DÉPARTEMENT D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

SIGLE DU COURS: IFT 2015 (E21)

NOM DU PROFESSEUR: Neil Stewart

TITRE DU COURS: Structures de Données

EXAMEN INTRA

Date : mardi-mercredi 8-9 juin, 2021
Heure : Examen disponible: mardi 8 juin 9:00; à remettre mercredi le 9 juin 11:00
Lieu : À la maison

DIRECTIVES PÉDAGOGIQUES:

- Vous disposez de 26 heures pour compléter cet examen.
- Toute documentation permise.
- Vous devez remettre *exactement* quatre pages en format pdf.
  La première page servira à identifier l'examen, mais elle doit surtout avoir une boîte rectangulaire avec votre nom de famille en majuscules, votre prénom, et votre matricule, style: STEWART, Neil. Matricule 235711.
- La réponse à la question i doit être mise sur la page $i + 1$, $i = 1, 2, 3$.

PLAGIAT. Constitue un plagiat:

- faire exécuter son travail par un autre
- utiliser, sans le mentionner, le travail d'autrui
- ÉCHANGER DES INFORMATIONS LORS D'UN EXAMEN
- falsifier des documents

Le plagiat est passible de sanctions allant jusqu'à l'exclusion du programme.

Figure 1: Style de la boîte exigée.

1. Question 1 (10 points) (Question relativement facile.)

   (a) Est-ce qu'il faut mettre la plus longue des deux sous-listes, obtenues dans une étape typique de *QuickSort,* sur la pile, ou bien la sous-liste la plus courte? Expliquez pourquoi (une preuve n'est pas exigée ici).

   (b) Si un arbre binaire non-vide n'est pas "Full", est-il possible que le nombre de feuilles soit égal au nombre de noeuds internes, plus un? Donnez la preuve, ou bien trouvez un contre-exemple.

   (c) À la page 239 de Weiss (il n'est même pas nécessaire de regarder cette page pour répondre à la question), il est montré qu'une certaine méthode pour trouver l'élément le k'ième plus grand dans une liste de N éléments exige $O(k + (N - k) \log k)$ unités de temps. "The first k elements are placed into the heap ..." [et ensuite d'autres opérations sont effectuées.] Démontrez que cela implique (comme Weiss le dit) que l'algorithme est $O(N \log k)$.

   (d) Dans le cours nous avons discuté des questions "Quels sont les problèmes?" et "quelles sont les méthodes?" qui sont impliqués dans notre étude.

   Nous avons remarqué dans le context du TAD *Liste* qu'il fallait décider, faut-il utiliser une liste chaînée, ou faut-il utiliser un tableau contigu? Tandis que dans le cas d'un *sac* (*bag* en anglais) il était évident que le tableau contigu était le meilleur choix. En utilisant la terminologie de "méthode" et "problème", expliquez dans une phrase simple pourquoi une si subite modification de notre conclusion est possible.

2. Question 2 *(10 points)*

Quand nous parlons du coût moyen de recherche dans un arbre binaire moyen, nous entendons que nous allons prendre un arbre moyen, où nous trouvons la moyenne par rapport à toutes les *formes* possible pour l'arbre, suivi d'un calcul de la moyenne par rapport à tous les *noeuds* possibles dans l'arbre moyen. Si nous parlons plutôt d'un arbre particulier, alors le coût moyen de recherche veut dire simplement la moyenne par rapport à tous les *noeuds* possibles dans l'arbre particulier.

(a) Soyons plus précis sur le nombre de comparaisons qu'il faut faire pour vérifier un noeud particulier dans l'arbre. Il y a trois possibilités, il va falloir alors deux comparaisons. Nous commençons par poser la question "la clef de recherche est-elle égale à la clef dans le noeud?", et si la réponse est négative, nous demandons si elle est plus grande ou plus petite? Si la clef est présente au niveau λ dans l'arbre, il nous faudra $2\lambda + 1$ comparaisons pour le savoir. Expliquez pourquoi. Comme conséquence, le nombre moyen de comparaisons pour une recherche réussie sera $C_N = (2 \cdot I(N) + N)/N$, où $I(N)$ est la longueur de chemin interne. Expliquez pourquoi.

