche maintenant à calculer le coût amorti d'une suite d'opérations réalisées au moyen d'$EVASER$. On définit pour celà une fonction potentiel, on suppose que toute clé i dans l'ensemble des clés est munie d'un poids $p(i)$, étant donné un arbre T et un sommet x de T on appelle rang de x, l'entier $r(x) = \lfloor log(n(x)) \rfloor$ où $n(x)$ est la somme des poids des sommets du sous-arbre enraciné en x. On note $r(T)$ le rang de la racine de T. On prend comme potentiel la somme des rangs des sommets de l'arbre.

Question 3 On considère le cas 1. dans la procédure $EVASER(i, T)$, soit x le sommet concerné. On note $r(x)$ le rang de x avant la rotation et $r'(x)$ le rang de x après la rotation. Montrer que le coût de la rotation relativement au potentiel est majoré par $3(r'(x) - r(x)) + 1$.

Question 4 On considère le cas 2.(a) dans la procédure $EVASER(i, T)$, soit x le sommet concerné. On note $r(x)$ le rang de x avant la double rotation et $r'(x)$ le rang de x après la double rotation. Montrer que le coût de la double rotation relativement au potentiel est majoré par $3(r'(x) - r(x))$.

Question 5 On considère le cas 2.(b) dans la procédure $EVASER(i, T)$, soit x le sommet concerné. On note $r(x)$ le rang de x avant la double rotation et $r'(x)$ le rang de x après la double rotation. Montrer que le coût de la double rotation relativement au potentiel est majoré par $3(r'(x) - r(x))$.

Question 6 On admet que le coût des autres double rotations relativement au potentiel est majoré par $3(r'(x) - r(x))$ (le résultat est prouvé de manière similaire aux cas précédents). Montrer que le coût total de l'opération $EVASER(i, T)$ relativement au potentiel est majoré par $3(r(T) - r(x)) + 1$ où x est le sommet remonté à la racine par $EVASER$.

Question 7 On considère une suite de n insertions et suppressions réalisées au moyen d'$EVASER$ sur un arbre contenant initialement m clés. On suppose $n > m$, montrer que le coût amorti d'une opération est en $O(log(n))$.

Question 8 On considère la même suite d'opérations. A chaque clé i on associe $q(i)$ le nombre de fois où l'évasion porte sur i. Montrer que le coût total est en :

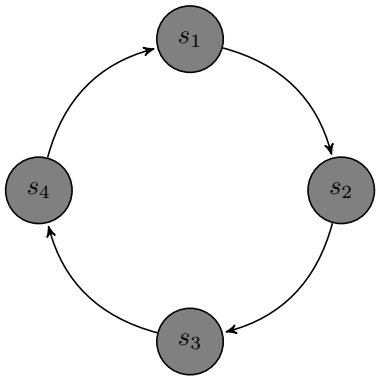
$$O(n + \sum_{i \in \text{clés}} q(i) log(\frac{n}{q(i)}))$$

Deuxième partie

Recherche d'un Leader

On considère un ensemble de n sites qui cherchent à désigner un leader parmis eux. On nomme les sites $s_1, s_2, ..., s_n$ et on suppose que chaque site s_i peut envoyer des message à s_{i+1} (sauf s_n qui peut en envoyer à s_1), il s'agit du seul moyen de communication dont disposent les sites. On suppose aussi qu'il existe un classement des sites selon leurs qualités de leader, chaque site dispose d'une fonction qui lui permet de départager deux sites selon ce classement et donc de savoir qui sera meilleur leader. On représente dans la Figure II les communications pour un ensemble de 4 sites.

FIG. 2 – Représentation des Communications pour 4 sites



Chaque site s_i dispose de deux variables :
- $état_i$: Représente l'état de i, elle peut prendre trois valeurs : *repos*, *en_cours* et *terminé*. Elle est initialisée à *repos*.
- $leader_i$: Contient l'identité du leader provisoirement soutenu par i.

Chaque site s_i qui est dans l'état *repos* peut à tout moment décider de lancer un processus de désignation en se proposant comme leader. Il effectue alors les actions suivantes :

1. Il passe $état_i$ à *en_cours*.
2. Il affecte s_i à $leader_i$.
3. Il envoie à son successeur une requête en se proposant comme leader.

Il est important de remarquer que tous les sites peuvent potentiellement lancer un processus de désignation au même moment.

Quand un site s_i reçoit un message lui proposant un autre site s_k comme leader on distingue plusieurs cas :
- s_i était dans l'état *repos*. Dans ce cas :
  1. s_i passe $état_i$ à *en_cours*.
  2. s_i passe $leader_i$ à s_k.
  3. s_i envoie une requête à son successeur en proposant s_k comme leader.
- s_k est un meilleur leader que $leader_i$. Dans ce cas :

1. s_i passe $leader_i$ à s_k.
2. s_i envoie une requête à son successeur en proposant s_k comme leader.

– $s_i = s_k = leader_i$. Dans ce cas s_i passe dans l'état *terminé* et envoie un message de confirmation à son successeur indiquant qu'il est leader.
– s_k est un moins bon leader que $leader_i$. Dans ce cas s_i ne fait rien.

Quand un site reçoit une message de confirmation il passe dans l'état *terminé* et transmet la confirmation à son successeur.

Question 1 Prouver que la procédure est correcte, c'est à dire qu'à la fin tous les sites sont dans l'état *terminé*, que $leader_i$ est identique pour tous les sites s_i et que le leader choisi est bien le meilleur parmis ceux qui ont lancé la procédure de désignation.

On s'intéresse maintenant au nombre de messages envoyés lors du processus de désignation.

Question 2 Montrer que le nombre de messages envoyés dans le pire des cas est en $\Theta(n^2)$.

Question 3 On considère que tous les sites lancent le processus de désignation. Calculer le nombre moyen de messages envoyés jusqu'à la désignation du leader sur l'ensemble des exécutions possibles.