NOTE TO STUDENTS: This file contains sample solutions to the term test together with the marking scheme and comments for each question. Please read the solutions and the marking schemes and comments 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 or that the marker misunderstood your solution.

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

Question 1. [7 MARKS]

Part (a) [4 MARKS]

True or False? For each statement below, place a checkmark (✓) in the appropriate box.

TRUE	FALSE	
☐	☑	For all decision problems D_1 and D_2, if $D_1 \to_p D_2$ and $D_1 \in NP$, then $D_2 \in NP$.
☑	☐	For all decision problems D_1 and D_2, if $D_1 \to_p D_2$ and D_1 is NP-hard, then D_2 is NP-hard.
☑	☐	For all decision problems D_1 and D_2, if $D_1 \to_p D_2$ and $D_2 \in NP$, then $D_1 \in NP$.
☐	☑	For all decision problems D_1 and D_2, if $D_1 \to_p D_2$ and D_2 is NP-hard, then D_1 is NP-hard.
☐	☑	For all decision problems D, if $D \in coNP$, then $D \notin P$.
☑	☐	Every Linear Program can be expressed as a *maximization* problem (through appropriate adjustments to the objective function).
☐	☑	For all networks N and cuts X, if the capacity of every edge across X is reduced by 1, then the value of the maximum flow in N is reduced by at least 1.
☐	☑	For all decision problems D_1 and D_2, if D_2 can be solved in worst-case time $\mathcal{O}(n^2)$ and $D_1 \to_p D_2$, then D_1 can also be solved in worst-case time $\mathcal{O}(n^2)$.

SAMPLE SOLUTION: (See above.)

MARKING SCHEME AND COMMENTS: 0.5 mark for each statement

Part (b) [3 MARKS]

State the two properties you must prove to conclude that a decision problem D is NP-complete:

1. $D \in NP$
   (the "easy" property)
2. D is NP-hard
   (the "hard" property)

Then, complete the following description of how to prove that D satisfies the "hard" property:

Show that D' *(decision problem)* $\to_p$ D *(decision problem)*
for some / all *(quantifier)* decision problem(s) D' that is(are) NP-hard / in NP *(property)*.

SAMPLE SOLUTION: (See above. Note that there are at least two obvious correct solutions.)

MARKING SCHEME AND COMMENTS: 0.5 mark for each element

Question 2. [10 MARKS]

Part (a) [5 MARKS]

Company A has factories in n cities to make its products. It also runs one shop in each of these cities to sell its products. For $1 \leqslant i \leqslant n$, the maximum number of products that the factory can produce at city i in one day is $p_i \in \mathbb{Z}^+$. The maximum number of products that the shop at city i can sell in one day is $q_i \in \mathbb{Z}^+$. In addition, there are one-way highways connecting some pairs of cities. The highway between city i and city j allows a maximum of $c_{i,j} \in \mathbb{Z}^+$ products to be transferred from city i to city j every day (if there is no highway from city i to city j, $c_{i,j} = 0$). Your goal is to maximize the sales of the product while maintaining the balance between the factory production and the shop consumption—at the end of the day, every product that has been produced during that day must be sold. What is the maximum number of products Company A can produce and sell per day, and how will it achieve this goal?

For example, suppose $n = 3$, $p_1 = 20$, $p_2 = 0$, $p_3 = 1$, $q_1 = 2$, $q_2 = 4$, $q_3 = 10$, $c_{1,2} = 7$, and $c_{2,3} = 10$ (the remaining $c_{i,j} = 0$). Then the optimum solution produces and sells 10 products every day: city 1 produces 9 products, sells 2, and sends 7 to city 2; city 2 sells 4 products and sends 3 to city 3; city 3 produces 1 product and sells 4.

Design an algorithm to solve this problem using network flow techniques. Make sure that your algorithm outputs how much each factory will produce, how much each shop will sell, and how much will be shipped between any two cities. **For this part, do not discuss the correctness of your construction.**

SAMPLE SOLUTION:

- Create network $N = (V, E)$ as follows:
  - $V = \{s, a_1, a_2, \ldots, a_n, t\}$
  - $E = \{(s, a_i)$ with capacity $p_i : 1 \leqslant i \leqslant n\} \cup$
    $\{(a_i, t)$ with capacity $q_i : 1 \leqslant i \leqslant n\} \cup$
    $\{(a_i, a_j)$ with capacity $c_{i,j} : 1 \leqslant i, j \leqslant n$ and $c_{i,j} > 0\}$
- Find a maximum flow f^* over N.
- Output the following values:
  - production of city $i = f^*(s, a_i)$;
  - consumption of city $i = f^*(a_i, t)$;
  - shipping from city i to city $j = f^*(a_i, a_j)$.

MARKING SCHEME AND COMMENTS:

- **Structure** [1 mark]: clear attempt to construct a network, solve a network flow problem, and use the solution to generate an appropriate output (even if the network is incorrect)
- **Network** [2 marks]: clear description of a correct network
- **Output** [2 marks]: clear description of the correct output
- *Common Error* [−0.5]: many students forgot the last element of the output (how much to ship between pairs of cities)
- *Error Code* E_1 [−1]: using bipartite graph for the network (could work but had to be done carefully)

Question 2. (CONTINUED)

Part (b) [5 MARKS]

Company A bought some new trucks! Each new truck **doubles** the transportation capacity of a one-way highway between two cities, if it is deployed on that highway. Suppose the company can deploy such new trucks to a maximum of m highways (for some $m \in \mathbb{Z}^+$). What is the maximum number of products Company A can produce and sell per day under these new conditions?

