IFT2015 A12 — Examen Intra

Miklós Csűrös

14 novembre 2012

English translation on the last three pages.

Aucune documentation n'est permise. L'examen vaut 100 points. Vous pouvez avoir jusqu'à 15 points de boni additionnels.

► **Répondez à toutes les questions dans les cahiers d'examen.**

F0 Votre nom (1 point)

► Écrivez votre nom et code permanent sur tous les cahiers soumis.

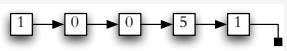
F1 Taux de croissance (20 points)

► Remplissez le tableau suivant : chaque réponse vaut 2 points. Pour chaque paire f, g, écrivez "=" si $\Theta(f) = \Theta(g)$, "$\ll$" si $f = o(g)$, "$\gg$" si $g = o(f)$, et "???" si aucun des trois cas n'applique. Il n'est pas nécessaire de justifier vos réponses. $\lg n$ dénote le logarithme binaire de n.

	$f(n)$	$g(n)$
a	$f(n) = n^2$	$g(n) = (2015n + 1)^2$
b	$f(n) = \lg n$	$g(n) = \lg(n^{2015})$
c	$f(n) = \sum_{i=0}^{n} 2^i$	$g(n) = 2^n$
d	$f(n) = \sum_{i=1}^{n} 1/i$	$g(n) = \ln n$
e	$f(n) = n!$	$g(n) = (2n)!$
f	$f(n) = n^{2015}$	$g(n) = (2n)^{2015}$
g	$f(n) = n$	$g(n) = \begin{cases} 1 & \{n = 0, 2, 4, 6, 8, \dots\} \\ n & \{n = 1, 3, 5, 7, 9, \dots\} \end{cases}$
h	$f(n) = n \ln n$	$g(n) = \lg(n!)$
i	$f(n) = 4^n$	$g(n) = \begin{cases} 1 & \{n = 0\} \\ 4^{g(n-1)} & \{n > 0\} \end{cases}$
j	$f(n) = 2^{2^n}$	$g(n) = 4^n$

F2 Polynômes (30 points)

On veut représenter des polynômes avec une seule variable par une liste chaînée. Pour un polynôme $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_dx^d$, on utilise une liste avec $(d+1)$ nœuds. Le premier nœud stocke le coefficient a_0, le deuxième a_1, etc.



Liste représentant $f(x) = x^4 + 5x^3 + 1$, un polynôme de degré 4.

a. Degré du polynôme (15 points) Le *degré* d'un polynôme est l'indice du dernier coefficient non-zéro ($\max\{i : a_i > 0\}$), ou 0 si $a_0 = 0$.

▶ Donnez un algorithme degree(N) pour calculer le degré d'un polynôme spécifié par la tête (nœud N contenant a_0) de sa liste. Notez que tout a_i (même a_d) peut être 0. La liste vide (null) représente $f(x) = 0$.

b. Évaluation (15 points) Pour évaluer la valeur d'un polynôme $f(x)$ à une x donnée, on utilise la règle Horner. On écrit

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_dx^d = \underbrace{\Big(\Big(\big((a_d)x + a_{d-1}\big)x + a_{d-2}\Big)x + \cdots + a_0\Big)}_{(d+1) \text{ parèntheses imbriqués}}$$

Par exemple, $2x^3 + x^2 - x + 7 = ((2x + 1)x - 1)x + 7$. En général, l'évaluation se fait par d multiplications et d additions : c'est plus économique que l'évaluation directe quand on calcule $x, x^2, \ldots, x^d$ séparamment.

▶ Donnez un algorithme *récursif* eval(N, x) pour évaluer un polynôme (tête à nœud N) par la règle Horner.

♡ **c. Addition de polynômes éparses (15 points boni)** Pour un polynôme *éparse* avec beaucoup de coefficients 0 (p.e., $x^{88} + 1$), il vaut mieux stocker juste les termes non-zéro. Dans une telle implantation, chaque nœud contient un paire (a_i, i) de coefficient (a_i) et d'exposant (i). Les nœuds sont dans un ordre strictement croissant de i.

