15. IFT2015 Structures de données: Liste détaillée de sujets[1]

[1] Detailed List of Subjects for the Final Examination — English translation starts on Page 10

Introduction

LE BUT DE CE DOCUMENT est de définir les compétences et connaissances requises dans le cours IFT2015 à l'examen final. L'examen constitue également la deuxième 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://ift2015a16.wordpress.com/.
- ✷ Aucune documentation ne sera permise à l'examen final.

Principes d'analyse d'algorithmes

Références

- ▷ Sedgewick chapitre 2
- ▷ Sedgewick & Wayne §1.4[2]
- ▷ Notes sur les fondations : `notes01-recursion.pdf`.
- ▷ Notes sur l'analyse d'algorithmes : `notes04-analysis.pdf`.

[2] http://algs4.cs.princeton.edu/14analysis/

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[3] nombres Fibonacci[4], nombres harmoniques[5], logarithme itéré. S§2.3
- ⋆ Notation asymptotique[6] : 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 de la définition pour démontrer $f = O(g)$ ou $f = o(g)$.
- ⋆ Détermination du temps de calcul et d'usage de mémoire pour algorithmes (itératifs) simples, et pour algorithmes récursifs (comme expression récursive).
- ⋆ Récurrences simples. S§2.5,2.6

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

- ⋆⋆ Preuve par induction pour récurrences asymptotiques.
- ⋆ Notion de temps amorti.
- ⋆⋆ Preuves de résultats sur le coût amorti d'opérations. Principe d'analyse crédit/débit[7].
- ⋆ Validation expérimentale de temps de calcul

[3] $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$

[4] $F_0 = 0; F_1 = 1; F_n = F_{n-1} + F_{n-2}\ \{n > 1\}$

[5] $H_n = \sum_{i=1}^{n} 1/i = \ln n + \gamma + o(1)$

[6] W(fr):comparaison asymptotique

[7] W(en):accounting method

Structures élémentaires et types abstraits

Références

- ▷ Sedgewick chapitres 3 et 4
- ▷ Sedgewick & Wayne §1.1[8], §1.2[9], §1.3[10]
- ▷ Notes sur la liste chaînée : `notes02-linkedlist.pdf`.
- ▷ Notes sur les tableaux : `notes03-tableaux.pdf`.

[8] http://algs4.cs.princeton.edu/11model/
[9] http://algs4.cs.princeton.edu/12oop/
[10] http://algs4.cs.princeton.edu/13stacks/

Sujets

- ⋆ Blocs de construction pour programmes Java. S§3.1 ;SW§1.1
- ⋆ Notions de 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
- ⋆ Listes chaînées[11]. Variations : listes circulaires, doublement chaînées. Sentinelles[12] 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
- ⋆ Tableaux[13]. S§3.2
- ⋆ 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

[11] W(fr):liste chaînée
[12] W(en):sentinel
[13] W(fr):tableau

Tableaux de hachage

Références

- ▷ Sedgewick chapitre 14
- ▷ Sedgewick & Wayne §3.4[14], §3.5[15].
- ▷ Notes sur le hachage : `notes06-hashing.pdf`.

[14] http://algs4.cs.princeton.edu/34hash/
[15] http://algs4.cs.princeton.edu/35applications/

Sujets

- ⋆ Notions de base pour tableaux de hachage[16] : facteur de charge/remplissage, collisions. S§14.1
- ⋆ Fonctions de hachage : méthodes de la division et de la multiplication.
- ⋆ Résolution de collisions par chaînage séparé. Coût moyen des opérations de l'interface (table de symboles) en fonction de la facteur de charge. S§14.2
- ⋆ Addressage ouvert : notion de sondage/test. Procédures de recherche et d'insertion avec addressage ouvert. Suppression paresseuse et hachage dynamique. Sondage linéaire, grappe forte. Double hachage. S§14.3–14.6

[16] W(fr):table de hachage

Appartenance-union

Références

- ▷ Sedgewick §1.2–1.3
- ▷ Sedgewick & Wayne §1.5[17]
- ▷ Notes sur Union-Find : `notes07-unionfind.pdf`.

