IFT2015 Structures de données
Liste détaillée de sujets à l'intra

Miklós Csűrös

Département d'informatique et de recherche opérationnelle
Université de Montréal

Automne 2013

The English translation starts on page 7

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 ⋆ est exigée pour un «B/A-». Les sujets marqués par ⋆⋆ correspondent plutôt à un niveau «A+/A».

★ Les notes marginales sont des références aux ouvrages suivants

S Sedgewick, R. *Algorithmes en Java*, 3e édition (2004)

SW Sedgewick, R. et K. Wayne. *Algorithms*, 4e édition (2011)

✶ Les notes de cours et des liens vers des articles Wikipedia sont affichés sur le site `http://www.iro.umontreal.ca/~csuros/IFT2015/A13/`.

✷ Aucune documentation ne sera permise à l'examen intra.

F1 Principes d'analyse d'algorithmes

Références

- ▷ Sedgewick chapitre 2 ; Sedgewick & Wayne section §1.4
- ▷ Note sur les fondations : `notes01-recursion.pdf`.
- ▷ Note sur l'analyse d'algorithmes : `notes04-analysis.pdf`.
- ▷ Note sur les aspects pratiques : `notes05-experiments.pdf`.

Sujets

- ⋆ Principes de base : pire cas, meilleur cas, moyen cas. S§2.1,2.2,2.7
- ⋆ 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 ($H_n = \sum_{i=1}^{n} 1/i = \ln n + \gamma + o(1)$), logarithme itéré. S§2.3
- ⋆⋆ Fonction d'Ackerman et son inverse.
- ⋆ Notion de temps amorti.
- ⋆ Notation asymptotique : définitions de grand $O(f)$, petit $o(f)$, $\Theta(f)$ et $\Omega(f)$. Asymptotiques exactes $f \sim g$. Expressions avec $O()$ ou $o()$, règles d'arithmétique : $O(f) + O(g)$, $O(f) \cdot O(g)$. Relations avec la limite S§2.4

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c > 0 \qquad \Rightarrow \qquad f(n) = O\big(g(n)\big);$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \qquad \Leftrightarrow \qquad f(n) = o\big(g(n)\big);$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1 \qquad \Leftrightarrow \qquad f(n) \sim g(n)$$

- ⋆ Application directe de la définition pour démontrer $f = O(g)$ ou $f = o(g)$.
- ⋆⋆ Preuve par induction pour récurrences asymptotiques.
- ⋆ Détermination informelle du temps de calcul et d'usage de mémoire pour algorithmes (itératifs) simples
- ⋆ Récurrences simples. S§2.5,2.6

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

- ⋆ Validation expérimentale de temps de calcul

F2 Structures élémentaires et types abstraits

Références

- ▷ Sedgewick chapitres 3 et 4 ; Sedgewick & Wayne sections 1.1–1.3
- ▷ Note sur les listes : `notes02-linkedlist.pdf`.
- ▷ Note sur les tableaux : `notes03-tableaux.pdf`.

Sujets

⋆ Blocs de construction pour programmes Java.	S§3.1 ;SW§1.1
⋆ 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
⋆ Gestion de mémoire pour listes.	S§3.5
⋆ Notion d'un type abstrait, interface, implantation, client.	S§4.1 ;SW§1.2
⋆ Types abstraits de files généralisées, piles et queues/files FIFO.	S§4.2,4.7
⋆ Implantations de pile et de queue par tableaux ou listes chaînées. Efficacité d'implantations différentes (temps de calcul pour les opérations standardes). Débordement.	S§4.4,4.5,4.7 ;SW§1.3

F3 Arbres

Références

- ▷ Sedgewick sections 4.3, 5.4–5.7
- ▷ Note sur les arbres : `notes06-trees.pdf`.

