COMP 3711H – Honors Design and Analysis of Algorithms
2014 Fall Semester – Written Assignment # 4
Distributed: November 18, 2014 – Due: December 1, 2014
Revised November 24, 2014

Your solutions should contain (i) your name, (ii) your student ID #, and (iii) your email address

Information:

- Please write clearly and briefly.
- Please follow the guidelines on doing your own work and avoiding plagiarism given on the class home page. Don't forget to *acknowledge* individuals who assisted you, or sources where you found solutions.
- Please make a *copy* of your assignment before submitting it. If we can't find your answers, we will ask you to resubmit the copy.
- Your solution should be submitted as a PDF. This can be generated by Latex, from Word or a scan of a (legible) handwritten solution, etc..
- Your solution should be submitted via the CASS system by 11:59PM on December 1, 2014. The class web page has reminders on how to use CASS.

Problem 1: [20 points] Suppose $A = \{a_1, a_2, \ldots, a_k\}$ is a set of distinct coin values (all the a_i are positive integers) available in a particular country. The *coin changing problem* is defined as follows: Given integer n find the *minimum* number of coins from A that add up to n assuming that all coins have value in A. You should assume that $a_1 = 1$ so that it is always possible to find some set that adds up to n.

For example, if , $A = \{1, 5, 8\}$ and $n = 26$ the best way of making change uses 4 coins, i.e., $\{5, 5, 8, 8\}$.

Design a $O(nk)$ dynamic programming algorithm for solving the coin changing problem; that is, given inputs A and n it outputs the minimum number of coins in A to add up to n. (It is not necessary to say what the coins are.) Prove that your algorithm is correct.

Problem 2: [20 points] *Arbitrage* is the use of discrepancies in currency-exchange rates to make a profit. For example, there may be a small window of time in which 1 U.S. dollar buys 0.75 British pounds, 1 British pound buys 2 Australian dollars and 1 Australian dollar buys 0.70 U.S. dollars. Then, a smart trader can trade one U.S. dollar and end up with $0.75 \times 2 \times 0.7 = 1.05$ U.S. dollars, a profit of 5% (note that this assumes there are no trading costs).

Suppose that there are n currencies $c_1, \ldots, c_n$ and an $n \times n$ table R of exchange rates such that one unit of currency c_i buys $R[i, j]$ units of currency c_j. The *value* of a series of exchanges $c_{i_1}, c_{i_2}, \ldots, c_{i_t}$ is

$$R[i_1, i_2] \cdot R[i_2, i_3] \cdots R[i_{t-1}, i_t].$$

which is the number of units of currency c_t that result when starting with one unit of currency c_1, exchanging it into currency c_2, exchanging what results into c_3, etc. until ending by exchanging units of c_{t-1} in currency c_t.

A series of exchanges yields a profit if it both starts and ends in the same currency and the value of the series is greater than 1.

Devise and analyze a dynamic programming algorithm that, given the $R[\ ]$ table, determines whether or not it is possible to make a profit by trading currencies. You must prove that your algorithm is correct and also give the running time of the algorithm using $O()$ notation.

Problem 3 [20 points]

Suppose you are given three strings of characters: $X = x_1 x_2 \cdots x_n$, $Y = y_1 y_2 \cdots y_m$, $Z = z_1 z_2 \cdots z_p$, where $p = n + m$. Z is said to be a *shuffle* of X and Y iff Z can be formed by interleaving the characters from X and Y in a way that maintains the left-to-right ordering of the characters from each string. The goal of this problem is to design an efficient dynamic-programming algorithm that determines whether Z is a shuffle of X and Y.

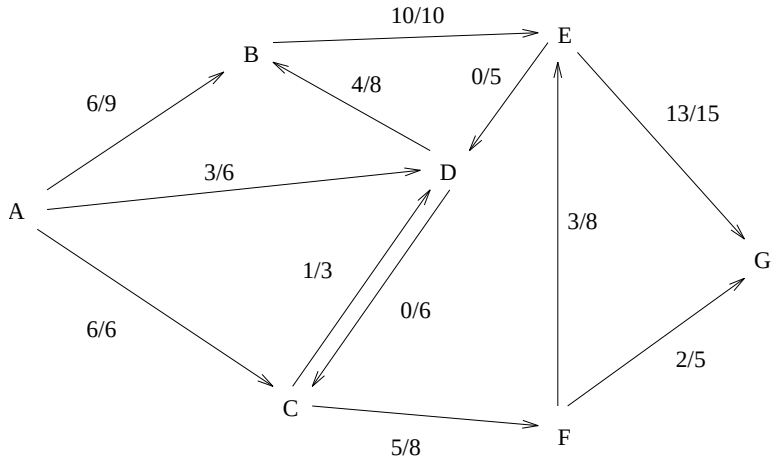
(a) Show that *cchocohilaptes* is a shuffle of *chocolate* and *chips*, but that *chocochilatspe* is not.

(b) For any string $A = a_1a_2 \cdots a_r$, let $A_i = a_1a_2 \cdots a_i$ be the substring of A consisting of the first i characters of A. For example, if A is *chocolate*, then A_3 is *cho* and A_6 is *chocol*.

Define $f(i, j)$ to be 1 if Z_{i+j} is a shuffle of X_i and Y_j, and 0 otherwise. Derive a recursive formula for $f(i, j)$. Remember to include the basis cases. Briefly explain your derivation.

(c) Give an efficient algorithm for determining whether Z is a shuffle of X and Y. Analyze the running time of your algorithm.

Problem 4: [20 points]



(a) Given the above graph with flow values f and capacities c as shown (the values are written as f/c following the notation on the class slides), draw the residual network.

(b) Find an augmenting path p in the residual network and draw it, showing the maximum flow f_p that can be pushed through p.

(c) Draw the new flow $f' = f + f_p$.

Problem 5: [20 points]

Prove the following three statements about flows from page 9 of the Network Flow slides. V is the set of vertices of the graph and f is a flow,

(a) $\forall X \subseteq V, \quad f(X, X) = 0.$

(b) $\forall X, Y \subseteq V, \quad f(X, Y) = -f(Y, X).$

(c) $\forall X, Y, Z \subseteq V$ with $X \cap Y = \emptyset$

$$f(X \cup Y, Z) = f(X, Z) + f(Y, Z) \quad \text{and}$$
$$f(Z, X \cup Y) = f(Z, X) + f(Z, Y).$$