UNIVERSITÉ DE MONTRÉAL
Département IRO

Été 2021
Neil Stewart

IFT 2015 E21

Devoir 2.

10/10, soit 10% de la note finale.

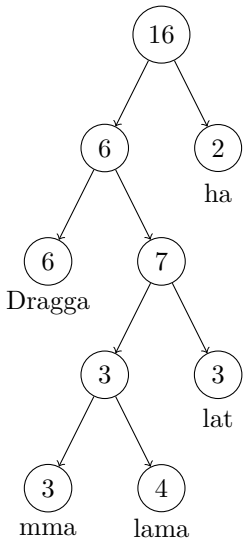
Les 10 points "Partie pratique" pour le cours E21 seront distribués de la façon suivante : Devoir #1 (1 point), Devoir #2 (3 points), Devoir #3 (6 points).

1 Partie Pratique (3 points)

Représenter une longue chaîne de caractère par une structure linéaire, comme un tableau, peu présenter certains inconvénients, surtout si la chaîne de caractère est assez longue. Par exemple, un retrait ou une insertion dans la chaine se ferait en temps linéaire, ce qui, dans certains contextes, peut être un peu trop lent.

Une *corde* (*rope* en anglais) est une structure de données servant à représenter une chaîne de caractères sous la forme d'un arbre binaire. Dans une corde, les noeuds feuilles contiennent des fragments de la chaine de caractères, tandis que les noeuds internes contiennent le nombre de caractères présents dans le sous-arbre gauche.

Voici un exemple d'une corde contenant la chaine "Draggammalamalatha" :



(Remarquez que cet exemple n'est pas unique, et qu'il existe beaucoup d'autre cordes pouvant représenter la même chaine).

En supposant l'arbre suffisamment balancé, l'avantage de la corde est qu'elle permet de faire certaines opération, comme un retrait ou une insertion, en temps logarithmique plutôt que linéaire. Par contre, certaines opérations qui pourraient être faites en temps constant si on utilisait un tableau se font en temps logarithmique dans une corde (ce qui reste tout de même assez petit). Pour maintenir ces complexités, il faudra à l'occasion rebalancer l'arbre, qui risque de se débalancer dépendamment des opérations qui sont faites dessus. Dans la pratique, ce genre de structure de données est parfois utilisé dans des éditeurs de texte.

Pour ce deuxième devoir, vous devez implanter cette structure de données en *Java*. Un fichier squelette nommé `Rope.java` vous est déjà fourni. Des instructions plus détaillées à propos des méthodes à implanter

sont décrites dans les commentaires du fichier. Veuillez remettre votre version complétée du fichier sur StudiUM, avec les noms et matricules des auteurs en commentaire au début du fichier.

Ne changez pas le nom du fichier, ni les signatures des méthodes déjà présentes. Si vous décidez d'ajouter une ligne "package" au début du fichier, retirez-la dans la version que vous soumettrez. Assurez-vous que votre fichier compile : il est difficile de donner des points pour du code qui ne compile pas ! Vous pouvez cependant (et vous êtes même encouragés à le faire !) vous ajouter des méthodes privées qui seraient utiles pour découper votre code.

Un fichier `Main.java` contenant quelques tests vous est aussi fourni. Vous pouvez l'utiliser comme bon vous semble pour tester votre implémentation. Vous n'avez pas à le remettre, et nous ne tiendrons pas compte des modifications que vous y ferez : nous ne regarderons que votre fichier `Rope.java`, ainsi que n'importe quel autre fichier que vous aurez créer vous-même et qui serait utiliser par `Rope.java`.

2 Partie Théorique (7 points)

1. (1 point)

   Ceux qui ne font pas attention aux détails reliés aux valeurs de départ, dans une preuve par induction, risquent de démontrer des choses qui ne sont pas vraies. Un exemple classique est la preuve que pour toute valeur de N, N personnes dans une même salle ont toujours le même âge. Faites un dessin d'une grande salle, avec une porte pour laisser entrer et sortir les personnes.

   Le théorème est vrai pour $N = 1$. Imaginons maintenant une salle avec un nombre arbitraire de personnes dans la salle. Supposons que mon énoncé est vrai pour une salle avec N personnes. Prenons une salle avec $N + 1$ personnes, $N + 1$ arbitraire, et montrons que l'énoncé est toujours vrai.

   Je sors l'une des $N+1$ personnes, disons Julien, de la salle, et je me sers de l'hypothèse inductive pour affirmer que toutes les personnes qui restent dans la salle, dont Samuel, ont le même âge. Ensuite, je fais rentrer Julien dans la salle et je fais sortir Samuel, et je me sers une deuxième fois de l'hypothèse inductive pour affirmer maintenant que toutes les personnes qui restent dans la salle, dont Julien, ont le même âge. En mettant ces deux résultats ensemble, Samuel a le même âge que tous les autres, Julien a le même âge que tous les autres, et donc tout le monde a le même âge.

