Assignment 2
CSC373 Winter 2010
Due Date: March 4th, Noon, BA2220 Drop Box 8

For all of the problems below, you will only get full credit for algorithms that are efficient – i.e. optimal, or close to optimal. For all of the algorithms below, you are expected to:

- Give an efficient algorithm.
- Argue (briefly) that your algorithm is correct.
- Give a simple analysis of its running time.

For any algorithms that we learned in class, or are discussed in the DPV book you can just cite the algorithm without explaining running time or correctness.

Problem 1

Consider the String Generation problem defined as follows. Input: A list of "generator" strings $\{s_1, s_2, \ldots, s_k\}$ and a "target" string t over some fixed alphabet Σ.

Output: A list of indices $i_1, i_2, \ldots, i_r$ such that $t = s_{i_1} \cdot s_{i_2} \cdot \cdots \cdot s_{i_r}$, if such a list exists; the special value ∅ otherwise. For example, for input s_1 = bab, s_2 = aba, s_3 = babb, s_4 = a, t = babbaba the output could be either $i_1 = 1, i_2 = 1, i_3 = 4$ or $i_1 = 3, i_2 = 2$ because t can be written as $t = s_1 \cdot s_1 \cdot s_4$ but also as $t = s_3 \cdot s_2$; for input s_1 = bab, s_2 = aba, s_3 = babb, s_4 = a, t = aab the output would be ∅ because t cannot be written as a combination of the s's. By convention, we say that t = ∅ (the empty string) can be written as a combination of 0 generator strings.

Design a Dynamic Programming algorithm to solve the String Generation problem

In your answer, please use the following notation:

- |s| represents the length of string s (by convention, |∅| = 0)
- t_i represents the i^{th} symbol of t and $t_{i...j}$ represents the substring of t from the i^{th} symbol to the j^{th} symbol, inclusively (indices start at 1, i.e., $t = t_{1...|t|}$)

Problem 2

DPV 4.21 Hint: You may want to recall logarithms from high school.

Problem 3

DPV 6.9

Problem 4

Consider the all-pairs shortest path algorithm described on pp. 172-173 of DPV. The algorithm, as given requires $O(|V|^3)$ space and memory, and only saves the length of the shortest path between two nodes, not the path itself. In class we discussed how to reduce the memory requirement to $O(|V|^2)$ for computing the length, but again, not storing the actual paths.

Design an $O(|V|^3)$ time and $O(|V|^2)$ memory algorithm that can reconstruct the actual paths. More precisely, you algorithm should build a table, at most size $O(|V|^2)$, from which it should be possible, in linear time, to reconstruct the actual shortest path between any two nodes (not just the length of this path).

Problem 5

DPV 7.17

Problem 6

DPV 7.31