CS3230
Tutorial 6

1. Give an argument to show why dynamic programming algorithm gives optimal solution for coin changing problem.

2. Compute the longest common subsequence for the following. Give the table as generated by the dynamic programming algorithm.

   (a) $SLWOVNNDK$ and $ALWGQVNBBK$.

   (b) $AGCGATAGC$ and $ACAGATGAG$

3. (a) A Chomsky Normal Form grammar G is of the form $G = (\Sigma, V, S, \delta)$, where Σ is the alphabet set, V is a set of non-termimals (where $V \cap \Sigma = \emptyset$), $S \in V$ is a starting symbol and δ is a set of productions of the form:

   $$A \to BC \text{ or } A \to a, \text{ where}$$

   $A, B, C \in V$ and $a \in \Sigma$ is a terminal.

   In the following α, β, γ, w are strings in $(\Sigma \cup V)^*$.

   (b) $\alpha A \beta \Rightarrow_G \alpha w \beta$, if $A \to w$ is a production in G (that is member of δ).

   (c) $\alpha \Rightarrow_G^* \beta$ if one of the following holds: (i) $\alpha = \beta$, (ii) $\alpha \Rightarrow_G \beta$ or (iii) for some γ, $\alpha \Rightarrow_G \gamma$ and $\gamma \Rightarrow_G^* \beta$

   (d) $L(G) = \{w : w \in \Sigma^* \text{ and } S \Rightarrow_G^* w\}$.

   Given a Chomsky Normal Form grammar G, give a dynamic programming algorithm to determine if a string $w \in \Sigma^*$ is a member of $L(G)$.

4. Suppose the following coins are available in a country:

   1 cents, 3 cents, 5 cents, 10 cents, 20 cents, 25 cents, 50 cents.

   (a) What are the coins selected by the greey method for 48 cents?

   (b) Show that the greedy method does not give optimal number of coins.

5. (a) Suppose we modify the greedy algorithm for fractional knapsack problem to consider the objects in order of "non-increasing" value (rather than non-increasing ratio of value/weight as done in class).

   Is the modified algorithm still optimal? If so, give an argument for its optimality. If not, give a counterexample.

(b) Suppose we modify the greedy algorithm to consider the objects in order of "non-decreasing" weight (rather than non-increasing ratio of value/weight as done in class).

Is the modified algorithm still optimal? If so, give an argument for its optimality. If not, give a counterexample.

6. Consider the following undirected graph.

   $G = (V, E)$, where the set of vertices is $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$ and edges and their weights are given as follows:

   $wt(1,2) = 3$, $wt(1,5) = 2$, $wt(1,4) = 10$, $wt(2,3) = 4$, $wt(2,5) = 9$, $wt(3,5) = 6$,

   $wt(3,6) = 5$, $wt(4,5) = 4$, $wt(4,7) = 4$, $wt(5,6) = 3$, $wt(5,7) = 6$, $wt(5,8) = 2$,

   $wt(5,9) = 6$, $wt(6,9) = 6$, $wt(7,8) = 8$, $wt(7,10) = 3$, $wt(8,9) = 8$, $wt(8,10) = 3$,

   $wt(8,11) = 5$, $wt(8,12) = 7$, $wt(9,12) = 4$, $wt(10,11) = 5$, $wt(11,12) = 2$

   Use Dijkstra's algorithm to find the shortest path to all the nodes from node 8.

7. Job Scheduling:

   Consider the problem of scheduling n jobs with run-times of $t_1, t_2, \ldots, t_n$ respectively. All the jobs arrive at time $t = 0$. Only job can run on the processor at a time, and it will run until it ends. You need to schedule at what time each job starts. Waiting time for a job is the time it starts executing. Give an algorithm to minimize the total waiting time. Is your algorithm optimal? If so prove it. If not, give a counter-example.