CMSC451 (Kruskal) Homework 4 Due: Monday, May 4, 2020

Problem 1. Use the dynamic programming algorithm to find by hand an optimal parenthesization for multiplying matrices of dimensions are given by the sequence

$$< 6, 3, 10, 5, 8, 4, 20 > ,$$

so that the matrix dimensions are

$$6 \times 3,\ 3 \times 10,\ 10 \times 5,\ 5 \times 8,\ 8 \times 4,\ 4 \times 20 > \ .$$

Show the table. You may use a calculater.

Problem 2. In the traditional world chess championship, which changed format after Bobby Fischer became world champion, a match is 24 games. The current champion retains the title in case of a tie. Not only are there wins and losses, but some games end in a draw (tie). Wins count as 1, losses as 0, and draws as 1/2. The players take turns playing white and black. White moves first, which is an advantage. Assume the champion is white in the first game, has probabilities w_{w}, w_{d}, and w_{l} of winning, drawing, and losing playing white, and has probabilities b_{w}, b_{d}, and b_{l} of winning, drawing, and losing playing black.

(a) Write down a recurrence for the chance that the champion retains the title. Assume that there are g games left to play in the match and that the champion needs to win i games (where i is either an integer or an integer plus 1/2).

(b) Write a recursive algorithm to compute the chance that the champion retains the title, using the above recurrence.

(c) Produce a *memoized* version of your algorithm.

(d) Give a bottom-up dynamic programming algorithm to compute the chance that the champion retains the title.

(e) Analyze the running time of your dynamic programming algorithm.

Problem 3. In the *Euclidean Traveling-Salesman Tour* the cities are points in the Euclean plane and distances are measured in the standard way. The problem is NP-complete. A *Bitonic Euclidean Traveling-Salesman Tour* starts at the leftmost city, visits cities from left-to-right until it gets to the rightmost city, and then visits cities from right-to-left until it gets back to the leftmost city. (Of course, each city is visited only once, either going left-to-right or right-to-left.) Use dynamic programming to find an optimal bitonic tour in time $\theta(n^2)$. Make sure to state your recurrence. HINT: Scan left-to-right keeping track of the optimal left-to-right tour and the optimal right-to-left tour at the same time.