Algorithmique Avancée Examen Réparti 1

AVEC CORRECTION

Les seuls documents autorisés sont les polys de cours, ainsi que la copie personnelle. Le barème donné est indicatif.

Exercice 1 *: QCM [4 points]*

Dans ce QCM, pour chaque question vous devez donner 1 seule réponse et expliquer votre choix par une ligne de texte ou une figure. Le barème est le suivant : **réponse correcte et correctement argumentée : 1 point. Réponse incorrecte ou correcte mais mal argumentée : 0 point.** *Il n'y a pas de points négatifs.*

Question 1 L'arbre binomial B_k, pour $k \geq 1$ est tel que le nombre de nœuds qui sont les fils des fils (c'est-à-dire les petits fils) de la racine de B_k est
A. $\frac{k(k-1)}{2}$. **B.** k. **C.** k^2.

Question 2 Le nombre de comparaisons nécessaires pour faire la fusion d'une file binomiale de 1221 éléments avec une file binomiale de 829 éléments est
A. 15. **B.** 12. **C.** 10.

Question 3 Pour un arbre 2-3-4 contenant n clés, l'algorithme permettant de calculer la hauteur de l'arbre est au pire cas (en nombre de nœuds traversés)
A. $\Theta(n)$. **B.** $\Theta(\log n)$. **C.** $\Theta(1)$.

Question 4 On considère une table de hachage de taille $2n$ dans laquelle on insère $n/8$ clés, avec une fonction de hachage uniforme. Le nombre moyen de clés qui ont la même valeur de hachage i fixée est
A. 1/16. **B.** 1/4. **C.** 1/2.

Solution

1. A.

2. C.

$$1221 = 1024 + 128 + 64 + 4 + 1 = 2^{10} + 2^7 + 2^6 + 2^2 + 2^0.$$

$$829 = 512 + 256 + 32 + 16 + 8 + 4 + 1 = 2^9 + 2^8 + 2^5 + 2^4 + 2^3 + 2^2 + 2^0.$$

0-0 ; 2-2 -> 3-3 -> 4-4 -> 5-5 -> 6-6 -> 7-7 -> 8-8 -> 9-9 -> 10-10

3. B. Toutes les feuilles sont au même niveau, et la profondeur est en $\Theta(\log n)$. On parcourt une branche, n'importe laquelle.

4. A.

Exercice 2 *: Exercices [6 points]*

Question 1 Soit un arbre 2-3-4 qui contient n clés. Supposons que son sous-arbre gauche ne contienne que des 2-nœuds et son sous-arbre droit que des 4-nœuds. Montrer que le sous-arbre gauche contient $\Theta(\sqrt{n})$ nœuds.

Question 2 On souhaite utiliser la méthode du hachage extensible pour stocker les clés $A,L,G,O,R,I,T,H,M,Q,U,E,V,N,C$. Choisir (et indiquer) une fonction de hachage permettant d'associer

à chacune de ces lettres une valeur de hachage, puis construire étape par étape le tableau de hachage associé à des pages de capacité 4.

Solution

1. prendre des 4 nœuds partout sauf sur le sous-arbre gauche ou l'on prend uniquement des 2-nœuds nombre de clés : 3 (racine) + $2^h - 1$ (sous arbre gauche) $+3(4^h - 1)$ (les 3 autres) $=> h == \log_4 n$ donc dans le sous arbre gauche le nombre de clés est de l'ordre de $2^{log_4 n} = n^{Log2/Log4} = sqrt(n)$

2. Une fonction sur 4 bits suffit, mais ce n'est pas grave si une sur 5 bits est utilisée. $A \leftarrow 0000$ $C \leftarrow 0001$ $E \leftarrow 0010$ $G \leftarrow 0011$ $I \leftarrow 0100$ $L \leftarrow 0101$ $M \leftarrow 0111$ $N \leftarrow 1000$ $O \leftarrow 1001$ $Q \leftarrow 1010$ $R \leftarrow 1011$ $T \leftarrow 1100$ $U \leftarrow 1101$ $V \leftarrow 1110$.
Au départ tableau à 2 case et 2 pages contenant A,G,I,L et O,R,T. Puis on double le tableau mais on n'a que 3 pages (les 2 dernière cases pointent sur la même), avec A,G et H,I,L,M et O,Q,R,T. Puis on a besoin de 4 pages.