[17] http://algs4.cs.princeton.edu/15uf/

Sujets

- ⋆ Problème de connexité, opérations d'appartenance-union. S§1.2
- ⋆ Structure Union-Find[18]. 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 ; fonction d'Ackermann[19] et son inverse.

[18] W(fr):union-find
[19] W(fr):fonction d'Ackermann

Arbres

Références

- ▷ Sedgewick §4.3, §5.4–5.7
- ▷ Notes sur les arbres : `notes08-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

File de priorité

Références

- ▷ Sedgewick §9.1–9.6
- ▷ Sedgewick & Wayne §2.4[20]
- ▷ Notes sur la file de priorité : `notes09-heap.pdf`.

[20] http://algs4.cs.princeton.edu/24pq/

Sujets

- ⋆ Type abstrait de file de priorité[21] 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[22]. Manipulation du tas : nager et couler (heapisation montante et descendante). Tas binaire[23], 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[24], son temps de calcul et usage de mémoire. — S§9.4
- ⋆⋆ Tas d-aire[25].

[21] W(en):priority queue

[22] W(fr):tas

[23] W(fr):tas binaire

[24] W(fr):tri par tas

[25] W(en):d-ary heap

Méthodes de tri

Références

- ▷ Sedgewick §6.1–6.4, §6.6, §6.9 ; chapitres 7, 8.
- ▷ Sedgewick & Wayne §2.1[26], §2.2[27], §2.3[28].
- ▷ Notes sur les tris : `notes10-tris.pdf`.
- ▷ Notes sur le tri rapide : `notes11-quicksort.pdf`.

[26] http://algs4.cs.princeton.edu/21elementary/
[27] http://algs4.cs.princeton.edu/22mergesort/
[28] http://algs4.cs.princeton.edu/23quicksort/

Sujets

- ⋆ Terminologie : tri interne et externe. — S§6.1
- ⋆ Tri par sélection[29] et tri par insertion[30]. — S§6.3,6.4 ; SW§2.1
- ⋆ Performances des tris élémentaires (pire cas, meilleur cas, cas moyen).
- ⋆ Fusion de tableaux. — S§6.6
- ⋆ Tri par fusion[31] (descendant), sa performance. — S§8.1
- ⋆ Tri rapide[32] : algorithme de base. Améliorations : partition par la médiane-de-trois, petits sous-fichiers. — S§8.3,8.4 ; SW§2.2
- ⋆ Génération d'une permutation aléatoire
- ⋆ Performances du tri rapide (pire cas, meilleur cas, cas moyen) — S§7.1,7.4,7.5 ; SW§2.3
- ⋆⋆ Preuve de la performance moyenne $O(n \log n)$ du tri rapide. — S§7.2,7.3 ; SW 2.3
- ⋆ Preuve de la borne inférieure $\lg(n!)$ sur le nombre de comparaisons au pire pour trier — SW§2.2

[29] W(fr):tri par sélection
[30] W(fr):tri par insertion
[31] W(fr):tri fusion
[32] W(fr):tri rapide

Arbres binaires de recherche

Références

- ▷ Sedgewick chapitres 12, §13.1, §13.3, §13.4
- ▷ Sedgewick & Wayne §3.1[33], §3.2[34], §3.3[35].
- ▷ Notes sur l'arbre binaire de recherche : `notes12-abr.pdf`.
- ▷ Notes sur l'arbre rouge et noir : `notes13-rn.pdf`.

[33] http://algs4.cs.princeton.edu/31elementary/
[34] http://algs4.cs.princeton.edu/32bst/
[35] http://algs4.cs.princeton.edu/33balanced

