

Post-Homework Practice Problems

Below is a set of additional practice problems on approximation algorithms, including linear programming approaches. You can find solutions to these problems on the course web site. We're also willing to listen to you describe your solutions or to look at your writeups during office hours.

Practice Problems

1. Consider the well-known *bin-packing* problem: given a collection of n items, with item i having size ℓ_i , pack the items into bins of fixed size m , such that the total number of bins used is minimized.

There are good approximations for bin packing (considerably better than a factor of 2). However, one cannot do arbitrarily better. Show that, if we could find a polynomial-time algorithm A for bin packing that achieved an approximation ratio of $3/2 - \epsilon$, for any $\epsilon > 0$, then $P = NP$.

(*Hint*: reduce from PARTITION.)

2. One of the best practical approximations known for bin-packing is the "first-fit decreasing" (FFD) algorithm. In 1973, Johnson showed that, if B^* is the optimal number of bins for a bin-packing problem, the FFD algorithm's solution uses at most $11/9B^* + 4$ bins. Clearly, $11/9 < 3/2$, so why doesn't this result conflict with the negative result of the previous problem?
3. Consider the following generic integer program P . Let $X = \{x_1 \dots x_n\}$ be variables whose values may be either 0 or 1. The program P is: minimize $\sum_i c_i \cdot x_i$ subject to m constraints $\sum_i a_{ij}x_i \geq b_j$, where the a 's, b 's and c 's are all constants.

Suppose one could find an optimal solution \bar{X} to the LP relaxation of P that sets each \bar{x}_i to a rational number $\frac{p_i}{k}$, where $p_i \in \{0, 1, 2, \dots, k\}$. (For example, if $k = 2$, then each \bar{x}_i is 0, 1, or $\frac{1}{2}$). Show how to turn the solution \bar{X} into a k -approximation for the optimal integer-valued solution to P .

(Interestingly, it turns out that the IP we defined for weighted vertex cover always has the kind of solution described above with $k = 2$. This *half-integral* solution is the one returned by the simplex algorithm for linear programming. LPs with half-integral optima appear in other problems as well, and they lead to easy 2-approximations by the method you are asked to derive here.)

4. Consider the LP algorithm shown in class for weighted parallel machine scheduling. The algorithm shown there gave a 2-approximation guarantee for the case of $m = 2$ machines. Suppose now that we define the analogous IP for $m > 2$ machines and solve its LP relaxation.

(a) At most how many jobs will be split by the LP optimum?

- (b) Assume that m is a *constant* with respect to our problem specification. Generalize the two-machine LP algorithm into a polynomial-time algorithm that achieves an approximation ratio of 2 for any number of machines, given this assumption. Give a bound on the running time of your algorithm, exclusive of the time spent computing the LP optimum.