# Assignment 2
# CSC373 Winter 2009
# Due Date: Feb 24th, Noon, BA2220 Drop Box 7

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.

## 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 $t = s_{i_1} \cdot s_{i_2} \cdots s_{i_r}$, 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, $|\emptyset| = 0$)
- $t_i$ represents the $i^{th}$ symbol of t and $t_{i \ldots j}$ represents the substring of t from the $i^{th}$ symbol to the $j^{th}$ symbol, inclusively (indices start at 1, i.e., $t = t_{1 \ldots |t|}$ )

## Problem 2

A. Give an algorithm that takes as input a directed graph with only positive weighted edges and returns the weight of the smallest cycle in the graph. Your algorithm should run in time at most $O(|V|^3)$.

B. DPV 4.21(b). Hint: You may want to (re-)read Section 4.6 of DPV and recall logarithms from high school.

## Problem 3

DPV 6.20

# Problem 4

In HW1 we considered the problem of making change for n cents using as few coins as possible from the set $\{c_1, c_2 \ldots c_k\}$.

A. Design a Dynamic programming algorithm to compute the smallest number of coins that allow you to make change for n. If it is impossible to make change using the given coin set, you should report this. The running time should be O(nk).

B. Design an algorithm that will decide if it is possible to make change using at most one coin of each denomination. The algorithm should run in $O(nk^2)$ time.

C. Are these algorithms polynomial time? Briefly explain why or why not. State any assumptions.