Smoothed analysis for the knapsack problem

Uriel Feige

July 19, 2021

1 Introduction

Part four of the BWCA book contains three chapters on smoothed analysis. The topic of this lecture is based on Chapter 15 (a part in that chapter that is based in [1]), but is simplified so as to convey the main ideas, while presenting a less general result.

2 The knapsack problem

In the knapsack problem, there is a set U of n items, where every item i has positive weight w_i and positive profit p_i. In addition there is a knapsack of capacity W. The goal is to select a set S of items maximizing the profit $\sum_{i \in S} p_i$, subject to not violating the capacity constraint $\sum_{i \in S} w_i \leq W$.

The problem is NP-hard, but has weakly polynomial time algorithms. Namely, if all weights are integers, it can be solved by dynamic programming in time polynomial in (n, W) (but NP-hard for $(n, \log W)$). For example, one can inductively fill up an n by W table T, where entry $T_{i,w}$ specifies the most profitable solution of weight w, among those that include items from $[1, \ldots, i]$. Likewise, if all profits are integers, it can be solved in time polynomial in $(n, \max_i p_i)$ (but NP hard for $(n, \log \max_i p_i)$).

There are also fully polynomial time approximation schemes (FPTAS) for knapsack. The following lemma is based on one of them.

Lemma 1 *Let $W^* \leq W$ denote the weight of an optimal solution to the knapsack problem and let P^* denote its profit. Suppose that for a given input instance it is known that for some explicit $\epsilon > 0$, no solution of weight in the range $(W^*, W^* + \epsilon W]$ has profit at least P^*. Then the input instance can be solved in time polynomial in $(n, \frac{1}{\epsilon})$.*

Proof: We shall run two algorithms, at least one of which will produce the optimal solution.

The first algorithm assumes that $W^* > (1 - \frac{\epsilon}{2})W$. Modify the weights as follows. Let $k = \frac{\epsilon W}{2n}$. Round down each weight w_i to the nearest multiple of k (consequently, also W can be rounded down to the nearest multiple of k), giving weight w'_i (and W'). Observe that with the new weights, the original optimal solution remains both feasible and optimal. Now the dynamic programming table has size $n \cdot \lfloor \frac{W}{k} \rfloor = O(\frac{n^2}{\epsilon})$, and the problem instance can be solved in time polynomial in $(n, \frac{1}{\epsilon})$.

The second algorithm assumes that $W^* \leq (1 - \frac{\epsilon}{2})W$. In this case each weight w_i is rounded up (rather than down) to the nearest multiple of k (again, W can be rounded down to the nearest multiple of k), and the analysis proceeds as for the first algorithm. □

3 Smoothed analysis

Smoothed analysis is a framework for illustrating that certain classes of NP-hard problems with numerical data parameters are likely not to be difficult in practice. The framework applies to situations in which there is some randomness in the process of generating the input data. Hence input is being modelled as generated in two phases. In the first phase, an adversary generates an arbitrary input instance. In the second stage, the numerical parameters are perturbed at random. The goal is to design algorithms that with high probability solve the perturbed instances optimally, where the probability is taken only over the perturbation.

Another possible application for the smoothed analysis framework is as follows. Given an arbitrary instance, first smooth it (here it is assumed that the smoothing operation can be done in random polynomial time), and then solve the smoothed instance. If the optimal solution (or its value) enjoys some Lipschitz continuity with respect to the smoothing operation, and the smoothing operation is "small", then the solution to the smoothed instance may provide a good approximation to the solution for the original instance.

4 Smoothed analysis of knapsack

Consider a smoothed version of the knapsack problem. The adversary first selects an arbitrary instance. Thereafter, weights of items are perturbed at random. For simplicity and concreteness, we consider the following perturbation, parametrized by $\delta > 0$. For each weight w_i, we select uniformly at random r_i in the range $[0, \delta W]$, and the perturbed weight becomes $\hat{w}_i = w_i + r_i$. (The perturbation is not symmetric around 0, so that we never end up with an item of negative weight.)

Assume some consistent tie breaking rule among solutions (e.g., lexicographic). We assume that this rule breaks ties in favour of lower weight solutions. Let G denote the optimal solution for the smoothed instance, let P^* denote its profit, and let W^* denote its weight. Note that $W^* \leq W$. The following lemma is a variation on the *isolating lemma* of [2].

Lemma 2 *With probability at least* $1 - \frac{\epsilon n}{\delta}$*, there is no solution of weight in the range* $(W^*, W^* + \epsilon W]$ *that has profit at least* P^**.*

The lemma may sound surprising, as the number of solutions of weight in a range of width ϵW may well be exponential in n.

Proof: If $\sum_i \hat{w}_i \leq W^*$, then there is no set of weight above W^*, and there is nothing to prove. Hence we assume that $\sum_i \hat{w}_i > W^*$. Consequently, G misses at least one item.

For $1 \leq i \leq n$, let U_i denote the collection of all sets of items that do not include item i. Let P_i denote the maximum profit of a feasible solution (of weight at most W) within U_i, and let G_i denote the corresponding solution. Let W_i denote the minimum weight of a

feasible solution within U_i among those that have profit at least $P_i - p_i$, and let B_i denote the corresponding solution. (It might happen that $B_i = G_i$.) Observe that for every $i \notin G$ it holds that $G = G_i$. Moreover, in this case $W_i + \hat{w}_i \geq W^*$, as otherwise G would not be the optimal solution (recall the tie breaking rule favouring solutions of lower weight).

Suppose that the lemma fails. Then there is a set of weight in the range $(W^*, W^* + \epsilon W]$ that has profit at least P^*. Among these sets, consider the set B of smallest weight. There must be an item i in B but not in G. Consequently, $G_i = G$ for this i. Likewise, it can be seen that $B_i = B \setminus \{i\}$. Hence for the lemma to fail, there must be a bad event E_i of the form $W^* < W_i + \hat{w}_i \leq W^* + \epsilon W$. Fixing all weights and perturbations except for r_i, we see that $Pr[E_i] \leq \frac{\epsilon W}{\delta W}$. The lemma follows from a union bound over E_i for $1 \leq i \leq n$. □

Corollary 3 *In the smoothed model for knapsack described above, the optimal solution can be found with high probability in time polynomial in n and in $\frac{1}{\delta}$.*

Proof: Choose ϵ to be much smaller than $\frac{\delta}{n}$. By Lemma 2, there is high probability that there is no solution of weight in the range $(W^*, W^* + \epsilon W]$ that has profit at least P^*. If this event holds, Lemma 1 provides an algorithm with the desired time complexity. □

References

[1] Rene Beier, Berthold Vocking: Random knapsack in expected polynomial time. J. Comput. Syst. Sci. 69(3): 306–329 (2004).

[2] Ketan Mulmuley, Umesh V. Vazirani, Vijay V. Vazirani: Matching is as easy as matrix inversion. Comb. 7(1): 105–113 (1987).