DÉPARTEMENT D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

SIGLE DU COURS: IFT 2015 (E16)

NOM DU PROFESSEUR: Neil Stewart

TITRE DU COURS: Structures de Données

EXAMEN FINAL

Date : Lundi, 18 juillet, 2015
Heure : 13:00-16:00
Lieu : Z-330

DIRECTIVES PÉDAGOGIQUES:

- Vous disposez de trois heures pour compléter cet examen.
- Documentation permise: une page 8" x 11", recto-verso.
- Mettez tout de suite votre nom et votre matricule dans la case (Figure 1).
- Répondez sur l'examen. Il y a trois pages brouillons (page 5, page 9 et page 13).
- L'espace alloué pour la réponse indique la longueur de la réponse cherchée.
- Il y a 12 questions, avec 100 points au total.

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.

	* = 5 pts
1.	
2*.	
3*.	
4.	
5.	
6.	
7*.	
8*.	
9.	
10.	
11.	
12.	

Figure 1: Mettez votre nom et matricule ici.

1. Question 1 *(10 points)*

   Skip-list déterministe

   (a) Dans le cas du retrait d'une Skiplist déterministe, nous étions souvent dans l'obligation d'utiliser une *gap* voisine. Nous avons utilisé la *gap* 2 si nous devions descendre dans la *gap* 1, et nous avons utilisé la *gap* i si nous devions descendre dans la *gap* $i+1$, $1 \leq i \leq n-1$. Le but de "changer de direction" était de s'assurer de pouvoir utiliser la *gap* plus loin comme voisine.

   Avec toutes les bonnes questions que vous m'avez posées, j'étais surpris que personne ne m'a pas demandé ce qui arrive si $n = 1$: *i.e.*, s'il n'y a pas de voisine.

   Dessinez la Skiplist déterministe qui correspond à l'arbre 2-4 qui a trois noeuds, avec clefs 10, 20, et 30, une clef dans chaque noeud, avec 10 et 30 comme enfants de 20. La Skiplist aura les niveaux $h = 1$, $h = 2$ et $h = 3$.

   (b) Quel est l'écart entre le noeud Entête et le noeud Terminal (voir E et T par exemple, page 10) au niveau $h = 3$?

   (c) Selon les règles énoncées, qu'elle est la première chose que nous sommes supposés faire? Quelle est la difficulté?

   (d) Est-ce que mon exemple d'arbre 2-4 est très spécial, ou est-ce que ce cas va arriver souvent?

   (e) Quoi faire?

2. Question 2 *(5 points)*

   Mises à jour des arbres de recherche: "Top-down" vs "Bottom-up"

   Nous avons vu un algorithme "bottom-up" pour l'insertion dans le cas des arbres 2-3. Par contre, il a été dit (même si nous n'avons pas pu voir les détails) qu'il existe un algorithme "top-down" pour la mise-à jour des arbres Rouge-Noir, et les algorithmes pour les Skiplist déterministes (voir par exemple la Question 1) donnent évidemment des algorithmes "top-down" pour la mise-à-jour des arbres 2-4. Mentionnez une difficulté pour l'implantation des algorithmes "bottom-up" de mise-à-jour, et expliquez comment cette difficulté peut être contournée (autre que de remplacer l'algorithme par un algorithme "top-down").

3. Question 3 *(5 points)*

   Clefs formées de caractères

   Expliquez brièvement comment on pourrait définir une fonction $h(x)$ qui servirait quand la clef x est une chaîne de caractères.

4. Question 4 *(10 points)*

Chemins de recherche (addressage ouvert)

Supposons que vous utilisiez le hâchage double, en commençant avec la clef x_0. Le calcul de $h(x_0)$ a fait que x_0 est emmagasinée dans le tableau à l'adresse $h(x_0)$.

Maintenant, vous essayez d'insérer la clef x_1 et vous constatez que $h(x_1) = h(x_0)$. Vous calculez donc $p(x_1)$, et vous suivez le "chemin de collision". Supposons que les deux prochaines cases que vous visitez sur le chemin soient occupées.

(a) Si vous calculez la valeur de $p(x_0)$ et vous constatez que la case $h(x_0) - p(x_0)$ est vide, il y a une façon de réduire le coût des recherches ultérieures. Comment?

(b) Y'a-t-il une façon de généraliser cette idée au cas où les trois prochaines cases que vous visitez sur le chemin sont occupées? Et quatre ... etc?

(c) Quel est le compromis si vous utilisez cette méthode pour réduire le coût des recherches ultérieures?

Page brouillon: cette page ne sera pas prise en compte lors de la correction.

