CS302 - Problem Set 6

Due: Monday, Oct 30. Must be uploaded to Canvas before the beginning of class.

1. (**) **[6 points]** Suppose I would like to give you more flexibility on your exams, so I give you some large number (let's call it M) of problems, where the ith problem is worth P_i points. The time I give you to take the exam is not sufficient to solve all of the problems, so each student might solve a different subset of problems (and of course might get different grades on each problem). I would like to give a good grade to a student who does sufficiently well on a sufficient number of questions, and give a worse grade to a student who just gets a few points correct on a lot of problems. What is a (relatively) fair way I could use the knapsack problem to figure out grades?

2. You run a plant that produces sheets of alumnium alloy, and then you cut them to size for customers. Your machine produces sheets of dimension $A \times B$, and you can cut any sheet into two smaller sheets by making a vertical or horizontal cut. You can sell a piece of dimensions $x_i \times y_i$ for amount v_i for $i \in \{1, 2, \ldots, n\}$. (You can sell multiple copies of the ith product if you can produce multiple pieces of that size. Also assume the alloy can be rotated 90 degrees to create a product of the appropriate size.) Assume A, B, x_i, y_i, and v_i for $i \in \{1, 2, \ldots, n\}$ are positive integers. In this problem, you will be designing an algorithm that figures out what your maximum profit is. (Hint - you may not want to solve the parts of problems in the order given.)

   (a) (**) **[9 points]** Please provide psuedocode for a dynamic programming algorithm that outputs your maximum profit.

   (b) (**) **[11 points]** Prove your algorithm is correct.

   (c) (*) **[6 points]** What is the runtime of your algorithm?

   (d) (**) **[6 points]** Explain how you would modify your algorithm to tell you whether you should divide the current sheet, and if so, where you should make the cut.

   (e) (***) **[6 points]** The most straightforward algorithm (the one you probably came up with), does not have optimal time complexity. Describe how you could modify your algorithm to run in $O(\max\{n, AB \times \max\{A, B\}\})$ time.

3. How long did you spend on this problem set?