Sujets

⋆ 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, infixe/dans l'ordre, postfixe/postordre, ordre de niveau.	S§5.6
⋆ Arbre syntaxique. Conversions d'expressions arithmétiques : notations infixe, postfixe et préfixe.	S§4.3
⋆ Algorithmes récursifs sur les arbres : calcul de taille, hauteur ou profondeur de sous-arbres.	S§5.7

F4 Appartenance-union

Références

- ▷ Sedgewick sections 1.2–1.3 ; Sedgewick & Wayne section 1.5
- ▷ Note sur Union-Find : `notes07-unionfind.pdf`.

Sujets

- ⋆ Problème de connexité, opérations d'appartenance-union. S§1.2
- ⋆ Structure Union-Find. Astuces : union-par-rang/union-par-taille, compression de chemin. S§1.3 ;SW§1.5
- ⋆⋆ Coût amorti d'opérations : $O(\alpha(m, n))$ pour Union-Find avec union équilibrée et compression de chemin

F5 File de priorité

Références

- ▷ Sedgewick sections 9.1–9.6 ; Sedgewick & Wayne section 2.4
- ▷ Note sur les files à priorités : `notes08-heap.pdf`.
- ▷ Note sur le tri par tas : `notes08b-heapsort.pdf`.

Sujets

- ⋆ Type abstrait de file de priorité min-tas/max-tas : opérations insert, deleteMin ou deleteMax. Implantations par tableau ou liste chaînée. S§9.1,9.5
- ⋆ Arbre en ordre de tas. Manipulation du tas : nager et couler (heapisation montante et descendante). Tas binaire, sa représentation dans un tableau. S§9.2,9.3 ;SW§2.4
- ⋆ heapify (établissement de l'ordre de tas dans un tableau) ; tri par tas, son temps de calcul et usage de mémoire. S§9.4

F6 Algorithmes sur graphes

Référence

- ▷ Sedgewick sections 3.7, 5.8 ; Sedgewick & Wayne sections 4.1, 4.3
- ▷ Note sur l'arbre couvrant minimal et le plus court chemin : `notes09-acm.pdf`.
- ▷ Note sur le parcours de graphes : `notes10-dfs.pdf`.

Sujets

- ⋆ Représentation d'un graphe : matrice d'adjacence et listes d'adjacence. S§3.7 ;SW§4.1
- ⋆ Parcours d'un graphe par profondeur. S§5.8 ;SW§4.1
- ⋆ Notion d'un arbre couvrant minimal. Principe de base des algorithmes : la règle bleue. Logique générale des algorithmes de Prim et de Kruskal, choix de structures de données. L'origin des temps de calcul $O(n \log n)$ et $O(m \log n)$. SW§4.3
- ⋆⋆ Analyse détaillé du temps de calcul des algorithmes. Avantages d'un tas d-aire ou Fibonacci dans l'algorithme de Prim.

◄ français English ►

E0 Introduction

This document defines the skills and knowledge for the mid-term examination in IFT2015, which is also the first part of the *examen pré-doctoral* in data structures.

✦ 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 the following books :

S Sedgewick, R. *Algorithms in Java*, Parts 1–4, 3^{e} édition (2003)

SW Sedgewick, R. et K. Wayne. *Algorithms*, 4^{e} édition (2011)

✶ The class notes and links to Wikipedia articles are available on the webpage `http://www.iro.umontreal.ca/~csuros/IFT2015/A13/`.

✷ No documentation is allowed at the examen.

E1 Principles of algorithm analysis

References

- ▷ Sedgewick chapter 2 ; Sedgewick & Wayne section §1.4
- ▷ Note on the foundations: `notes01-recursion.pdf`.
- ▷ Note on algorithm analysis: `notes04-analysis.pdf`.
- ▷ Note on practical aspects: `notes05-experiments.pdf`.

