NOTE TO STUDENTS: This file contains sample solutions to the term test together with the marking scheme for each question. Please read the solutions and the marking schemes carefully. Make sure that you understand why the solutions given here are correct, that you understand the mistakes that you made (if any), and that you understand *why* your mistakes were mistakes.

Remember that although you may not agree completely with the marking scheme given here it was followed the same way for all students. We will remark your test only if you clearly demonstrate that the marking scheme was not followed correctly.

For all remarking requests, please submit your request **in writing** directly to your instructor. For all other questions, please don't hesitate to ask your instructor during office hours or by e-mail.

Question 1. [9 MARKS]

Let $G = (V, E)$ be an undirected weighted graph, with weight $w(e)$ on each edge $e \in E$.

Part (a) [3 MARKS]

If the minimum weight in G is unique (*i.e.*, $w(e_1) < w(e_2) \leq w(e_3) \leq \cdots \leq w(e_m)$), prove or disprove that every minimum spanning tree of G always *contains* edge e_1.

SAMPLE SOLUTION:

> **Proof**: For a contradiction, assume there is a MST T that does not contain edge e_1. Then, T contains some path between the endpoints of e_1. Let e be any edge on this path. Then, $T' = T \cup \{e_1\} - \{e\}$ is a spanning tree whose total weight is strictly smaller than the total weight of T (because $w(e_1) < w(e)$). This contradicts the fact that T was a MST.

MARKING SCHEME:

A. 1: attempt to prove
B. 2: correct proof (-1 for anything like "Kruskal's algorithm always includes e_1", because this does not prove the claim for *every* MST)

Part (b) [3 MARKS]

If the maximum weight in G is unique (*i.e.*, $w(e_1) \leq w(e_2) \leq \cdots \leq w(e_{m-1}) < w(e_m)$), prove or disprove that every minimum spanning tree of G always *excludes* edge e_m.

SAMPLE SOLUTION:

> **Disproof**: If G is a tree, then there is only one spanning tree of G and it must include every edge.

MARKING SCHEME:

A. 1: attempt to disprove
B. 2: correct counter-example

Part (c) [3 MARKS]

Prove or disprove that the execution of Kruskal's and Prim's algorithm on G will always output the same minimum spanning tree.

SAMPLE SOLUTION:

> No: Let G be a triangle, a graph with three vertices and three edges, with all edges weighted 1. Then any one of the three MST's of G can be the output of either Kruskal's or Prim's algorithm. So, the executions of these two algorithms can output different MST's.

MARKING SCHEME:

A. 1: correct answer "no"
B. 2: correct justification

Question 2. [16 MARKS]

Explain how to solve each of the following problems using network flow techniques or linear programming—in particular, explain how to reconstruct a solution to the original problem given a solution to your network flow problem or linear program, and justify that this solution is guaranteed to be optimal.

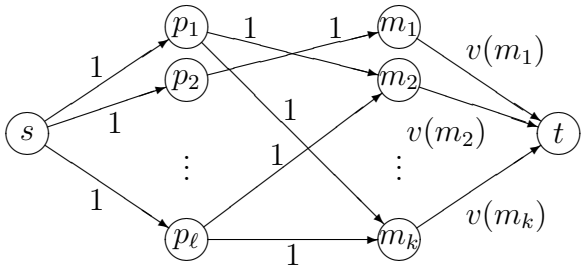
Part (a) [8 MARKS]

In the "disaster relief" problem, you are given a set of locations of injured people and a set of locations of medical facilities, along with the number of new patients that each facility can accept. You want to determine if it is possible to transport every injured person to a nearby facility so that every person gets treated—without overloading any facility.

More precisely, given the positions $p_1, \ldots, p_\ell$ of each injured person and $m_1, \ldots, m_k$ of each medical facility, the distances $d(p_i, m_j)$ between any person and facility, a distance bound D, and the "vacancy" $v(m_i)$ for each facility, determine an assignment of persons to facilities such that each person is assigned to a facility no more than distance D away, and each facility m_j is assigned no more than $v(m_j)$ new patients.