Sujets

- ⋆ Type abstrait de la table de symboles. S§12.1,12.2
- ⋆ Recherche séquentielle et recherche binaire. S§12.3–12.5
- ⋆ Arbre binaire de recherche[36]. Procédures fondamentales sur un ABR : recherche, insertion, suppression. Recherche de minimum ou maximum, successeur ou prédecesseur. [36] W(fr):ABR S§12.6–12.9
- ⋆ Performance moyenne des opérations sur un ABR standard avec clés aléatoires. S§13.1
- ⋆ Notion d'un ABR équilibré. Maintenance d'équilibre : rotations simples et doubles.
- ⋆ ABR rouge et noir[37]. Définition par rang (hauteur noire) ou coloriage ; équivalence des deux définitions. Coût des opérations dans le pire cas. [37] W(fr):arbre bicolore S§13.4
- ⋆⋆ Hauteur maximale d'un arbre rouge et noir.
- ⋆ Techniques de base sur les ABR rouges et noirs : promotion/rétrogradation, changement de couleur, rotation. Déroulement général d'une insertion ou suppression.
- ⋆⋆ Déroulement détaillé de l'insertion et de la suppression.
- ⋆ Les arbres 2-3-4[38], et leur équivalence avec les arbres rouges et noirs. Techniques de base sur les arbres 2-3-4 : décalage et découpage, leur relation aux rotations et promotions. [38] W(en):2-3-4 tree S§13.3

Algorithmes sur graphes

Références

- ▷ Sedgewick §3.7 ; Sedgewick & Wayne §4.1[39], §4.3[40]
- ▷ Notes sur l'arbre couvrant minimal : `notes14-acm.pdf`.

[39] http://algs4.cs.princeton.edu/41graph/
[40] http://algs4.cs.princeton.edu/43mst/

Sujets

- ⋆ Représentation d'un graphe : matrice d'adjacence et listes d'adjacence[41].
- ⋆ Notion d'un arbre couvrant minimal[42]. Principe de base des algorithmes : la règle bleue. Logique générale des algorithmes de Prim[43] et de Kruskal[44], choix de structures de données.
- ⋆⋆ Analyse détaillé du temps de calcul des algorithmes. Choix de tas dans l'algorithme de Prim.

S§3.7 ;SW§4.1
[41] W(en):adjacency list
[42] W(fr):ACM
SW§4.3
[43] W(fr):algorithme de Prim
[44] W(fr):algorithme de Kruskal

Introduction

THIS DOCUMENT defines the skills and knowledge for the final examination in IFT2015, which is also the second part of the *examen pré-doctoral* in data structures.

- ✦ Topics for a «B/A-» level are denoted by ⋆; ⋆⋆ 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, 3rd edition (2003)

  SW Sedgewick, R. et K. Wayne. *Algorithms*, 4th edition (2011)

- ✶ The class notes and links to Wikipedia articles are available on the webpage http://ift2015a16.wordpress.com/.
- ✷ No documentation is allowed at the examen.

Principles of algorithm analysis

References

- ▷ Sedgewick chapter 2
- ▷ Sedgewick & Wayne §1.4[45]
- ▷ Notes on the foundations: `notes01-recursion.pdf`.
- ▷ Notes on algorithm analysis: `notes04-analysis.pdf`.

[45] http://algs4.cs.princeton.edu/14analysis/

Topics

- ⋆ Basic principles : worst case, best case, average case. §§2.1,2.2,2.7
- ⋆ Growth of common functions : constants, logarithms, polynomials, exponentials. Factorial ($n!$), Stirling's formula[46], Fibonacci numbers[47], harmonic numbers[48], iterated logarithm §§2.3
- ⋆ Asymptotic notation[49] : 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 $\lim$ §§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 $f = o(g)$.
- ⋆ Determination of space and time complexity for simple (iterative) algorithms, and for recursive algorithms (as a recursive expression).
- ⋆ Basic recurrences. §§2.5,2.6

$$\begin{aligned}
f(n) &= f(n-1) + O(1) && f(n) = O(n);\\
f(n) &= f(n-1) + O(n) && f(n) = O(n^2);\\
f(n) &= f(n/2) + O(1) && f(n) = O(\log n);\\
f(n) &= f(n/2) + O(n) && f(n) = O(n);\\
f(n) &= 2f(n/2) + O(1) && f(n) = O(n);\\
f(n) &= 2f(n/2) + O(n) && f(n) = O(n\log n);
\end{aligned}$$