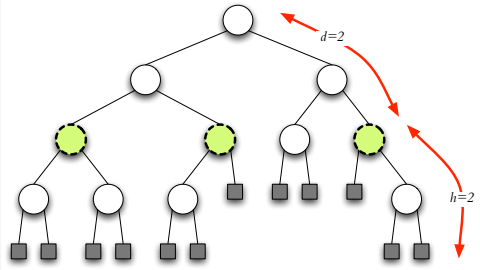
▶ Donnez un algorithme pour calculer $f(x) + g(x)$ quand f et g sont des polynômes. L'algorithme prend les deux têtes de liste, et retourne la tête de la liste représentant le résultat. Par exemple, si $f(x) = x^{88} + 1$ (2 nœuds) et $g(x) = 6x^2 - x - 1$ (3 nœuds), alors $f(x) + g(x) = x^{88} + 6x^2 - x$ comprend trois nœuds $(-1, 1)$, $(6, 2)$, $(1, 88)$, dans cet ordre.

F3 $\Theta \leftrightarrow o$ (14 points)

Soit $f, g\colon \{0, 1, 2 \dots\} \mapsto (0, \infty)$ deux fonctions positives avec $f(n) = \Theta\Big(2^{g(n)}\Big)$.

► Démontrez que si $\lim_{n\to\infty} g(n) = \infty$, alors $\lg f(n) = \big(1 + o(1)\big)g(n)$.

F4 La médiane d'un arbre (20 points)



Dans cet exercice, un nœud se dit *médian* si sa hauteur est égale à sa profondeur.

► Donnez un algorithme *récursif* qui affiche tous les nœuds médians dans un arbre binaire. L'algorithme doit prendre un temps linéaire dans la taille de l'abre.

Indice: *Passer la profondeur comme argument et retourner la hauteur.*

F5 Suppression dans le tas binaire (15 points)

► Donnez un algorithme delete(H, i, n) pour supprimer un élément dans un tas binaire $H[1..n]$ par son indice i dans le tableau. (L'argument n dénote le nombre d'éléments : la capacité du tableau peut être plus grande que ça.) L'algorithme a le droit de réarranger les éléments dans le tableau, mais $H[1..n-1]$ doit contenir la reste des éléments dans l'ordre de tas après suppression. Montrez tous les détails de l'algorithme.

Indice: *Pour $i = 1$, le déroulement est identique à* deleteMin *; autrement, il faut examiner le parent et les enfants de la case i et faire sink ou swim.*

BONNE CHANCE !

English translation

No documentation is allowed. The examen is worth 100 points, and you can collect up to 15 additional bonus points. You may write your answers in English or in French.

Answer each question in the exam booklet.

E0 Your name (1 point)

▶ Write your name and *code permanent* on each booklet that you submit.

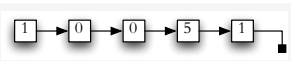
E1 Growth rates (20 points)

▶ Fill out the following table : every answer is worth 2 points. For each pair f, g, write "=" if $\Theta(f) = \Theta(g)$, "$\ll$" if $f = o(g)$, "$\gg$" if $g = o(f)$, and "???" if neither of the three applies. You do not need to justify the answers. $\lg n$ denotes the binary logarithm of n.

	$f(n)$	$g(n)$
a	$f(n) = n^2$	$g(n) = (2015n + 1)^2$
b	$f(n) = \lg n$	$g(n) = \lg(n^{2015})$
c	$f(n) = \sum_{i=0}^{n} 2^i$	$g(n) = 2^n$
d	$f(n) = \sum_{i=1}^{n} 1/i$	$g(n) = \ln n$
e	$f(n) = n!$	$g(n) = (2n)!$
f	$f(n) = n^{2015}$	$g(n) = (2n)^{2015}$
g	$f(n) = n$	$g(n) = \begin{cases} 1 & \{n = 0, 2, 4, 6, 8, \dots\} \\ n & \{n = 1, 3, 5, 7, 9, \dots\} \end{cases}$
h	$f(n) = n \ln n$	$g(n) = \lg(n!)$
i	$f(n) = 4^n$	$g(n) = \begin{cases} 1 & \{n = 0\} \\ 4^{g(n-1)} & \{n > 0\} \end{cases}$
j	$f(n) = 2^{2^n}$	$g(n) = 4^n$

E2 Polynomials (30 points)

We want to use a linked list to represent polynomials over a single variable. For a polynomial $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_dx^d$, the list has $(d+1)$ nodes. The first node stores coefficient a_0, the second a_1, and so on.



