

IFT2015 STRUCTURES DE DONNÉES
LISTE DÉTAILLÉE DE SUJETS À L'INTRA

Miklós Csűrös

Département d'informatique et de recherche opérationnelle
Université de Montréal

Automne 2012

F0 Introduction

Le but de ce document est de définir les connaissances requises dans le cours IFT2015 à l'examen intra. Cet examen constitue aussi la première partie de l'examen pré-doctorale en structures de données.

La connaissance des sujets marqués par \star est exigée pour un «B/A-». Les sujets marqués par $\star\star$ correspondent plutôt à un niveau «A+/A». Les notes marginales sont des références au livre de Sedgewick [*Algorithmes en Java*]. Les notes de cours sont disponibles à <http://www.iro.umontreal.ca/~csuros/IFT2015/A12/>.

Aucune documentation ne sera permise à l'examen intra.

F1 Principes d'analyse d'algorithmes

Références

- ▷ Sedgewick chapitre 2
- ▷ Notes sur les fondations : `notes01-recursion.pdf`
- ▷ Notes sur l'analyse d'algorithmes : `notes04-analysis.pdf`

Sujets

- \star Principes de base : pire cas, meilleur cas, moyen cas. S§2.1,2.2,2.7
- \star Croissance de fonctions communes : constantes, logarithmiques, polynomiales, exponentielles. Factorielle ($n!$), approximation de Stirling, nombres Fibonacci ($F_n = F_{n-1} + F_{n-2}$), nombres harmoniques ($H_n = \sum_{i=1}^n 1/i = \ln n + \gamma + o(1)$). S§2.3
- \star Notion de temps amorti.
- \star Notation asymptotique : définitions de grand $O(f)$, petit $o(f)$, $\Theta(f)$ et $\Omega(f)$. Expressions avec $O()$ ou $o()$, règles d'arithmétique : $O(f) + O(g)$, $O(f) \cdot O(g)$. Relations avec la limite S§2.4

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0 \quad \Rightarrow f(n) = O(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad \Leftrightarrow f(n) = o(g(n)).$$

Comparaison de fonctions de forme $f(n) = \sum_i a_i b_i^n n^{c_i} (\lg n)^{d_i}$.

- ★ Techniques de preuve. Application directe de la définition pour démontrer $f = O(g)$ ou $f = o(g)$.
- ★ Récurrences typiques. S§2.5,2.6

| | |
|--------------------------|-----------------------|
| $f(n) = f(n - 1) + O(1)$ | $f(n) = O(n);$ |
| $f(n) = f(n - 1) + O(n)$ | $f(n) = O(n^2);$ |
| $f(n) = f(n/2) + O(1)$ | $f(n) = O(\log n);$ |
| $f(n) = f(n/2) + O(n)$ | $f(n) = O(n);$ |
| $f(n) = 2f(n/2) + O(1)$ | $f(n) = O(n);$ |
| $f(n) = 2f(n/2) + O(n)$ | $f(n) = O(n \log n);$ |

- ★★ Preuve par induction pour récurrences asymptotiques.
- ★★ Solution de récurrences par substitution de variables.

F2 Structures élémentaires et types abstraits

Références

- ▷ Sedgewick chapitres 3 et 4
- ▷ Notes sur les listes : [notes02-linkedlist.pdf](#).
- ▷ Notes sur les tableaux : [notes03-tableaux.pdf](#).

Sujets

- ★ Blocs de construction pour programmes Java. S§3.1
- ★ Tableaux. S§3.2
- ★ Listes chaînées. Variations : listes circulaires, doublement chaînées. Sentinelles pour la tête et/ou la queue. Manipulation d'éléments sur la liste, insertion et suppression. Parcours d'une liste. S§3.3,3.4
- ★ Notion d'un type abstrait, interface, implantation, client. S§4.1
- ★ Types abstraits de files généralisées, piles et queues/files FIFO. S§4.2,4.7
- ★ Implémentations de pile et de queue par tableaux ou listes chaînées. Efficacité d'implantations différentes (temps de calcul pour les opérations standards). Débordement. S§4.4,4.5,4.7

F3 Réursion et arbres

Références

- ▷ Sedgewick chapitre 5
- ▷ Notes sur les arbres : `notes05-trees.pdf`

Sujets

- ★ Algorithmes récursifs. Diviser pour régner. S§5.1,5.2
- ★ Terminologie pour structures arborescentes : arbre k -aire, hauteur, niveau, profondeur. Implémentation d'un arbre. S§5.4
- ★ Propriétés d'arbres binaires (relations entre le nombre de nœuds internes et externes ou la hauteur). S§5.5
- ★ Parcours d'un arbre : préfixe/préordre, infixé/dans l'ordre, postfixe/postordre, S§5.6 ordre de niveau.
- ★ Algorithmes récursifs sur les arbres : calcul de taille, hauteur ou profondeur de sous-arbres. S§5.7

F4 File de priorité

Références

- ▷ Sedgewick sections 9.1–9.5.
- ▷ Notes sur les files à priorités : `notes06-heap.pdf`

Sujets

- ★ Type abstrait de file de priorité min-tas/max-tas : opérations `insert`, `deleteMin` S§9.1,9.5 ou `deleteMax`. Implantations par tableau ou liste chaînée.
- ★ Arbre en ordre de tas. Manipulation du tas : nager et couler (heapsort S§9.2,9.3 montante et descendante). Tas binaire, sa représentation dans un tableau.
- ★ `heapify` (établissement de l'ordre de tas dans un tableau). S§9.4
- ★★ Tas *d*-aire.
- ★ Tri par tas, son temps de calcul et usage de mémoire.



