Introduction to Algorithms

Note

April 16, 2020

Contents

Part I

Foundations

Chapter 1

The Role of Algorithms in Computing

1. Algorithms: an *algorithm* is any well defined computational procedure that takes some value, or set of values, as *input* and produces some value, or set of values, as *output*.

2. Data Structure: a *data structure* is a way to store and organize data in order to facilitate access and modifications.

3. **NP-complete Problems:**

   - Although no efficient algorithm for an NP-complete problem has ever been found, nobody has ever proven that an efficient algorithm for one cannot exist.
   - If an efficient algorithm exists for any one problem, then efficient algorithms exist for all of them.
   - Several NP-complete problems are similar, but not identical, to problems for which we do know of efficient algorithms. Hence a small change to the problem statement can cause a big change to the efficiency of the best known algorithm.

Chapter 2

Getting Started

2.1 Insertion sort

1. **Insertion Sort:**

   - **Pseudocode:**

     INSERTION-SORT(A)
     1 **for** $j = 2$ **to** $A.length$
     2 $key = A[j]$
     3 // Insert $A[j]$ into the sorted sequence $A[1..j-1]$.
     4 $i = j - 1$
     5 **while** $i > 0$ and $A[i] > key$
     6 $A[i+1] = A[i]$
     7 $i = i - 1$
     8 $A[i+1] = key$

   - IN PLACE: it rearranges the nunbers within the array A, with at most a constant nunber of them stored outside the array at any time.
   - **Loop invariants:**
     - **Initialization:** It is true prior to the first iteration of the loop.
     - **Maintenance:** If is true before an iteration of the loop, it remains true before the next iteration.
     - **Termination:** When the loop terminates, the invariant gives us a useful property that helps show that the algorithms is correct.