Describe an algorithm to solve this problem with linear programming or integer linear programming. You only need to formulate the problem as a linear program or integer linear program and to justify *briefly* the correctness of your formulation (in other words, explain the meaning of each element of your answer).

SAMPLE SOLUTION:

- ***Integer** Variables:*
  - x_i for $1 \leqslant i \leqslant n$ (the amount of production for each city)
  - y_i for $1 \leqslant i \leqslant n$ (the amount of consumption for each city)
  - $z_{i,j}$ for $1 \leqslant i,j \leqslant n$ (the amount of shipping between cities)
  - $w_{i,j}$ for $1 \leqslant i,j \leqslant n$ (whether or not to deploy a truck between two cities)
- *Objective Function:* maximize $x_1 + x_2 + \cdots + x_n$ (total amount produced in every city) [maximize $y_1 + y_2 + \cdots + y_n$ is also fine]
- *Constraints:*
  - $0 \leqslant x_i \leqslant p_i$ for $1 \leqslant i \leqslant n$ (production for city i cannot exceed p_i)
  - $0 \leqslant y_i \leqslant q_i$ for $1 \leqslant i \leqslant n$ (consumption for city i cannot exceed q_i)
  - $0 \leqslant w_{i,j} \leqslant 1$ for $1 \leqslant i,j \leqslant n$ (there is at most one truck deployed on each highway)
  - $\sum_{1 \leqslant i,j \leqslant n} w_{i,j} \leqslant m$ (there are at most m trucks deployed in total)
  - $0 \leqslant z_{i,j} \leqslant (1 + w_{i,j})c_{i,j}$ for $1 \leqslant i,j \leqslant n$ (the shipping capacity from city i to city j is either $c_{i,j}$ or $2c_{i,j}$, depending on whether or not a truck is deployed on that highway)
  - $x_i + \left(\sum_{j=1}^{n} z_{j,i}\right) - \left(\sum_{j=1}^{n} z_{i,j}\right) - y_i = 0$ for $1 \leqslant i \leqslant n$ (for each city, the number of products produced and shipped in is equal to the number of products shipped out and consumed)

MARKING SCHEME AND COMMENTS:

- **Structure** [1 mark]: clear attempt to define variables, objective function, and constraints
- **Variables** [1 mark]: correct variables (including explanations)
- **Objective Function** [1 mark]: correct objective function (including explanation)
- **Constraints** [2 marks]: correct constraints (including explanations)
- *Error Code E_1* [−1]: deploying more than one truck on a single highway

Question 3. [8 MARKS]

Recall the definition of the SUBSETSUM decision problem:

- *Input:* A finite set of **positive** integers $S = \{x_1, x_2, \ldots, x_n\}$, and a **positive** integer target t.
- *Question:* Is there a non-empty subset of S whose sum is exactly t (i.e., $\exists S' \subseteq S, S' \neq \varnothing \wedge \sum_{x \in S'} x = t$)?

Consider the following ZEROSUM decision problem:

- *Input:* A finite set of integers $S = \{x_1, x_2, \ldots, x_n\}$ (each x_i can be positive, negative, or zero).
- *Question:* Is there a non-empty subset of S whose sum is zero (i.e., $\exists S' \subseteq S, S' \neq \varnothing \wedge \sum_{x \in S'} x = 0$)?

Write a detailed proof that ZEROSUM is NP-complete. You may use the fact that SUBSETSUM is NP-hard.

SAMPLE SOLUTION:

ZEROSUM $\in$ *NP*: The following verifier runs in polytime and outputs TRUE for some c iff S is a yes-instance of ZEROSUM:

$V(S, c)$: return ($c \subseteq S$ and $\sum_{x \in c} x = 0$)

ZEROSUM is *NP*-hard: We show that SUBSETSUM $\rightarrow_p$ ZEROSUM:

On input (S, t), output $S \cup \{-t\}$

Clearly, this can be computed in polytime.

Also, if S contains a subset S' whose sum is t, then the sum of the elements in $S' \cup \{-t\}$ is zero.

Finally, if $S \cup \{-t\}$ contains a non-empty subset S' whose sum is zero, then S' must contain $-t$ (because all other elements are positive), so $S' - \{-t\}$ is a subset of S whose sum is exactly t.

MARKING SCHEME AND COMMENTS:

- **Structure** [1 mark]: clear attempt to argue that ZEROSUM $\in$ *NP* by describing a verifier and ZEROSUM is *NP*-hard by giving a polytime reduction from some *NP*-hard problem (even if neither the verifier nor the reduction are actually correct)
- **Verifier** [1 mark]: verifier is correct and runs in polytime
- **Reduction** [6 marks]: reduction runs in polytime [1 mark], transforms yes-instances to yes-instances [2 marks], and transforms no-instances to no-instances [3 marks]
- *Error Code E_1* [-4 at least]: reduction in the wrong direction (ZEROSUM $\rightarrow_p$ SUBSETSUM)
- *Error Code E_2* [-4 at least]: reduction makes use of a certificate (algorithm implicitly or explicitly depends on a subset $S' \subseteq S$ such that $\sum_{x \in S'} x = t$)