HW 6

Due on **Wednesday, October 16, 2024 at 9am CST**

CS/ECE 374A: Intro. Algorithms & Models of Computation, Fall 2024 Version: **1.3**

Submission instructions as in previous homeworks.

Instructions for dynamic programming. In any dynamic programming solution, you should follow the steps below (in this order):

(I) Provide a clear and precise definition of the subproblems (i.e., what the recursive function is computing).

(II) Derive a recursive formula to solve the subproblems (including base cases), with justification or proof of correctness if the formula is not obvious.

(III) Specify a valid evaluation order.

(IV) Give pseudocode to evaluate your recursive formula bottom-up (with loops instead of recursion).

(V) Analyze the running time and space used.

(VI) Describe the associated configuration graph, and explain what problem needs to be solved on the configuration graph, if one wants to solve the dynamic program directly on the graph.

Do not jump to pseudocode immediately. Never ever skip step (I)!

---

11 (100 PTS.) DFAs are back!

You are given as input a DFA M, an input string $w = w_1 w_2 \dots w_m$, and a parameter $k > 0$. Let $\Sigma = \{0, 1\}$, and let $M = (Q, \Sigma, \delta, s, A)$ be a given, where $n = |Q|$, and $Q = \{1, \dots, n\}$. In particular, δ is provided as a lookup table, and can be computed per specific state, and input character in constant time (i.e., the input size here is $O(n + m)$).

Describe a dynamic-programming algorithm that decides if there are exactly k characters in w, such that if we delete them from w, the resulting string w' is in $L(M)$, and furthermore, there are exactly k distinct prefixes of w' (including itself) that are in $L(M)$.

What is the running time of your algorithm? For credit, the running time of your algorithm should be polynomial in all parameters (i.e., n, k, m).

12 (100 PTS.) Closest subpolygon

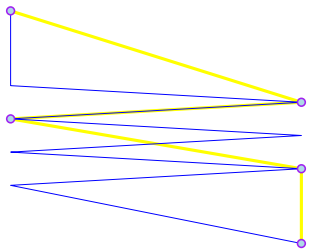

Figure 1: A polygon (blue), and a possible subpolygon (yellow). Note that a subpolygon does not have to include the start and end vertices of the original polygon. The marked vertices are the subset of the vertices used to form the subpolygon.

Given two polygonal curves in the plane $P = p_1, \dots, p_n$ and $Q = q_1, \dots, q_n$ (i.e., P is the polygonal line formed by the segment $p_1p_2, p_2p_3, \dots$), their ***matching distance*** is $d(P, Q) = \sum_{i=1}^{n} ||p_i - q_i||^2$.

Given a polygon Q, its subpolygon is a polygon formed by picking a subset of the vertices and connecting them in the same order as the original polygon, see Figure 1.

Given two such polygons P and Q with n and m vertices, respectively, with $n \leq m$, consider the problem of computing a subpolygon Q' (i.e., a polygon formed by connecting a subset of the vertices of the original polygon in the same order) of Q with n vertices, such that Q' minimizes its matching distance from P.

12.A. (70 PTS.) Describe an algorithm, as fast as possible, using dynamic programming, that computes the minimum distance between P and any subpolygon of Q with n vertices – see Figure 2 for an example. Bound the running time of your algorithm as a function of m and n.

Figure 2: An input, on the left, and a possible solution, on the right.

12.B. (30 PTS.) Describe how to modify your algorithm in (A) so that it computes and outputs the optimal subpolygon Q'.