Exercice 3 *: Dichotomie dynamique [10 points]*

La recherche dichotomique dans un tableau trié consomme un temps logarithmique, mais le temps d'insertion d'un nouvel élément est linéaire par rapport à la taille du tableau. Dans tout l'exercice, la mesure de complexité correspond au nombre de comparaisons d'éléments. On peut améliorer le temps d'insertion en conservant séparément plusieurs tableaux triés. Plus précisément, on suppose que l'on souhaite implanter les opérations de recherche et d'insertion sur un ensemble E ayant n éléments. Supposons que n s'écrive $b_{k-1}\dots b_1 b_0$ en base 2. On a k tableaux triés $A_0, A_1, \dots, A_{k-1}$, de tailles respectives $2^0, 2^1, \dots, 2^{k-1}$. Chaque tableau A_i est soit plein, soit vide selon que $b_i = 1$ ou $b_i = 0$. Le nombre total d'éléments contenus dans les k tableaux est donc $b_0 \cdot 2^0 + b_1 \cdot 2^1 + \dots + b_{k-1} \cdot 2^{k-1} = n$. Bien que chaque tableau soit trié, il n'existe pas de relation particulière entre les éléments des différents tableaux.
Voilà un exemple :
Si $E = \{1, 2, \dots, 13\}$ alors n s'écrit 1101 en base 2. Le tableau A_1 est vide et les tableaux A_0, A_2 et A_3 sont pleins, par exemple : $A_0 = [13]$, $A_2 = [1, 6, 8, 11]$ et $A_3 = [2, 3, 4, 5, 7, 9, 10, 12]$ (mais ce n'est pas la seule manière de remplir ces trois tableaux).

Recherche :

Question 1 Décrire une méthode de recherche opérant sur la structure de données présentée dans l'introduction.

Question 2 Calculer la complexité de l'opération de recherche dans le pire cas.

Insertion :

Question 3 On suppose que la complexité (en nombre de comparaisons) de l'interclassement de deux tableaux triés de tailles t_1 et t_2 est égale à $t_1 + t_2$. On suppose que les tableaux $A_0, A_1, \dots, A_{i-1}$, de tailles $2^0, 2^1, \dots, 2^{i-1}$, sont pleins et triés. Montrer que l'on peut réaliser leur interclassement en au plus 2^{i+1} comparaisons (lorsque $i \geq 1$). Comment faut-il procéder ? Justifier la complexité.

Question 4

1. Décrire une méthode d'insertion d'un nouvel élément (distinct de tous éléments déjà présents) opérant sur la structure de données présentée dans l'introduction.
2. On suppose que A_0, A_1, A_2 et A_3 sont vides et que $A_4 = [1, 2, \dots, 16]$. Insérer 20.
3. On suppose que A_3 est vide et que $A_0 = [20], A_1 = [3, 13], A_2 = [1, 6, 8, 11]$ et $A_4 = [21, 22, \dots, 36]$. Insérer 12.

Question 5 Calculer la complexité de l'insertion (en nombre de comparaisons) dans le pire cas :

1. en fonction de i, où i est tel que $A_0, A_1, \dots, A_{i-1}$ sont pleins et A_i est vide
2. en fonction de n.

Coût amorti d'une insertion :

Question 6 Soit $n = 2^k - 1$. On effectue une suite de n insertions dans notre structure de données, initialement vide. Pour un entier $1 \leq \ell \leq n$ s'écrivant $c_{k-1}\dots c_1 c_0$ en binaire, on note $\alpha(\ell)$ le plus petit indice i tel que $c_i = 0$ (dans le cas où $\ell = n$, on définit $\alpha(\ell) = k$).

Montrer que la complexité amortie d'une insertion est $O(1)$.

Solution

1. On recherche par dichotomie dans chacun des tableaux.

2. La complexité vaut

$$\sum_{i=0}^{k-1} \ln 2^i = \sum_{i=0}^{k-1} i \ \ln 2 = \ln 2 \cdot \frac{k(k-1)}{2}.$$

Donc la recherche, au pire cas, se fait en $\Theta(\log^2 n)$. On peut faire plus simple en disant qu'on herche dans *logn* tableaux de taille au plus *logn*, mais on n'obtient que $O(\log^2 n)$.

3. On trie la suite de tableaux 2 à 2 en prenant à chaque fois les 2 plus petits. La complexité vaut $(2^0 + 2^1)(2^0 + 2^1 + 2^2) + \cdots + (2^0 + \ldots 2^i - 1)$. 2^0 est sommé $i - 1$ fois et sinon ($j \geq 1$), 2^j est sommé $(i - j)$ fois. Le nombre exact de comparaisons vaut $2^{i+1} - i + 1$.

4. Soit i le plus petit indice tel que le tableau A_i est vide. On interclasse tous les A_j pour $j < i$ et on insère le nouvel élément, on obtient un tableau A_i, et on vide les A_j.
On met 20 dans A_0
On suit l'algo pour remplir A_3 et vider les A_0, A_1, A_2.

5. $2^{i+1} + \log 2^i = \Theta(2^{i+1})$
$i \leq \log_2 n$ donc la complexité au pire vaut $\Theta(n)$.

6. la complexité amortie vaut le double par rapport à celle de l'incrémentation du compteur (exo 2.4)