List representing x^4+5x^3+1, a polynomial of degree 4.

a. Degree (15 points). The degree of the polynomial is the largest power of the variable with non-zero coefficient ($\max\{i\colon a_i > 0\}$), or 0 if $a_0 = 0$. ▶ Give an algorithm $\mathsf{degree}(N)$ that computes the degree of a polynomial stored in a list starting with node N (containing a_0) Note that any a_i, including a_d may be 0. The empty list (null) corresponds to $f(x) = 0$.

b. Evaluation (15 points). One uses Horner's rule to evaluate a polynomial $f(x)$ at a given value x. Write

$$f(x) = a_0+a_1x+a_2x^2+\cdots+a_dx^d = \underbrace{\Big(\Big(\big((a_d)x + a_{d-1}\big)x + a_{d-2}\Big)x + \cdots + a_0\Big)}_{(d+1)\text{ nested parentheses}}$$

For instance, $2x^3 + x^2 - x + 7 = ((2x+1)x - 1)x + 7$. In general, the evaluation is done using d additions and d multiplications : this is faster than a naïve evaluation in which one computes $x^1, x^2, \ldots, x^d$ separately.

▶ Give a *recursive* algorithm $\mathsf{eval}(N, x)$ that evaluates a polynomial (constant term stored at the list head N) by Horner's rule.

♡ **c. Addition of sparse polynomial (15 bonus points)** For a *sparse* polynomial with many 0 coefficients (such as $f(x) = x^{88} + 1$), it is better to keep only the non-zero terms. One can use thus a linked list where the nodes store pairs (a_i, i) of non-zero coefficients a_i and powers i. (The nodes are kept in a strict increasing order of powers i.)

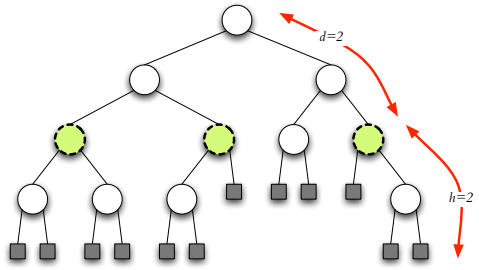
▶ Give an algorithm that computes $f(x) + g(x)$ symbolically for two polynomials f, g. The algorithm takes the two list heads as arguments, and returns the head for the list storing the result. For instance, if $f(x) = x^{88} + 1$ (2 nodes) and $g(x) = 6x^2 - x - 1$ (3 nodes), the result $f(x) + g(x) = x^{88} + 6x^2 - x$ consists of 3 nodes $(-1, 1)$, $(6, 2)$, $(1, 88)$.

E3 $\Theta \leftrightarrow o$ (14 points)

Let $f, g\colon \{0, 1, 2 \dots\} \mapsto (0, \infty)$ be two positive functions with $f(n) = \Theta\Big(2^{g(n)}\Big)$.

► Show that if $\lim_{n\to\infty} g(n) = \infty$, then $\lg f(n) = \big(1 + o(1)\big)g(n)$.

E4 Median of a tree (20 points)



In this exercise, a node is called *median*, if its height equals its depth.

► Give a *recursive* algorithm that lists all the median nodes in a binary tree. The algorithm must take linear time in the tree size.

Hint: *Pass the depth as argument, and return the height.*

E5 Deletion in a binary heap (15 points)

► Give an algorithm delete(H, i, n) that deletes an element in a binary heap $H[1..n]$ specified by its index i in the table. (The argument n denotes the number of elements in the heap : the table might have a larger capacicity than that.) The algorithm may rearrange the elements in the table, but $H[1..n-1]$ must contain the remaining elements after the deletion. Show all the details of the algorithm.

Hint: *For $i = 1$, the execution is identical to* deleteMin *; otherwise, inspect the parent and the children of cell i to decide between sink and swim.*

GOOD LUCK !