E0 Introduction

Topics for a «B/A-» level are denoted by \star ; $\star\star$ denote somewhat more advanced topics for «A+/A» level. The margin notes refer to Sedgewick's *Algorithms in Java*. The class notes are available on the webpage <http://www.iro.umontreal.ca/~csuros/IFT2015/A12/>.

No documentation is allowed at the examen.

E1 Principles of algorithm analysis

References

- ▷ Sedgewick chapter 2
- ▷ Notes on the foundations : `notes01-recursion.pdf`
- ▷ Notes on algorithm analysis : `notes04-analysis.pdf`

Topics

- \star Basic principles : worst case, best case, average case. S§2.1,2.2,2.7
- \star Growth of common functions : constants, logarithms, polynomials, exponentials. Factorial ($n!$), Fibonacci numbers ($F_n = F_{n-1} + F_{n-2}$), harmonic numbers ($H_n = \sum_{i=1}^n 1/i$). S§2.3
- \star Notion of amortized cost.
- \star Asymptotic notation : definitions of big-Oh $O(f)$, small-oh $o(f)$, $\Theta(f)$, and $\Omega(f)$. Arithmetic expressions involving asymptotics, rules : $O(f) + O(g)$, $O(f) \cdot O(g)$. Connections to lim S§2.4

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = c > 0 \quad \Rightarrow f(n) = O(g(n));$$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad \Leftrightarrow f(n) = o(g(n)).$$

Comparing functions in the form $f(n) = \sum_i a_i b_i^n n^{c_i} (\lg n)^{d_i}$.

- \star Proof techniques. Using the definitions to prove $f = O(g)$ or $f = o(g)$.
- \star Basic recurrences. S§2.5,2.6

| | |
|--------------------------|-----------------------|
| $f(n) = f(n - 1) + O(1)$ | $f(n) = O(n);$ |
| $f(n) = f(n - 1) + O(n)$ | $f(n) = O(n^2);$ |
| $f(n) = f(n/2) + O(1)$ | $f(n) = O(\log n);$ |
| $f(n) = f(n/2) + O(n)$ | $f(n) = O(n);$ |
| $f(n) = 2f(n/2) + O(1)$ | $f(n) = O(n);$ |
| $f(n) = 2f(n/2) + O(n)$ | $f(n) = O(n \log n);$ |

- ★★ Variations on basic recurrences : “guessing” the asymptotics [iterated substitutions].
- ★★ Proof by induction for asymptotic recurrences.
- ★★ Variable substitution for solving recursions.

E2 Elementary structures and abstract data types

References

- ▷ Sedgewick chapters 3 and 4
- ▷ Notes on lists : `notes02-linkedlist.pdf`
- ▷ Notes on tables : `notes03-tableaux.pdf`

Topics

- ★ Java building blocks. S§3.1
- ★ Tables. S§3.2
- ★ Linked lists. Variations : circular, doubly-linked lists. Sentinels for the head and/or tail. Manipulation of elements, insertion and deletion. List traversal. S§3.3,3.4
- ★ Concept of an abstract type, interface, implementation, client. S§4.1
- ★ Abstract types for stacks, queues and generalized queues, S§4.2,4.7
- ★ Implementations of stack and queue by tables or linked lists. Running time for standard operations in different implementations. Overflow/underflow. S§4.4,4.5,4.7

E3 Recursion and trees

Reference

- ▷ Sedgewick chapter 5
- ▷ Notes on recursion techniques :
- ▷ Notes on trees : `notes05-trees.pdf`

Topics

- ★ Recursive algorithms. Divide-and-conquer. S§5.1,5.2
- ★ Terminology for tree structures : k -ary tree, height, level, depth. Tree im- S§5.4
plementations.
- ★ Mathematical properties of binary trees (relationships between number of S§5.5
internal and external nodes, height)
- ★ Tree traversal : preorder, inorder, postorder, level-order. S§5.6
- ★ Recursions on trees : computing the size, height, or depth of subtrees. S§5.7

E4 Priority queues

References

- ▷ Sedgewick sections 9.1–9.5.
- ▷ Notes on priority queues : `notes06-heap.pdf`

Topics

- ★ ADT for priority queue : operations insert, deleteMin or deleteMax. Implementations by table or linked list. S§9.1,9.5
- ★ Heap order for a tree. Heap manipulation : swim and sink. Binary heap, its representation in a table. S§9.2,9.3
- ★★ d -ary heap.
- ★ heapify (linear-time construction of heap order in a table). S§9.4
- ★ Heapsort, its running time and memory requirements.