SAMPLE SOLUTION:

Create a network $N = (V, E)$ as pictured below, where there is an edge (s, p_i) with capacity 1 for each person p_i, an edge (m_j, t) with capacity $v(m_j)$ for all medical facilities m_j, and an edge (p_i, m_j) with capacity 1 for each person p_i and medical facility m_j such that $d(p_i, m_j) \leqslant D$.



Then, find a maximum flow in N and assign each person p_i to the medical facility m_j such that $f(p_i, m_j) = 1$ (or don't assign p_i to any medical facility if there is no such edge).

Every valid assignment of people to medical facilities yields a valid flow in N, by setting $f(s, p_i) = 1$ if p_i has been assigned to some medical facility, $f(s, p_i) = 0$ otherwise; $f(p_i, m_j) = 1$ if p_i has been assigned to facility m_j, $f(p_i, m_j) = 0$ otherwise; and $f(m_j, t) =$ the number of new patients assigned to facility m_j. Such a flow respects both the capacity and conservation constraints.

Conversely, every valid flow in N yields a valid assignment of people to medical facilities, by assigning each person p_i to a medical facility m_j such that $f(p_i, m_j) = 1$. Because the capacity into p_i is exactly 1, each person can be assigned to at most one facility. Because the capacity out of m_j is $v(m_j)$, each medical facility is assigned at most $v(m_j)$ new patients.

Hence, the value of a maximum flow is equal to the maximum number of people that can be assigned to medical facilities and the algorithm is guaranteed to produce an optimal solution.

MARKING SCHEME:

- **Structure:** [1 mark]
  clear attempt to give an explicit network/LP to represent the original problem and to justify that the optimal solution can be obtained from the network/LP
- **Network/LP:** [4 marks]
  correct network/LP (even if correctness is not justified)
- **Justification:** [3 marks]
  reasonable explanation of how to reconstruct solution from network/LP and justification that solution is optimal

Part (b) [8 MARKS]

In the "minimum cost flow problem", you are given a network with a *price* $p(e)$ for each edge e (where $p(e)$ represents the cost per unit flow, *i.e.*, sending x units of flow across edge e will cost $p(e) \cdot x$), and you want to find a flow in the network that meets a certain target while keeping the cost as low as possible (where the cost for the entire network is the sum of the costs for each edge in the network).

More precisely, given a network $N = (V, E)$ with non-negative integer capacity $c(e)$ and non-negative integer price $p(e)$ for each edge $e \in E$, together with a non-negative integer demand d, find a flow f in N (*i.e.*, an assignment of flow value $f(e)$ for each edge $e \in E$) such that the total flow in N is at least d (*i.e.*, $|f| = f^{\text{out}}(s) = \sum_{(s,u)\in E} f(s,u) \geqslant d$) and the total cost of the flow, $\text{cost}(f) = \sum_{e\in E} p(e) \cdot f(e)$, is minimum.

SAMPLE SOLUTION:

Construct the following linear program, over variables f_e for each edge $e \in E$:

$$\begin{aligned} \text{minimize} \quad & \sum_{e\in E} p(e) \cdot f_e \\ \text{subject to:} \quad & \sum_{(s,u)\in E} f_{(s,u)} \geqslant d \\ & \sum_{(u,v)\in E} f_{(u,v)} = \sum_{(v,u)\in E} f_{(v,u)} \qquad \text{for all } v \in V \\ & 0 \leqslant f_e \leqslant c(e) \qquad \text{for all } e \in E \end{aligned}$$

Then, assign flow $f(e) = f_e^*$ to each edge $e \in E$, for any optimal solution f_e^* to the linear program.

Every valid flow in N yields a set of values for f_e that satisfies all of the constraints, and conversely, every feasible solution f_e to the linear program yields a valid flow in N (because the network constraints are represented exactly in the linear constraints).

Hence, every optimal solution to the linear program yields an optimal solution to the network flow problem.

MARKING SCHEME:

- **Structure:** [1 mark]
  clear attempt to give an explicit network/LP to represent the original problem and to justify that the optimal solution can be obtained from the network/LP
- **Network/LP:** [4 marks]
  correct network/LP (even if correctness is not justified)
- **Justification:** [3 marks]
  reasonable explanation of how to reconstruct solution from network/LP and justification that solution is optimal