Algorithm Design, Assignment 2

Akiko Takeda (Problems are given by Yutaro Yamaguchi)

Due date: January 9, 2015

Choose two problems from the below and solve them. You must choose one from 1–4, and the other from 5–7. The notation $\mathrm{O}(f(x))$ or $\Omega(f(x))$ means that $cf(x)$ is an upper-bound or lower-bound, respectively, for some constant c.

1. Consider the situation when we pay an arbitrary positive-integer amount of money by using only three kinds of coins whose values are $1, 5, 7$. In this case, the greedy algorithm that use as many coins as possible in decreasing order of the values does not always return a payment with the fewest coins. Based on the dynamic programming, construct an algorithm that finds a payment with the fewest coins for any amount, and show the correctness and the computational time bound.

2. Let n be a positive integer. Given a sequence $a_1, a_2, \ldots, a_n$ of n integers, we want to find a consecutive subsequence whose sum is maximum. In the previous assignment, we construct an $\mathrm{O}(n \log n)$-time algorithm for this problem based on a divide-and-conquer idea, whereas it takes $\Omega(n^2)$ time to check all patterns of the starts and ends of consecutive subsequences. In this assignment, construct an $\mathrm{O}(n)$-time algorithm based on the dynamic programming, and show the correctness.

3. Let n be a positive integer, and so $a_1, a_2, \ldots, a_n$. For each $i = 1, 2, \ldots, n$, there is a box M_i containing a_i coins. Consider the following two-player game between Players A and B.

   (i) Players alternately do the following procedure (ii), and the player who cannot do it loses. The first move is done by Player A.

   (ii) The player chooses a box M_i $(1 \leq i \leq n)$ containing at least one coin, and remove an arbitrary amount of coins, but at least one, from M_i.

   For an arbitrary input $(n; a_1, a_2, \ldots, a_n)$, either Player A or B can win at this game absolutely. Based on the dynamic programming, construct an algorithm to test which player can win at the game for such an input, and show the correctness and the computational time bound.

4. For a directed acyclic graph (i.e., it contains no directed cycle) with nonnegative length on each arc, we want to find a longest path from the start s to the end t. Solve the following problems.

   (a) Show that any directed acyclic graph contains a vertex with no entering arc.

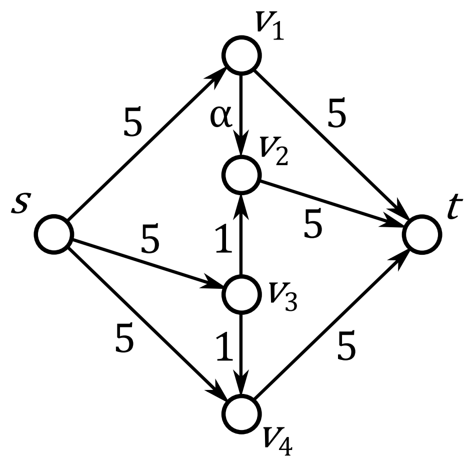
   (b) Using an idea to remove a vertex with no entering arc repeatedly, based on the dynamic programming, construct an algorithm for this problem.

   (c) Show the correctness and the computational time bound of your algorithm.

5. Ford–Fulkerson algorithm (choose an arbitrary augmenting path in a residual network, and increase the flow along the path) for finding a maximum flow does not always halt when there exists an arc with an irrational capacity. Show such an example using the network figured below, along the following procedures. In the figure, the number and script by each arc represents its capacity, and let α be a real with $1/2 < \alpha < 1$.

   (a) Choose $P_1 = (s, v_3, v_2, t)$, $P_2 = (s, v_1, v_2, v_3, v_4, t)$, $P_3 = (s, v_3, v_2, v_1, t)$, P_2, $P_4 = (s, v_4, v_3, v_2, t)$ as augmenting paths and update the flow in this order. Compute the residual capacity (obtained by subtracting the flow value from the capacity) on each arc v_1v_2, v_3v_2, v_3v_4 just after each update.

   (b) Show that there exists an irrational α such that we can update the flow infinitely repeatedly along the augmenting paths P_2, P_3, P_2, P_4 after (a).



6. Suppose that you choose a shortest augmenting path (i.e., with the fewest arcs) in each iteration step of Ford–Fulkerson algorithm. Solve the following problems.

   (a) For a nonnegative integer k and a vertex v, let $d_k(v)$ denote the length of a shortest directed path from the source s to v in the residual network (if there is no such path, then define it as $\#(\text{vertices}) + 1$) just after the k-th flow update. Show that $d_k(v) \leq d_{k+1}(v)$ for any vertex v.

   (b) Just after updating the flow, at least one arc in the selected augmenting path is removed in the residual network. For integers i, j with $0 < i < j$, suppose that an arc uv which was removed just after the i-th flow update appears in the residual network again for the first time just after the j-th flow update. Show $d_{i-1}(u) < d_{j-1}(u)$.

   (c) Estimate the number of iterations (i.e., of flow updates) in this situation.

7. Regarding Menger's theorem, "for any distinct vertices s, t in any directed graph, the maximum number of arc-disjoint directed paths from s to t is equal to the minimum number of arcs which are removed to make the graph containing no directed path from s to t," solve the following problems. You can use this theorem.

   (a) Show the undirected graph version, "for any distinct vertices s, t in any undirected graph, the maximum number of edge-disjoint undirected paths from s

to t is equal to the minimum number of edges which are removed to make the graph containing no undirected path from s to t."

(b) Show the vertex-disjoint version, "for any distinct vertices s, t in any directed graph, the maximum number of vertex-disjoint directed paths from s to t is equal to the minimum number of vertices except for s and t which are removed to make the graph containing no directed path from s to t." Here, the term "vertex-disjoint" means that each two paths do not share any vertex in between (i.e., except for s and t).