

Duration: **50 minutes**
 Aids Allowed: **NONE** (in particular, no calculator)

Student Number:

Last (Family) Name(s):

First (Given) Name(s):

*Do **not** turn this page until you have received the signal to start.*
 In the meantime, please read the instructions below carefully.

This term test consists of 3 questions on 10 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete, fill in the identification section above, write your student number where indicated at the bottom of every odd-numbered page (except page 1), and write your name on the back of the last page.*

Answer each question directly on the test paper, in the space provided, and use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and *indicate clearly the part of your work that should be marked.*

In your answers, you may use without proof any result or theorem covered in lectures, tutorials, homework, tests, or the textbook, as long as you give a clear statement of the result(s)/theorem(s) you are using. You must justify all other facts required for your solutions.

Write up your solutions carefully! In particular, use notation and terminology correctly and explain what you are trying to do — part marks *will* be given for showing that you know the general structure of an answer, even if your solution is incomplete.

If you are unable to answer a question (or part), you will get 20% of the marks for that question (or part) if you write “I don’t know” and nothing else — you will *not* get those marks if your answer is completely blank, or if it contains contradictory statements (such as “I don’t know” followed or preceded by parts of a solution that have not been crossed off).

MARKING GUIDE

1: _____/11

2: _____/12

3: _____/10

TOTAL: _____/33

Use this page for rough work — clearly indicate any section(s) to be marked.

Question 1. [11 MARKS]

For any network $N = (V, E)$, we can classify the vertices into three types:

- *upstream* vertices belong to the source side S of every minimum-capacity cut (S, T) ;
- *downstream* vertices belong to the sink side T of every minimum-capacity cut (S, T) ;
- *mid-stream* vertices belong to the source side S of some minimum-capacity cuts (S, T) and to the sink side T' of some minimum-capacity cuts (S', T') .

For example, s is an upstream vertex, v is a mid-stream vertex, and t is a downstream vertex in the network $N = s \xrightarrow{2} v \xrightarrow{2} t$.

Give an efficient algorithm that takes a network $N = (V, E)$ and returns the set of all upstream, mid-stream, and downstream vertices in N . Justify that your algorithm is correct and runs in polytime.

Use this page for rough work — clearly indicate any section(s) to be marked.

Question 2. [12 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) [6 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, \dots, p_ℓ of each injured person and m_1, \dots, 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.

Use this page for rough work — clearly indicate any section(s) to be marked.

Question 2. (CONTINUED)**Part (b)** [6 MARKS]

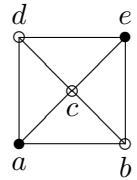
In the “minimum cost flow problem”, you are given a network with a *price* $p(e)$ for each edge e (where $p(e)$ represents the cost per unit flow, *i.e.*, sending x units of flow across edge e will cost $p(e) \cdot x$), and you want to find a flow in the network that meets a certain target while keeping 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.*, $|f| = f^{\text{out}}(s) = \sum_{(s,u) \in E} f(s,u) \geq d$) and the total cost of the flow, $\text{cost}(f) = \sum_{e \in E} p(e) \cdot f(e)$, is minimum.

Use this page for rough work — clearly indicate any section(s) to be marked.

Question 3. [10 MARKS]

A *cut* in an undirected graph $G = (V, E)$ is any partition of the vertices into sides $S \subseteq V$ and $\bar{S} = V - S$. The *size* of a cut (S, \bar{S}) is measured as the number of edges across the cut, *i.e.*, with one endpoint in S and the other in \bar{S} . For any cut (S, \bar{S}) and any vertex $v \in V$, “ $\text{cut}(v)$ ” represents the set of edges adjacent to v that cross the cut, and “ $\text{uncut}(v)$ ” represents the set of edges adjacent to v that do not cross the cut. For example, $\text{cut}(c) = \{(a, c), (c, e)\}$ and $\text{uncut}(c) = \{(b, c), (c, d)\}$ for the graph pictured on the right, where $S = \{a, e\}$.



Consider the following algorithm that attempts to find a large cut in an undirected graph G :

```

start with any non-empty subset  $S \subset V$ 
while there is some vertex  $v \in V$  with  $|\text{uncut}(v)| > |\text{cut}(v)|$ :
    move  $v$  from  $S$  to  $\bar{S}$  (or from  $\bar{S}$  to  $S$ )
    
```

Prove that for all undirected graphs G , this algorithm will produce a cut whose size is at least $1/2$ of the maximum cut size for G . (HINT: When the algorithm stops, what can you say about each vertex v and its adjacent edges? What does this imply for the size of the cut produced by the algorithm, compared to the total number of edges in G ? Keep in mind that for any undirected graph $G = (V, E)$, $\sum_{v \in V} \text{deg}(v) = 2|E|$, where $\text{deg}(v)$ is the *degree* of v , *i.e.*, the number of edges adjacent to v .)

On this page, please write nothing except your name.

Last (Family) Name(s): _____

First (Given) Name(s): _____