Homework Assignment #4

Note

This assignment is due 2:10PM Tuesday, October 15, 2019. Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. Late submission will be penalized by 20% for each working day overdue. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. (5.3) Consider algorithm *Mapping* (see notes/slides). Is it possible that the set S will become empty at the end of the algorithm? Show an example, or prove that it cannot happen.

2. (5.7) Write a program (or modify the code discussed in class) to recover the solution (i.e., enumerate the elements in the solution) to a knapsack problem using the *belong* flag. You should make your algorithm as efficient as possible.

3. (5.18) Here is a variation of the knapsack problem: There are unlimited supplies of n items, each with an associated *value*. Design an algorithm to find how to pack the knapsack (of size K) fully, such that the items in it have the maximal value among all possible ways to pack the knapsack.

4. (5.20) Let x_1, x_2, $\ldots$, x_n be a set of integers, and let $S = \sum_{i=1}^{n} x_i$. Design an algorithm to partition the set into two subsets of equal sum, or determine that it is impossible to do so. The algorithm (presented in suitable pseudocode) should run in time $O(nS)$.

5. (5.22) In the **towers of Hanoi** puzzle, there are three pegs A, B, and C, with n (generalizing the original eight) disks of different sizes stacked in decreasing order on peg A. The objective is to transfer all the disks on peg A to peg B, moving one disk at a time (from one peg to one of the other two) and never having a larger disk stacked upon a smaller one.

   (a) Give an algorithm to solve the puzzle. Explain how induction works here.

   (b) Compute the total number of moves in the algorithm. Show the details of your calculation.