   Il y a vraisemblablement une erreur dans ma preuve : où exactement est l'erreur ? (Ça vaut 0 points.)

   Dans la preuve du théorème sur la profondeur d'un arbre binaire moyen j'ai commencé par définir la valeur de I pour tous les arbres, à partir de $N = 1$ (arbre avec un seul noeud, $I = 0$), donc $N = 1, 2, 3, \ldots$. Après j'ai écrit que

   $$I(\tau_N) = I(\tau_i) + I(\tau_{N-i-1}) + N - 1, \tag{1}$$

   mais, un peu paresseux comme tout le monde, j'avais omis de mentionner une petite définition. Donnez un exemple d'un arbre avec, disons, $N \geq 3$, où des quantités dans l'équation (1) ne sont pas bien définies. Ensuite, fournissez la définition qui manque, et finalement, indiquez la conséquence pour la valeur de $D(0)$ (valeur utilisée d'ailleurs dans la preuve du théorème).

   (Je ne veux pas vous inquiéter : il n'y a pas d'erreur dans la preuve, il s'agit ici d'écrire les choses un peu plus au complet.)

2. (2 points)

   Le coût de recherche quand la clef n'est pas présente est une question importante pour les arbres de recherche binaire. Relié à cette question, l'idée d'étendre l'arbre binaire en conçevant les références nulles comme des feuilles dans un arbre étendu a déjà été mentionnée. Ces nouveaux noeuds-feuilles sont appelés en anglais "failure nodes", et en français nous dirons "noeuds-d'échec". Nous avons remarqué dans l'un des Chat que le nombre de noeuds d'échec, dans un arbre de recherche avec N noeuds, est égal à $N + 1$.

   La valeur de I est intéressant parce que le coût moyen de recherche (clef présente) pour un arbre particulier avec N noeuds est $C(N) = 1 + I(N)/N$. Si je fais la même chose pour la recherche de clefs

non-présentes, je dois définir E, la longueur de chemin extérieur ("External Path Length"), qui est égale à la somme des profondeurs des noeuds d'échec. Le coût moyen de recherche (clef absente) pour un arbre particulier avec N noeuds est $C'(N) = E(N)/(N+1)$.

Dans le Chat nous avons remarqué que

$$E(N) = I(N) + 2N \qquad\qquad N \geq 1. \tag{2}$$

et nous avons pensé faire une preuve par induction sur N.

(a) Démontrez que

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

(b) Il y a plus qu'une façon de faire une preuve de (2) par induction. Par exemple, nous pouvons regarder les sous-arbres, et procéder par l'induction forte ("strong induction", par exemple Johnsonbough, 7ième édition, p. 102). Prenons plutôt l'approche de regarder un noeud de profondeur maximale, et de procéder par induction simple en enlèvant ce noeud. Donnez la preuve de (2) en utilisant cette deuxième approche.

3. (2 points)

La fonction H_N est approximativement égal à $\ln N$, et dans le Tp5 du 2 juin vous avez vu démontrées des bornes logarithmiques inférieures et supérieures dans le cas spécial où N a la forme $N = 2^{m+1}$. Ce qui veut dire que H_N tend vers l'infinie, mais pas très vite. (Moins vite est meilleur pour nous.)

On pourrait même dire que la série H_N frôle la finitude, *i.e.*, elle échappe à peine de rester finie, parce que la série

$$H_N^{(r)} = \sum_{i=1}^{N} \frac{1}{i^r}$$

reste finie pour toute valeur de r strictement supérieure à 1.

Démontrez que $H^{(r)}_{2^m-1}$ est inférieure à $\sum_{k=0}^{m-1} \frac{2^k}{2^{kr}}$, ce qui est inférieure à $\frac{2^{r-1}}{2^{r-1}-1}$ (aussi à démontrer). De ce point de vue, H_N ne tend pas très vite vers l'infinie.

4. (2 points)

Weiss, Exercice 4.25, p. 163, celui qui commence "How many bits are required … ?". J'ajoute des questions. L'exercice n'est pas si difficile, le but de Weiss est de vous faire remarquer quelque chose.

(a) Un estimé $O(\ldots)$ suffit.

(b) Supposons la représentation complément-à-deux, de sorte que les 8 bits emmagasinent les entiers entre -128 et 127. Weiss entend "What is the *height* of the smallest … ? ".

(c) Quelle est la pertinence de la question ? Pourquoi Weiss l'a-t-il posé ?

(d) Si la réponse à la partie 4b est h, donnez une borne inférieure grossière pour G_h, $G_h >> \phi^{h+3}$. Parce que $\phi > 1.618$, en faisant le carré cinq fois nous pouvons facilement calculer que $\phi^{32} >> 4{\cdot}10^6$ par exemple. Quelle est la pertinence de ma question ?

À réaliser en équipes de 1 ou 2. À remettre le 16 juin, 2021, avant `8:00`. Les devoirs en retard ne seront pas acceptés.