Algorithmique et programmation
Travaux dirigés, 19 et 20 octobre 1998
Louis Granboulan

1 Structures de données

1. Files et piles

   Rappels : une file est munie de deux opérations, `enfiler` et `défiler` et d'un test `vide`. Fonctionnement FIFO (first in, first out). Une pile munie de deux opérations, `push` et `pop` et d'un test `vide`. Fonctionnement LIFO (last in, first out).

   (a) Trouvez une méthode pour simuler une file avec des piles, telle que la complexité amortie d'une opération sur la file soit une constante.

2. Arbres binaires

   Rappels : tout nœud d'un arbre binaire a éventuellement un fils gauche et un fils droit. Si on complète cet arbre en rajoutant des fils à tous les nœuds qui n'en ont pas deux, on obtient un arbre binaire complet et les nœuds rajoutés sont les feuilles de l'arbre. On note n la taille d'un arbre binaire, c'est-à-dire le nombre de ses nœuds. L'arbre binaire complet correspondant a donc $n+1$ feuilles.
   On se propose de calculer le nombre C_n d'arbres binaires de taille n, appelé *nombre de Catalan*.

   (a) Commencez par établir une correspondance entre les arbres binaires complets de taille n dont un nœud ou une feuille est distingué et les arbres binaires complets de taille $n+1$ dont une feuille est distinguée. En déduire que $(n+2)C_{n+1} = 2(2n+1)C_n$.

   (b) En déduire que $C_n = \frac{1}{n+1}\binom{2n}{n}$.

   (c) Donner un algorithme de génération aléatoire et équiprobable d'un arbre binaire de taille n.

2 Comparaisons et tri

1. Minimum

   (a) Montrer, à l'aide d'un algorithme, que le minimum d'un ensemble totalement ordonné de n éléments peut être déterminé en $n-1$ comparaisons.

   (b) Est-ce la meilleure borne possible ?

   (c) Évaluer le nombre moyen d'opérations effectuées par votre algorithme

2. Minimum et maximum

   (a) Montrer, à l'aide d'un algorithme, que le minimum et le maximum d'un ensemble totalement ordonné de n éléments peuvent être déterminés en $\lceil 3n/2 \rceil - 2$ comparaisons.

   (b) Est-ce la meilleure borne possible ?

3. Tri d'un petit nombre d'éléments

   Soit $S(n)$ le nombre minimal de comparaisons pour trier n éléments.

   (a) Montrer que $S(n) \geqslant \lceil \log_2 n! \rceil$.

   (b) Déterminer les valeurs exactes de $S(1)$, $S(2)$, $S(3)$, $S(4)$ et $S(5)$.

3 Structures récursives

1. On suppose que les piles ont une représentation chaînée. Écrire les primitives `vide`, `push` et `pop`, puis implanter sans récursivité la recherche et suppression d'un élément dans une pile.
   (a) dans le cas où la pile vide est représentée par le pointeur NULL.
   (b) dans le cas où une garde précède le premier élément.
2. On utilise une représentation chaînée pour les arbres binaires complets. on appelle *frontière* la liste (de droite à gauche) des valeurs des feuilles de l'arbre.
   (a) Concevoir un algorithme qui calcule la forntière de l'arbre.
   (b) Imaginer un algorithme qui annonce si deux arbres binaires complets ont la même frontière. On doit éviter de calculer toute la frontière si ce n'est pas nécessaire.

4 Algorithmique

1. Exponentiation

   Dans un monoïde multiplicatif G, on suppose que les opérations `mul` et `sqr` de multiplication et d'élévation au carré sont efficaces. On cherche à calculer la puissance n-ième.

   (a) Trouver un algorithme qui effectue ce calcul en $\mathcal{O}(\log n)$ opérations. Compter plus précisément le nombre de carrés et de multiplications.
   (b) Imaginer une variation de cet algorithme qui, avec un petit précalcul, diminue le nombre moyen de multiplications. (Hint : représenter en base 2^k au lieu de 2.)
2. Avec inversion

   Dans le cas où G est un groupe, on peut espérer faire mieux si le calcul de l'inverse est efficace.

   (a) Trouver un algorithme qui donne une représentation de longueur minimale de l'entier n comme somme signée de puissances de 2.
   (b) Utiliser ceci pour le calcul de la puissance n-ième.
3. Systèmes de numération redondants

   On vient de voir que la représentation classique en base B d'un entier n n'est pas la seule intéressante. Ici on représente $n = \sigma_{i=0..k-1} n_i B^i$ avec $-c_{\min} \leqslant n_i \leqslant c_{\max}$ avec $c_{\min}$ et $c_{\max} < B$.

   (a) Donner une condition nécessaire et suffisante sur les entiers naturel B, $c_{\min}$ et $c_{\max}$ pour que tout entier naturel admette une représentation. Donner une exemple de système de numération redondant, i.e. tel que certains entiers admettent plusieurs représentations.
   (b) Supposant que $c_{\min} = c_{\max} = a$, que fait l'algorithme suivant qui prend $x = (x_{k-1} ... x_0)$ et $y = (y_{k-1} ... y_0)$ pour fournir les $k+1$ valeurs z_0, ... z_k ?

      1. Pour $i = 0..k-1$ calculer $t_{i+1} = \begin{cases} 1 & \text{si } x_i + y_i \geqslant a \\ -1 & \text{si } x_i + y_i \leqslant -a \\ 0 & \text{si } x_i + y_i \in]-a, a[\end{cases}$

      2. Pour $i = 0..k$ calculer $z_i = w_i + t_i$ avec la convention $t_0 = w_k = 0$.
   (c) Quel est l'intérêt d'un tel algorithme ?