5. Question 5 *(10 points)*

Splay Tree

La règle ZIG-ZAG pour les Splay Tree dit que si X est l'enfant droit de P, et P est lui-même l'enfant gauche de G, alors après la rotation, P sera l'enfant gauche de X et G sera l'enfant droit de X. Aussi, la règle ZIG-ZIG dit que si X est l'enfant gauche de P, et P est lui-même l'enfant gauche de G, alors après la rotation, P sera l'enfant droit de X et G sera l'enfant droit de P.

Supposons que les clefs 3, 2, 1, 4 sont insérées dans l'ordre, et que nous accédions après à la clef 3. Dessinez les cinq Splay Tree correspondant à ces cinq opérations.

6. Question 6 *(10 points)*

Prim/Dijkstra

(a) Expliquez ce qu'il faut faire pour transformer l'algorithme de Prim en algorithme de Dijkstra.

(b) Quelle est l'interpretation précise des distances dans ce cas?

(c) Donnez un résumé des avantages ou inconvénients d'utiliser un monceau, plutôt que d'utiliser une simple colonne dans un tableau, pour emmagasiner les distances dans le cas de l'algorithme de Dijkstra. Exprimez-vous de façon informelle en utilisant $O(\ldots)$ pour dire "est de l'ordre de".

7. Question 7 *(5 points)*

   Représentation des graphes

   Expliquez brièvement la différence entre la matrice d'adjacence, et les listes d'adjacence, dans le contexte de la représentation des graphes.

8. Question 8 *(5 points)*

   Représentation des ensembles

   Dans le contexte de l'algorithme Union/Find, une représentation d'arbres a été utilisée. Expliquez *très* brièvement les similarités, et les différences, entre cette représentation et la représentation des arbres "complete".

Page brouillon: cette page ne sera pas prise en compte lors de la correction.

9. Question 9 *(10 points)*

Équivalences de représentations

Dans la Figure 2 nous voyons un exemple de SkipList déterministe. Les niveaux dans l'exemple sont 1, 2, 3 et 4.

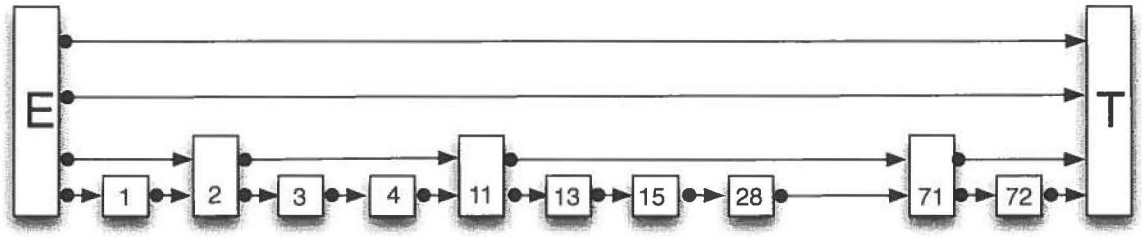

Figure 2: Skiplist déterministe

Dessinez un arbre 2-4 équivalent, et aussi un arbre Rouge-Noir équivalent. (Dans ce dernier cas, dans le cas de 2 clefs mettez la plus petite clef dans le noeud rouge. Indiquez les noeuds rouges avec un cercle très foncé, et les noeuds noirs avec un cercle peu foncé.)

Question 10 *(10 points)*

Parcours d'un graphe

Soit un graphe non-orienté (V, E), où $V = \{A, B, C, D\}$, qui est le graphe complet: chaque noeud est relié à tous les autres noeuds. Les listes d'adjacence sont $A : B, C, D$; $B : C, D, A$; $C : D, A, B$; $D : A, B, C$.

(a) Montrez le fonctionnement de l'algorithme pour un parcours en profondeur *dfs*.

(b) L'algorithme *dfs* trouve un arbre sous-tendant du graphe. Dessinez l'arbre sous-tendant (qui n'aura peut-être pas la forme classique d'un arbre dans le cas actuel!)

Question 11 *(10 points)*

Dans le cas de parcours de *listes* nous avons utilisé une variable *mpf* qui distinguait les deux cas: sommes nous montés par une ficelle, ou sommes nous arrivés par un lien ordinaire? Montrez un cas très simple qui montre qu'il est obligatoire de savoir, dans l'algorithme, de quel cas il s'agit.

Question 12 *(10 points)*

(a) Donnez les quatre catégories de liste généralisées qui ont été mentionnées dans le cours.

(b) Il y a une relation très étroite entre l'une de ces catégories et la représentation "puînée" d'un arbre ordonné. Expliquez cette relation avec un dessin.

Page brouillon: cette page ne sera pas prise en compte lors de la correction.

Fin de l'examen.

Neil Stewart