Notes of the Introduction To Algorithms

Kai Zhao

August 2, 2016

Contents

I Foundations **5**

1 The Role of Algorithms in Computing **7**

1.1 Algorithms 8

1.2 Algorithms as a technology 8

2 Getting Started **9**

3 Growth of Functions **11**

4 Divide-and-conquer **13**

5 Probabilistic Analysis and Randomized Algorithms **15**

II Sorting and Order Statistics **17**

6 Heapsort **19**

III Data Structures **21**

IV Advanced Design and Analysis Techniques **23**

V Advanced Data Structures **25**

VI Graph Algorithms **27**

7 Minimum Spanning Tree **29**

7.1 Notes . . . 30
7.2 Growing a minimum spanning tree . . . 30
7.2.1 Definition . . . 30
7.2.2 Generic-MST . . . 30
7.2.3 Theorem . . . 30

VII Selected Topics 33

VIII Appendix: Mathematical Background 35

Part I

Foundations

Chapter 1

The Role of Algorithms in Computing

1.1 Algorithms

Exercies

1.1-1 Give a real-world example that requires sorting or a real-world example that requires computing a convex hull.

Answer: One example that requires sorting is that teachers will sort our scores after the exam.

1.1-2 Other than speed, what other measures of efficiency might one use in a real-world setting ?

Answer: cost, space, manpower, material resources. In different cases, each can be the key of meausres of efficiency.
Reference: https://www.quora.com/Other-than-speed-what-other-measures-of-efficiency-might-one-use-in-a-real-world-setting

1.1-3 Select a data structure that you have seen previously, and discuss its strengths and limitations.

Answer: Array
strengths: access directly
limitations: costs lot when insert or delete

1.1-4 How are the shortest-path and traveling-salesman problems given similar? How they are different?

Answer:

1.1-5 Come up with a real-world problem in which only the best solution will do. Then come up with one in which a solution that is "approximately" the best is good enough.

Answer:

1.2 Algorithms as a technology

Chapter 2

Getting Started

Chapter 3

Growth of Functions

Chapter 4

Divide-and-conquer

Chapter 5

Probabilistic Analysis and Randomized Algorithms

Part II

Sorting and Order Statistics

Chapter 6

Heapsort

Part III

Data Structures

Part IV

Advanced Design and Analysis Techniques

Part V

Advanced Data Structures

Part VI

Graph Algorithms

Chapter 7

Minimum Spanning Tree

7.1 Notes

(i) There maybe more than one MST in a forest.

(ii) The number of all the edges in the MST is equal to $\boldsymbol{V} - 1$.

7.2 Growing a minimum spanning tree

7.2.1 Definition

A

A is a subset of some minimum spanning tree.

Safe edge

Safe edge is a edge that add to A and A is also a subset of some minimum spanning tree.

7.2.2 Generic-MST

GENERIC-MST(G, w)
1 A = ∅
2 **while** A does not form a spanning tree
3 find an edge (u, v) that is **safe edge** for A
4 $A = A \cup \{(u, v)\}$
5 return A

Initialization: After line 1, the set A trivially satisfies the loop invariant.
Maintenance: The loop in lines 2-4 maintains the invariant by adding only safe edges.
Termination: All edges added to A are in a minimum spanning tree, and so the set A returned in line 5 must be a minimum spanning tree.

7.2.3 Theorem

Let $G = (V, E)$ be a connected, undirected graph with a real-valued weight function ω defined on E. Let A be a subset of E that is inclued in some minimum spanning tree for G, let $(S, V - S)$ be any cut of G that respects A, and let (u, v) be a light edge crossing $(S, V - S)$. Then, edge (u, v) is **safe** for A. Namely, $A \cup (u, v)$ is also included in some minimum spanning tree for G.

Proof Let T be a minimum spanning tree that includes A, and assume that T does not contain the light edge (u, v), since if it does, the edge is obviously **safe** for A. We shall construct another minimum spanning tree $T^{'}$ that includes $A \cup (u, v)$ by using cut-and-paste technique, thereby showing that (u, v) is a **safe** edge for A.

The edge (u, v) forms a cycle with the edges on the simple path p from u to v in T. Since u and v are on opposite sides of the cut $(S, V - S)$, at least one edge in T lies on the simple path p and also crosses the cut. Let (x, y) be any such edge. The edge (x, y) is not in A, because the cut respects A. Since (x, y) is on the unique simple path from u to v in T, removing (x, y) breaks T into two components. Adding (u, v) reconnects them to form a new spanning tree $T^{'} = T - \{(x, y)\} \cup \{(u, v)\}$.

We next show that $T^{'}$ is a minimum spanning tree. Since (u, v) is a light edge crossing $(S, V - S)$ and (x, y) also crosses this cut, $w(u, v) \leq w(x, y)$. Therefore, $w(T^{'}) = w(T) - w(x, y) + w(u, v) \leq w(T)$.

When $w(T^{'}) == w(T)$, we know that $T^{'}$ is also a minimum spanning tree, so the edge (u, v) is **safe** for A.

When $w(T^{'}) < w(T)$, since we let T be a minimum spanning tree and **assume** that T does not contain the light edge (u, v). Therefore, the **assume** is false, so T must contain the light edge (u, v), and the edge (u, v) is **safe** for A.

7.2.4 Corollary

TODO: Prim -¿ Kruskal TODO: Kruskal -¿ Prim

Part VII

Selected Topics

Part VIII

Appendix: Mathematical Background