CS 5800 Algorithms	V. Pavlu, K. Lieberherr
Virgil Pavlu	Due: Jan. 27, 2014

Homework 3

1 Submission Rules

Point 7 has been updated.

`http://www.ccs.neu.edu/home/lieber/courses/algorithms/cs5800/sp14/homeworks/submission-rules.pdf`

2 Problems

1. (40p) Problem 4-1 from (a) to (d) (page 107)
2. (30p) Problem 4-3 from (a) to (f) (page 108)
3. (25p) **Task**: When multiplying n numbers $P = a_1 * a_2 * ... * a_n$, there are many choices of multiplication order, *or parenthesizing*, for example $(a_1 * a_2) * (a_3 * a_4)$, $((a_1 * a_2) * a_3) * a_4$, $(a_1 * (a_2 * (a_3 * a_4))$, etc. We want to know in how many different ways can we put the parentheses.

   Solution : divide and conquer; lets say T(n) is the number of ways to put parentheses on the product of n numbers. First we decide the last multiplication, say its between numbers a_k and a_{k+1}. Thus we decided $P = (a_1 * a_2 * ... * a_k) * (a_{k+1} * a_{k+2} * ... * a_n)$ There are $n-1$ possible such k, each leading to different parenthesis structure no matter how left and right sides get their parentheses.

   For each k, we have now two subproblems. We shall use the "earlier results" on subproblems: count the number of ways to parenthesize the k first half of numbers, that is T(k), and the number of ways to parenthesize the $n-k$ second half of numbers, that is T($n-k$). Then we have to combine these, and iterate across all k, to get back T(n).

   - What is the recurrence for T(n) embedded in this solution? From hw 2.
   - Solve this recurrence for an asymptotic lower bound (prove it is at least ...)
4. (40p) Perfect Avatar.

   The forced falsifier (Avatar Alice) in

   `http://www.ccs.neu.edu/home/lieber/courses/algorithms/cs5800/sp14/homeworks/m3/calculus-debate-imperfect-falsifier.xlsx`

is not perfect. Turn it into a perfect forced falsifier. A perfect forced falsifier detects any weakness in the verifier and if the verifier is not perfect (or very close to perfect), the verifier will lose. If the verifier is perfect, the forced falsifier will lose.

Note the two tables in the spreadsheet for translating wins and losses into non-forced-loss points. The fewer the better.

As suggested by the submission guidelines (point 7), when you make a claim and you have a winning strategy (a proof), you should check it in debates in your group. This has several advantages: (1) you engage the skills of your team members to find faults in your reasoning. The earlier we can find those faults, the better. (2) You get very concrete feedback through the debates why your solution does not work. (3) You have not only the opportunity to learn from others, you also have the opportunity to teach them!

For this problem, let's make this explicit: Do your debates on Piazza as you did for hw 1: 3 debates with random forcing. Each student is debater in 2 debates and admin in 1. Let's have a battle of the avatars.

As I was pointing out on Piazza, the imperfect avatar that I gave you is imperfect in several ways: it not only uses suboptimal algorithms. It also has a piece missing: When Bob gives the perfect th, he has not won yet. He must also find a "winning" y for any x that Alice gives him.

5. (15 pts) Problem 6-2.

6. (5 pts) Exercise 7.2-1.

7. (10 pts) Exercise 8.1-3.