Advanced Algorithms

k-best 0-1 Knapsack Problem

Petar Veličković

Easter Term 2016

1. Consider the *0-1 Knapsack* problem as described in the Algorithms course:

   You are given N items, and for each item an integer value V_i and integer weight W_i is known. You also have a knapsack of integer capacity C. You want to choose which items to place in the knapsack such as to maximise the overall value, while not exceeding its capacity in combined weight.

   (a) As a warmup, state the dynamic programming (DP) solution for this problem, with a brief intuition behind the stated recurrence relations. What are the time and space complexities of the algorithm?

   (b) Now consider a modified variant known as the "k-best 0-1 Knapsack", where we're no longer interested in just the single best way to fill up the knapsack, but the best k ways! Modify your algorithm from part (a) to accommodate for this change, and state its time and space complexities.

   (c) Consider that the problem now has the following constraints:

   - $N = 2000$
   - $C = 10000000000$
   - $K = 40$
   - values of V_i and W_i are generated uniformly at random and IID from the range $[0, C]$.

   Is the approach from (b) still feasible, and why? If not, suggest an approach which could feasibly solve the problem in the average case. State also its worst-case time complexity. [Hint: branch&bound]