Topics

- ⋆ Basic principles : worst case, best case, average case. S§2.1,2.2,2.7
- ⋆ Growth of common functions : constants, logarithms, polynomials, exponentials. Factorial ($n!$), Fibonacci numbers ($F_n = F_{n-1} + F_{n-2}$), harmonic numbers ($H_n = \sum_{i=1}^{n} 1/i$), iterated logarithm S§2.3
- ⋆ Notion of amortized cost.
- ⋆⋆ Ackermann's function and its inverse
- ⋆ 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

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c > 0 \qquad \Rightarrow \qquad f(n) = O\big(g(n)\big);$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \qquad \Leftrightarrow \qquad f(n) = o\big(g(n)\big);$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1 \qquad \Leftrightarrow \qquad f(n) \sim g(n)$$

- ⋆ Using the definitions to prove $f = O(g)$ or $f = o(g)$.
- ⋆ Informal determination of space and time complexity for simple (iterative) algorithms
- ⋆ Basic recurrences. S§2.5,2.6

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

- ⋆⋆ Proof by induction for asymptotic recurrences.
- ⋆ Experimental validation of running time

E2 Elementary structures and abstract data types

References

- ▷ Sedgewick chapters 3 et 4 ; Sedgewick & Wayne sections 1.1–1.3
- ▷ Note on lists: `notes02-linkedlist.pdf`.
- ▷ Note on tables: `notes03-tableaux.pdf`.

Topics

⋆ Java building blocks.	S§3.1 ;SW§1.1
⋆ 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
⋆ Memory management for lists.	S§3.5
⋆ Concept of an abstract data type, interface, implementation, client.	S§4.1 ;SW§1.2
⋆ 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 ;SW§1.3

E3 Trees

Reference

- ▷ Sedgewick sections 4.3, 5.4–5.7
- ▷ Note on trees: `notes06-trees.pdf`.

Topics

⋆ Terminology for tree structures : k-ary tree, height, level, depth. Tree implementations.	S§5.4
⋆ Mathematical properties of binary trees (relationships between number of internal and external nodes, height)	S§5.5
⋆ Tree traversal : preorder, inorder, postorder, level-order.	S§5.6
⋆ Syntax tree. Conversion between arithmetic notations : infix, prefix and postfix.	S§4.3
⋆ Recursions on trees : computing the size, height, or depth of subtrees.	S§5.7

E4 Union-find

References

- ▷ Sedgewick sections 1.2–1.3 ; Sedgewick & Wayne section 1.5
- ▷ Note on Union-Find: `notes07-unionfind.pdf`.

Topics

- ⋆ Connectivity problems, union-find operations. S§1.2
- ⋆ Union-Find data structure. Techniques : union-by-rank/union-by-size, path compression. S§1.3 ;SW§1.5
- ⋆⋆ Amortized cost per operation : $O(\alpha(m, n))$ for Union-Find with balanced trees and path compression.

E5 Priority queues

References

- ▷ Sedgewick sections 9.1–9.6 ; Sedgewick & Wayne section 2.4
- ▷ Note on priority queues: `notes08-heap.pdf`.
- ▷ Note on heapsort: `notes08b-heapsort.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 ;SW§2.4
- ⋆ heapify (linear-time construction of heap order in a table) ; Heapsort, its running time and memory. S§9.4

E6 Graph algorithms

References

- ▷ Sedgewick sections 3.7, 5.8 ; Sedgewick & Wayne sections 4.1, 4.3
- ▷ Note on minimum spanning tree and shortest path: `notes09-acm.pdf`.
- ▷ Note on graph traversal: `notes10-dfs.pdf`.

Sujets

- ⋆ Graph representations by adjacency matrix and adjacency lists. S§3.7 ;SW§4.1
- ⋆ Depth-first search in a graph S§5.8 ;SW§4.1
- ⋆ Concept of a minimal spanning tree (MST). Basic principles of MST algorithms : the blue rule (adding minimum-weight edge in a cut). General logic of Kruskal's and Prim's algortithms, choice of data structures. Basic justification for $O(n \log n)$ and $O(m \log n)$ running times (n nodes, m edges). SW§4.3
- ⋆⋆ Detailed analysis of running time for Kruskal's and Prim's algorithms. Uses of d-ary or Fibonacci heap in Prim's algorithm.