- ⋆⋆ Proof by induction for asymptotic recurrences.
- ⋆ Notion of amortized cost.
- ⋆⋆ Proving amortized cost. Credit/debit method.
- ⋆ Experimental validation of running time

[46] $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$

[47] $F_n = F_{n-1} + F_{n-2}$

[48] $H_n = \sum_{i=1}^{n} 1/i = \ln n + \gamma + o(1)$

[49] W(en):big-O notation

Elementary structures and abstract data types

References

- ▷ Sedgewick chapters 3 et 4
- ▷ Sedgewick & Wayne §1.1[50], §1.2[51], §1.3[52]
- ▷ Notes on linked lists: `notes02-linkedlist.pdf`.
- ▷ Notes on tables: `notes03-tableaux.pdf`.

[50] http://algs4.cs.princeton.edu/11model/
[51] http://algs4.cs.princeton.edu/12oop/
[52] http://algs4.cs.princeton.edu/13stacks/

Topics

- ⋆ Java building blocks. S§3.1 ;SW§1.1
- ⋆ 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
- ⋆ Linked lists[53]. Variations : circular, doubly-linked lists. Sentinels[54] for head and/or tail. Manipulation of elements, insertion and deletion. List traversal. S§3.3,3.4
- ⋆ Arrays[55]. S§3.2
- ⋆ 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

[53] W(en):linked list
[54] W(en):sentinel
[55] W(en):array

Hash tables

Reference

- ▷ Sedgewick chapter 14.
- ▷ Sedgewick & Wayne §3.4[56], §3.5[57].
- ▷ Notes on hashing: `notes06-hashing.pdf`.

[56] http://algs4.cs.princeton.edu/34hash/
[57] http://algs4.cs.princeton.edu/35applications/

Topics

- ⋆ Basic notions for hashtables[58] : load factor, collisions. S§14.1
- ⋆ Hash functions : division and multiplication methods.
- ⋆ Collision resolution by separate chaining. Average-case performance with separate chaining as function of the load factor. S§14.2
- ⋆ Open addressing : probe sequence. Search and insertion with open addressing. Lazy deletion, dynamic hashing. Linear probing, primary clustering. Double hashing. S§14.3–14.6

[58] W(en):hashtable

Union-find

References

- ▷ Sedgewick §1.2–1.3
- ▷ Sedgewick & Wayne §1.5[59]
- ▷ Notes on Union-Find: `notes07-unionfind.pdf`.

[59] http://algs4.cs.princeton.edu/15uf/

Topics

- ⋆ Connectivity problems, union-find operations. S§1.2
- ⋆ Union-Find[60] 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 ; Ackermann[61] function and its inverse

[60] W(en):union-find
[61] W(en):Ackermann function

Trees

References

- ▷ Sedgewick §4.3, §5.4–5.7
- ▷ Notes on trees: `notes08-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

Priority queues

References

- ▷ Sedgewick §9.1–9.6
- ▷ Sedgewick & Wayne §2.4[62]
- ▷ Notes on priority queues: `notes09-heap.pdf`.

[62] http://algs4.cs.princeton.edu/24pq/

Topics

- ⋆ ADT for priority queue[63] : operations insert, deleteMin or deleteMax. Implementations by table or linked list. — S§9.1,9.5
- ⋆ Heap[64] order for a tree. Heap manipulation : swim and sink. Binary heap[65], its representation in a table. — S§9.2,9.3 ;SW§2.4
- ⋆ heapify (linear-time construction of heap order in a table) ; Heapsort[66], its running time and memory. — S§9.4
- ⋆⋆ d-ary heap[67].

[63] W(en):priority queue

[64] W(en):heap

[65] W(en):binary heap

[66] W(en):heapsort

[67] W(en):d-ary heap

Sorting algorithms

References

- ▷ Sedgewick §6.1–6.4, §6.6, §6.9 ; chapters 7, 8.
- ▷ Sedgewick & Wayne §2.1[68], §2.2[69], §2.3[70].
- ▷ Notes on elementary sorting algorithms: `notes10-tris.pdf`.
- ▷ Notes on quicksort: `notes11-quicksort.pdf`.