(b) Démarrons avec un arbre parfaitement balancé τ_{gentil} avec les niveaux 0 à d inclusivement, et qui contient exactement $N = 2^{d+1} - 1$ noeuds.

i. Une valeur approximative de $I(N)$ pour cet arbre est $(N + 1)(\lg(N + 1) - 2$. Montrez que cette approximation est valide. (Indice: commencez par exprimer la valeur en fonction de d. Vous pouvez utiliser l'approximation habituelle: si une série a d termes, vous pouvez utiliser la série avec un nombre infini de termes comme approximation.)

ii. Alors quelle est la valeur approximative de C_N quand $N \to \infty$, si nous ne gardons que le terme dominant?

(c) À l'autre extrême, regardons l'arbre $\tau_{méchant}$ qui est simplement une liste linéaire de longueur N. Quelle est la valeur de $C(N)$ pour cet arbre?

(d) Entre ces deux extrêmes, nous pouvons regarder un arbre moyen, comme nous avons fait dans le cours; ici la *forme* de l'arbre a été choisie en regardant toutes les permutations possibles de l'ordre d'insertion des clef, et nous avons cherché la valeur espérée $D(N)$ de $I(N)$. La valeur espérée par rapport à tous les noeuds est ensuite donnée par $2D(N)/N \cong 2D(N)/(N + 1)$. (Vous pouvez prendre pour acquis le résultat démontré dans le cours.) Comment ça se compare avec les deux cas extrêmes déjà mentionnés, étant donné que $\ln 2 \cong 0.693$?

3. Question 3 *(10 points)*

(a) Un monceau emmagasiné dans un tableau, à partir de la case 1, est l'une des méthodes pour implanter une *Queue de Priorité.* Nous faisons ici l'hypothèse, comme dans le cours, que l'élément avec la plus petite clef a la plus haute priorité. Dans le but de garder les choses simples, faisons l'hypothèse que toutes les clefs sont distinctes.

Où dans la queue se trouve l'élément avec la *priorité la plus basse* (la clef la plus grande)? Comment trouver cette clef? Quel serait le coût de la trouver? (Notation $O(\ldots)$.) Donnez un exemple qui montre le plus mauvais cas.

(b) Dans la Question 2, nous avons dit que le nombre de comparaisons pour une recherche réussie est $C_N = (2I(N) + N)/N$. De façon semblable, le nombre de comparaisons pour une recherche échouée est $C'_N = 2E(N)/(N+1)$, où E est la longueur de chemin externe. En plus, dans l'un des Chat (aussi dans le Devoir 2), nous avons remarqué que $E(N) = I(N) + 2N$.

- Mettez ces faits ensemble pour montrer que

$$C'_N = \left(\frac{N}{N+1}\right)(C_N + 3) \tag{1}$$

(Note: il ne s'agit pas ici des C_N et C'_N du Devoir 2. Nous faisons plutôt le décompte du nombre de comparaisons ici. Vous n'avez pas besoin du Devoir 2 ici: mais ne soyez pas surpris par un manque de cohérence entre les deux choses.)

- Un argument semblable à la Question 2(b) du Devoir 2 montre que

$$C'_N = 4H_{N+1} - 4 \tag{2}$$

Vous pouvez prendre cette équation pour acquis.
Aussi, bien sûr

$$H_{N+1} = H_N + 1/(N+1) \tag{3}$$

Mettez ces trois faits ensemble pour montrer que

$$C_N = 4\left(1 + \frac{1}{N}\right)H_N - 7$$

Ce résultat est entièrement cohérent avec le résultat de la Question 2d ci-haut. Expliquez.

(Vous trouverez cette façon de démontrer le résultat dans certains livres. J'aime mieux notre preuve: la randomisation obtenue par la sélection de permutations est montrée explicitement, et le rôle des sous-arbres gauche et droites est montré explicitement. La preuve ici dépend bêtement de la manipulation algébrique.)

Fin de l'examen.

Neil Stewart