COSC 302: Analysis of Algorithms — Spring 2018
Prof. Darren Strash
Colgate University

Problem Set 9 — Dynamic Programming II, All-Pairs Shortest Paths
Due by 4:30pm Friday, April 27, 2018 as a single pdf via Moodle (either generated via LaTeX, or concatenated photos of your work). Late assignments are not accepted.

This is an *individual* assignment: collaboration (such as discussing problems and brainstorming ideas for solving them) on this assignment is highly encouraged, but the work you submit must be your own. Give information only as a tutor would: ask questions so that your classmate is able to figure out the answer for themselves. It is unacceptable to share any artifacts, such as code and/or write-ups for this assignment. If you work with someone in close collaboration, you must mention your collaborator on your assignment.

Suggested practice problems, from CLRS: 25.1-8; 25.1-9; 25.1-10; 25.2-6; 25.2-8 25.3-2; 25.3-4; 25.3-6

1. (10 points) Problem 15-9 from CLRS. Be sure to follow the steps for developing a dynamic programming algorithm. That is,

   (a) Describe how to divide the problem into subproblems. Explicitly define your subproblem, and the value you are optimizing on the subproblem. The sub problems are the two substring that are created when you choose a split. We will optimaize the cost of each cut and thus need the minimum cost. We do so by trying each cut between the start index and end index of a substring. We will do a recursive call for each possible cut.

   (b) Give a recurrence for the optimization problem.

$$L_{e,s} = \begin{cases} \infty, & \text{if } s < 0. \\ \infty, & \text{if } e < 0. \\ 0, & \text{if } k < 0. \\ min_{s<k<e}(e-s) + L_{s,k} + L_{k,e}, & \text{otherwise.} \end{cases} \tag{1}$$

   We reduce from Hamiltonian Path problem. Given a graph $G = (V, E)$, we add two vertices s and t that connects all v in V, respectively. Also s and t is connected. Then we get a graph $G' = (V', E')$, where $V' = V \cup \{s, t\}$ and $E' = E \cup \{(s,t)\} \cup \{(s,v),(t,v) \mid v \in V\}$. We assign a weight $|E| + 2|V|$ to (s,t), and 1 to all other edges. An illustration is given as follows:

   (c) Show that the problem has optimal substructure, that there is subproblem overlap, and that subproblems are independent. Optimal Substructure: I shall prove this problem has optimal substructure by contradiction. Assume that a solution does not need optimal solutions to the subproblems in order to be correct. Let k_OPT be a part of an optimal solution. Let k_L be our part of the solution which is not optimal. The cut K_OPT leads to a sum of subproblems called OPT_sum and k_L lead to a sum of subproblems called L_sum note by definition of optimal $OPT_sum < L_sum$. Which means, $OPT < L$ since $OPT = OPT_sum + (e-s)$ and $L = L_sum + (e-s)$. But we assumed L does not need

optimal sub problems whichi leads to a contradiction.
SubProblem Overlap: The problem has overlapping sub problems because each problem depends on two other sub problems. $L_{s,k}, L_{k,e}$. Which both have a value $L_{s,e}$.
SubProblem Independance: Each sub problem starts s to k and k to c which or are completely disjoint since the only cuts will be between but not including the numbers above. The problems are completely seperate.

(d) Show how to store subproblem solutions in an arrayDraw the array schematically, draw the range of indices, show where the final optimal value is stored, and show dependencies by drawing arrows and highlighting subproblems.

(e) Give pseudocode to fill in the array and recover the optimal breaks.

2. Problem 25.2-4 from CLRS.

3. Problem 25.2-5 from CLRS.