Algorithmique
Travaux dirigés, 26 novembre 2004
Louis Granboulan

1 Manipulation d'un ensemble ordonné

1.1 Arbres AVL

Dans un univers ordonné, on représente un ensemble par un arbre binaire dont les nœuds sont étiquetés de telle sorte qu'un parcours infixe donne une suite croissante. On dit que l'arbre est équilibré (on parle d'arbre AVL, d'après Adel'son, Vel'skii et Landis) si pour tout nœud la différence entre son sous-arbre gauche et son sous-arbre droit n'excède pas 1.

1. Quelle est la complexité des opérations APPARTENANCE, INSERTION, SUPPRESSION et MINIMUM avec une représentation des ensembles par des arbres de recherche non nécessairement équilibrés.
2. Encadrer le nombre de nœuds d'un arbre AVL de hauteur h.
3. Montrer que pour une représentation d'un ensemble de taille n par un arbre AVL, les quatre opérations ci-dessus peuvent être effectuées en $\mathcal{O}(\log n)$ instructions.
4. Que dire des autres opérations ?
5. Comment construire un arbre binaire de recherche pour un ensemble donné, tel que le coût moyen de l'opération APPARTENANCE soit minimal, connaissant une distribution de probabilité sur l'univers ?

1.2 Arbres 2-3

On représente un ensemble par un arbre dont les nœuds internes ont deux ou trois fils et dont toutes les feuilles sont à la même hauteur. Les feuilles contiennent les valeurs des éléments. Si l'univers est ordonné, on propose deux façons d'étiqueter les nœuds internes : 1. Les feuilles sont en ordre quelconque et chaque nœud est étiqueté par la valeur minimum de son sous-arbre. 2. Les feuilles sont en ordre croissant et chaque nœud est étiqueté par la valeur minimum de chacun de ses sous-arbres.

1. Encadrer le nombre de nœuds d'un arbre 2-3 de hauteur h.
2. Quelles sont les opérations qu'on peut effectuer en $\mathcal{O}(\log n)$ instructions ?

1.3 B-arbres

Il s'agit d'une généralisation, où les nœuds internes ont entre $\lceil B/2 \rceil$ et B fils (à l'exception de la racine qui a entre 2 et B fils).

1. Généraliser les résultats sur les arbres 2-3 aux B-arbres.
2. Quelles sont les avantages de cette structure de données ? Comment choisir B ?

2 Manipulation d'une partition de l'univers

2.1 Forêts

On s'intéresse à un univers de taille u et aux opérations d'UNION et LOCALISATION. On représente chaque ensemble par un arbre dont les nœuds sont étiquetés par les éléments de l'ensemble. On aimerait réaliser ces opérations en temps presque constant.

1. L'opération d'UNION se réalise naturellement en temps constant, en ajoutant l'un des deux arbres comme fils supplémentaire de la racine de l'autre. Montrer qu'en l'absence de rééquilibrage de l'arbre, une opération de LOCALISATION peut prendre un temps $\mathcal{O}(u)$.
2. Montrer que si pour une UNION, on choisit de rajouter le plus petit des deux ensembles comme fils supplémentaire de la racine du plus grand (en nombre de nœuds), une opération de LOCALISATION prend un temps $\mathcal{O}(\log u)$.

2.2 Compression d'une forêt

On suppose qu'à chaque LOCALISATION, on modifie l'ensemble trouvé de telle sorte que tous les ancêtres de l'élément cherché deviennent des fils directs de la racine. On définit $F(0) = 1$ et $F(i+1) = 2^{F(i)}$, puis $G(u) = \min\{k \text{ tq } F(k) \geqslant u\}$.
Pour $\alpha \in \mathbb{R}$, on s'intéresse à l'exécution d'une séquence de αu opérations d'UNION et LOCALISATION à partir d'une partition de l'univers en singletons.
On définit le *rang* d'un élément v comme sa hauteur dans la forêt résultant de la séquence des opérations d'UNION seules. On partitionne les éléments en groupes, selon l'image par G de leur rang.

1. Montrer que la fonction G est quasi constante.
2. Montrer qu'il y a au plus $u/F(g)$ éléments de groupe g.
3. On sait que pour chaque opération de LOCALISATION, le coût est le nombre d'ancêtres de l'élément cherché. On va répartir le coût comme suit : si v est la racine, ou bien si v et son père sont dans des groupes distincts, le coût attribué directement aux opérations de LOCALISATION est incrémenté de 1. Sinon, c'est le coût attribué à l'élément v qui est incrémenté de 1.
   Montrer que le coût attribué directement à chaque opération de LOCALISATION est au maximum $G(u)$ et que le coût attribué à l'ensemble des éléments d'un même groupe est au plus u.
4. En déduire que l'ensemble des opérations de LOCALISATION se fait en temps $\mathcal{O}((1+\alpha)uG(u))$.

2.3 Application

Soit un graphe non orienté connexe (V, E) dont les arêtes ont un poids réel. On cherche une arborescence recouvrante de poids minimal. Voici un algorithme qui va construire l'ensemble T des arêtes de l'arborescence recouvrante. Le principe de l'algorithme est de partir d'une forêt recouvrante de poids minimal et de la rendre connexe.
On manipule l'ensemble VS dont les éléments sont l'ensemble des sommets pour chaque arborescence de la forêt. C'est donc une partition de V. On initialise T à l'ensemble vide et VS à la partition en singletons.
À chaque étape, on choisit l'arête de poids minimal. On la suprime de E. si ses deux extrémités sont dans des éléments de VS distincts, on rajoute cette arête à T et on réunit les deux éléments de VS correspondants. On s'arrête quand VS est un singleton.

1. Montrer que cet algorithme est correct
2. Évaluer sa complexité.