CSE 347 Analysis of Algorithms April 17, 2019

Lecture 23 Summary

Guest lecturer: Zihao Deng

1. We can convert a pseudopolynomial time algorithm for KNAPSACK to a Polynomial Time Approximation Scheme (PTAS) by *rounding* the values.
   (a) The idea: since the only problem with our pseudopolynomial time algorithm is the bad dependence on the size of the numbers, if we take a coarser "unit" we get smaller numbers and hence a fast, approximate algorithm.
   (b) Small technical issue: our original KNAPSACK algorithm had a bad dependence on the weight; rounding this gives approximate feasibility rather than an approximately optimal feasible solution. So we'll give a different algorithm at the end that is pseudopolynomial with respect to the largest item's value instead.
   (c) We assume that the most valuable item fits in the knapsack (otherwise, toss out any items that can't possibly fit). If this item has value V, we use "units" of $\frac{\epsilon}{2n}V$ where there are n items. Notice that any solution we obtain must be at least as good as this one, i.e., must have value at least V.
   (d) Now, in this new unit, the maximum value is at most $2n/\epsilon$, so our dynamic programming algorithm will be polynomial time.
   (e) Since there are at most n items in the knapsack, if we total the rounding error over these items, the total approximation error of the value of the knapsack is indeed at most an ϵ-fraction of the value. So we indeed have a $(1-\epsilon)$-approximation to the optimal value.
2. We can use a slightly different dynamic programming algorithm to get an algorithm for KNAPSACK that is pseudopolynomial with respect to the values instead.
   (a) The pseudopolynomial dependence is due to the choice of subproblems. So instead of asking for the best value attainable in the remaining capacity (giving a pseudopolynomial dependence on the capacity), we ask for the smallest weight that achieves a given value, which is then pseudopolynomial in the value.
   (b) Given the table of solutions, then, we can then search for the largest value for which the weight is below our given bound.
   (c) Note that the largest target value is nV, so this is indeed pseudopolynomial in the maximum value V.