CS180 Homework 4

Due: December 6, 11:59pm (No late submission is allowed)

1. (25 pt) You're helping to run a high-performance computing system capable of processing several terabytes of data per day. For each of n days, you're presented with a quantity of data; on day i, you're presented with x_i terabytes. For each terabyte you process, you receive a fixed revenue, but any unprocessed data becomes unavailable at the end of the day (i.e., you can't work on it in any future day).

   You can't always process everything each day because you're constrained by the capabilities of your computing system, which can only process a fixed number of terabytes in a given day. In fact, it's running some one-of-a-kind software that, while very sophisticated, is not totally reliable, and so the amount of data you can process goes down with each day that passes since the most recent reboot of the system. On the first day after a reboot, you can process s_1 terabytes, on the second day after a reboot, you can process s_2 terabytes, and so on, up to s_n; we assume $s_1 > s_2 > s_3 > ... > s_n > 0$. (Of course, on day i you can only process up to x_i terabytes, regardless of how fast your system is.) To get the system back to peak performance, you can choose to reboot it; but on any day you choose to reboot the system, you can't process any data at all.

   The problem. Given the amounts of available data $x_1, x_2, ..., x_n$ for the next n days, and given the profile of your system as expressed by $s_1, s_2, ..., s_n$ (and starting from a freshly rebooted system on day 1), choose the days on which you're going to reboot so as to maximize the total amount of data you process.

   Example. Suppose n = 4, and the values of x_i and s_i are given by the following table.

	day1	day2	day3	day4
x	10	1	7	7
s	8	4	2	1

   The best solution would be to reboot on day 2 only; this way, you process 8 terabytes on day 1, then 0 on day 2, then 7 on day 3, then 4 on day 4, for a total of 19. (Note that if you didn't reboot at all, you'd process $8 + 1 + 2 + 1 = 12$; and other rebooting strategies give you less than 19 as well.)

   (a) Give an example of an instance with the following properties.
      - There is a "surplus" of data in the sense that $x_i > s_1$ for every i.
      - The optimal solution reboots the system at least twice.

      In addition to the example, you should say what the optimal solution is. You do not need to provide a proof that it is optimal.

   (b) Give an efficient algorithm that takes values for $x_1, x_2, ..., x_n$ and $s_1, s_2, ..., s_n$ and returns the total number of terabytes processed by an optimal solution.

2. (25 pt) A palindrome is a string that reads the same from left to right and from right to left. Design an algorithm to find the minimum number of characters required to make a given string to a palindrome if you are allowed to insert characters at any position of the string. For example, for the input "aab" the output should 1 (we'll add a 'b' in the beginning so it becomes "baab").

   The algorithm should run in $O(n^2)$ time if the input string has length n.

3. (30 pt) Given a large $W \times L$ rectangle, we want to cut it into smaller rectangles of specific shapes $(a_1 \times b_1), (a_2 \times b_2), \ldots, (a_K \times b_K)$. Note that all these numbers, including $W, L, a_1, \ldots, a_K, b_1, \ldots, b_k$ are integers, and each time we can only make a full horizontal or vertical cut on a rectangle at an integer point to split it into two. In the end we will get a collection of small rectangles, hopefully most of them have the shape matching one of the $a_i \times b_i$, but there could be pieces that don't match with any pre-specified shapes and those areas are wasted. For simplicity we assume the rectangles cannot be rotated (so $a_i \times b_i$ is different from $b_i \times a_i$). We don't care about how many of these smaller rectangles we get in the end, but our goal is to minimize the total wasted area. Design an algorithm that runs in polynomial time of k, W, L that computes the minimum possible wasted area.

For example, assume $W = 21, L = 11$ and the desired rectangles are $(10\times4), (9\times8), (6\times2), (7\times5), (15\times10)$. The minimum possible wasted area is 10 (the gray area), as shown in Figure 1.

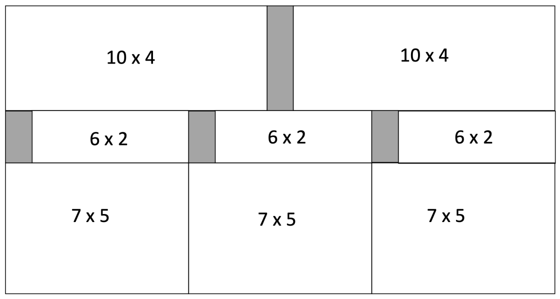

Figure 1:

4. (20 pt) Consider the set $A = \{a_1, \ldots, a_n\}$ and a collection $B_1, B_2, \ldots, B_m$ of subsets of A (i.e., $B_i \subseteq A$ for each i). We say that a set $H \subseteq A$ is a hitting set for the collection $B_1, B_2, \ldots, B_m$ if H contains at least one element from each B_i—that is, if $H \cap B_i$ is not empty for each i (so H "hits" all the sets B_i).

   We now define the Hitting Set Problem as follows. We are given a set $A = \{a_1, \ldots, a_n\}$, a collection $B_1, B_2, \ldots, B_m$ of subsets of A, and a number k. We are asked: Is there a hitting set $H \subseteq A$ for $B_1, \ldots, B_m$ so that the size of H is at most k?

   Prove that the vertex cover problem $\leq_p$ the hitting set problem.

★ Homework assignments are due on the exact time indicated. Please submit your homework using the Gradescope system. Email attachments or other electronic delivery methods are not acceptable. To learn how to use Gradescope, you can:

- 1. Watch the one-minute video with complete instructions from here: `https://www.youtube.com/watch?v=-wemznvGPfg`
- 2. Follow the instructions to generate a PDF scan of the assignments: `http://gradescope-static-assets.s3-us-west-2.amazonaws.com/help/submitting_hw_guide.pdf`
- 3. **Make sure you start each problem on a new page.**

★ We recommend to use LaTeX, LyX or other word processing software for submitting the homework. This is not a requirement but it helps us to grade the homework and give feedback. For grading, we will take into account both the correctness and the clarity. Your answer are supposed to be in a simple and understandable manner. Sloppy answers are expected to receiver fewer points.

★ Unless specified, you should justify the correctness and time complexity of your algorithm.

★ **Your answer to each problem should be written in 2 (A4 or letter sized) pages.** If you are typing for the homework, a font size of 12 points or larger must be used. If you are writing, the font size should not be too small to affect the readability. You will get 5 points penalty for each additional page used. For all these problems, the correct solution can be easily written in 1 page.