

Homework Assignment #4

Due: November 30, 2010, by 10:30 am

(in the drop box for your CSCC73 tutorial section, near room SW-626A)

Appended to this document is a cover page for your assignment. Fill it out, staple your answers to it, and deposit the resulting document into the course drop box (without putting it in an envelope).

Question 1. (10 marks)

- a. (2 marks) Give an example of a flow graph G with integer capacities, and a *maximum* flow f in G such that for some edge e , $f(e)$ is *not* an integer. Prove that your flow f is a maximum flow in G .
- b. (5 marks) Let G be any flow graph with integer capacities, and f be any maximum flow in G . Assume that the value of f is positive (i.e., the source and sink are connected through at least one path in which every edge has positive capacity). Prove that there is an edge e such that $f(e)$ is a positive integer. (In other words, it is not possible to construct an example as in part (a) where *every* edge used by the flow carries a fractional amount of traffic.)
- c. (3 marks) Let G be any flow graph with integer capacities, and m be the value of the maximum flow in G . Prove that, for each integer k such that $0 \leq k \leq m$, there is a flow f that has value k .

Question 2. (20 marks) The Ford-Fulkerson algorithm leaves the choice of augmenting path unspecified. In the Edmonds-Karp version of this algorithm we always choose an augmenting path with the smallest number of edges. In this question we explore another version, where we always choose an augmenting path *with the largest possible bottleneck*. We call such a path a *maximum-bottleneck augmenting path*.

Recall that the bottleneck of an augmenting path is the minimum residual capacity amongst the edges of the path. In the first part of this question we consider how to efficiently find such a path. In the other parts we get an upper bound on the number of augmenting steps required by this version of the Ford-Fulkerson algorithm.

- a. (5 marks) Suppose we are given a directed graph G with a positive “capacity” $c(e)$ on each edge e , and two distinguished nodes s and t . Explain briefly how to modify Dijkstra’s algorithm to find a path P from s to t in G so that the minimum capacity of the edges on P is as large as possible. (We can use this algorithm to find a maximum-bottleneck augmenting path.)

For the remaining parts of this question, let G be a flow graph, f^* be a maximum flow in G , and m be the number of edges in G .

- b. (5 marks) Let f be a flow in G , and δ be an upper bound on the residual capacity of all augmenting paths for f . That is, every path from s to t in G_f has an edge with residual capacity $\leq \delta$. Prove that $V(f^*) - V(f) \leq m \cdot \delta$.

Hint. Recall the proof that if there is no augmenting path in G_f (i.e., $\delta = 0$), then there is a cut with capacity equal to $V(f)$, and therefore $V(f^*) - V(f) = 0$. Modify this proof.

- c. (5 marks) Let f be a flow in G such that G_f has a path from s to t , P be a maximum-bottleneck augmenting path in G_f , and $f' = \text{AUGMENT}(f, P)$ (i.e., f' is the flow obtained by improving f via a maximum-bottleneck augmenting path in G_f). Use part (b) to show that $V(f^*) - V(f') \leq (V(f^*) - V(f)) \cdot (1 - \frac{1}{m})$.

d. (5 marks) Suppose that all capacities in G are integers, and let C be the sum of the capacities of all edges out of s . Prove that if we always choose a maximum-bottleneck augmenting path, the Ford-Fulkerson algorithm terminates after at most $O(m \log C)$ augmentation steps. (Note that this implies that this variant of the Ford-Fulkerson algorithm runs in time polynomial in the number of bits of the input.)

Hint. You may find useful the inequality $1 - x \leq e^{-x}$, which holds for any real x (and is an equality only for $x = 0$).

Question 3. (10 marks) We are given a directed graph $G = (V, E)$ and two distinguished nodes $s, t \in V$. A set of s -to- t paths in G are **node-disjoint** if no two paths in the set share a node other than s and t . We wish to find a maximum-size set of node-disjoint s -to- t paths in G . Give an algorithm that does this. Explain why your algorithm is correct, and analyse its time complexity in terms of the number of nodes n and the number of edges m in G . For the purposes of this problem, you can assume (without loss of generality) that there are no edges into s or out of t in the graph G .

Hint. Transform the problem of finding **node-disjoint** paths in G , into the problem of finding **edge-disjoint** paths in a related graph.

Question 4. (10 marks) Let $S = \{s_1, \dots, s_n\}$ be the set of Master's students in the Department of Computer Science (DCS), and $C = \{c_1, \dots, c_m\}$ be the set of graduate courses offered by DCS. C is partitioned into two sets T_1 and T_2 , the set of courses offered in the fall and winter term, respectively.

Each course c_i has an enrolment limit of ℓ_i . Each student s_j chooses a set $C_j \subseteq C$ of courses that s_j is interested in taking. (C_j may be any subset of C , including the empty set!)

DCS must assign each Master's student to a set of courses that the student is interested in taking. A student may be assigned to at most 5 courses, at most 3 of which may be in the same term. Furthermore, no course should be assigned more students than its enrolment limit stipulates.

a. DCS receives from the university \$1,000 for each student in each course. How should it assign students to courses so as to maximise its revenue? Express this problem as an instance of network flow. Justify your answer.

b. A student can graduate if he/she takes 5 courses. How should DCS assign students to courses so as to maximise the number of students who can graduate? Express this problem as a 0-1 integer program. Justify your answer.

THAT'S IT WITH HOMEWORK, FOLKS!

Cover page for CSCC73 Assignment #4

Submitted by

(1) Family Name: _____

(2) Family Name: _____

Given Name: _____

Given Name: _____

Student Number: _____

Student Number: _____

By virtue of submitting this homework I/we acknowledge that I am/we are aware of the policy on homework collaboration for this course.