[68] http://algs4.cs.princeton.edu/21elementary/
[69] http://algs4.cs.princeton.edu/22mergesort/
[70] http://algs4.cs.princeton.edu/23quicksort/

Topics

- ⋆ Terminology : stable sort, internal and external sort. — S§6.1
- ⋆ Insertion[71] sort and selection[72] sort. — S§6.3,6.4 ; SW §2.1
- ⋆ Performance of elementary sorting algorithms (worst case, best case, average case). — S§6.6
- ⋆ Merging arrays. — S§8.1
- ⋆ Mergesort[73] (top-down), its performance. — S§8.3,8.4 ; SW§2.2
- ⋆ Quicksort[74] : basic algorithm. Improvements : pivoting by median-of-three, small subarrays. — S§7.1,7.4,7.5 ; SW§2.3
- ⋆ Performance of quicksort (worst case, best case, average case). — S§7.2,7.3
- ⋆ Generating a random permutation
- ⋆⋆ Proof of $O(n \log n)$ average running time for quicksort.
- ⋆ Proof of the lower bound $\lg(n!)$ for the worst-case number of comparisons — SW§2.2

[71] W(en):insertion sort
[72] W(en):selection sort
[73] W(en):merge sort
[74] W(en):quicksort

Binary search trees

Reference

- ▷ Sedgewick chapters 12, §13.1, §13.3 and §13.4
- ▷ Sedgewick & Wayne §3.1[75], §3.2[76], §3.3[77].
- ▷ Notes on binary search trees: `notes12-abr.pdf`.
- ▷ Notes on red-black trees: `notes13-rn.pdf`.

[75] http://algs4.cs.princeton.edu/31elementary/
[76] http://algs4.cs.princeton.edu/32bst/
[77] http://algs4.cs.princeton.edu/33balanced

Topics

- ⋆ Abstract data type of symbol table. — S§12.1,12.2
- ⋆ Sequential and binary search. — S§12.3–12.5
- ⋆ Binary search tree[78]. Basic techniques : search, insertion, deletion. Searching for minimum or maximum, successor or predecessor. — [78] W(en):BST S§12.6–12.9
- ⋆ Average performance of a standard BST with random keys. — S§13.1
- ⋆ Notion of a balanced BST. Maintaining the balance : simple and double rotations.
- ⋆ Red-black tree[79]. Definition by rank (black height) or coloring ; equivalence of the two definitions. Time complexity for operations in the worst-case. — [79] W(en):red-black tree S§13.4
- ⋆⋆ Maximum height of a red-black tree.
- ⋆ Basic techniques for red-black trees : promotion/demotion, recoloring, rotations. General outline of insertion and deletion.
- ⋆⋆ Detailed (case-by-case) steps in insertion and deletion.
- ⋆ 2-3-4 tree[80], its equivalence with the red-black tree. Basic techniques with 2-3-4 trees : shifting and splitting, relationship with promotions and rotations in red-black tree. — [80] W(en):2-3-4 tree S§13.3

Graph algorithms

References

- ▷ Sedgewick §3.7
- ▷ Sedgewick & Wayne §4.1[81], §4.3[82]
- ▷ Notes on minimum spanning tree and shortest path: `notes14-acm.pdf`.

[81] http://algs4.cs.princeton.edu/41graph/
[82] http://algs4.cs.princeton.edu/43mst/

Topics

- ⋆ Graph representations by adjacency matrix and adjacency lists[83].
- ⋆ Concept of a minimal spanning tree[84] (MST). Basic principles of MST algorithms : the blue rule (adding minimum-weight edge in a cut). General logic of Kruskal's[85] and Prim's[86] algorithms, choice of data structures. Basic justification for $O(n \log n)$ and $O(m \log n)$ running times (n nodes, m edges).
- ⋆⋆ Detailed analysis of running time for Kruskal's and Prim's algorithms. Choice of heap in Prim's algorithm.

S§3.7 ;SW§4.1
[83] W(en):adjacency list
[84] W(en):MST
SW§4.3
[85] W(en):Kruskal's algorithm
[86] W(en):Prim's algorithm