THE UNIVERSITY OF SYDNEY

CONFIDENTIAL EXAM PAPER

This paper is not to be removed from the exam venue.

Computer Science

EXAMINATION

Semester 1 - Main, 2022

COMP3027 Algorithm Design

EXAM WRITING TIME: 2 hours
READING TIME: 10 minutes

EXAM CONDITIONS:
This is an OPEN book examination.

All submitted work must be **done individually** without consulting someone else's solutions, in accordance with the University's "Academic Dishonesty and Plagiarism" policies.

INSTRUCTIONS TO STUDENTS:
Type your answers in your text editor (LaTeX, Word, etc.) and convert it into a pdf file.

Submit this pdf file via Canvas. No other file formats will be accepted. Handwritten responses will **not** be accepted.

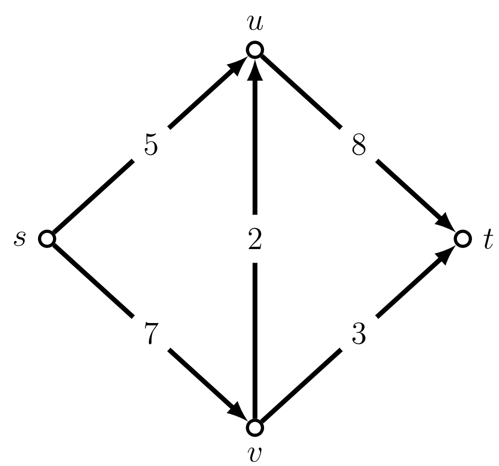
Start by typing your student ID at the top of the first page of your submission. Do **not** type your name.

Submit only your answers to the questions. Do **not** copy the questions.

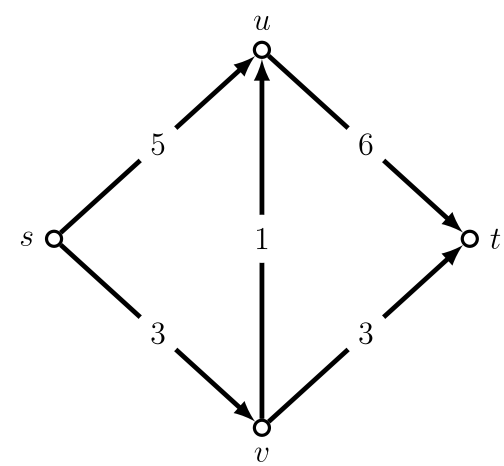
Do **not** copy any text from the permitted materials. Always write your answers in your own words

Problem 1 [10 marks]

Consider the network flow instance (G, c) on the left and s-t flow f on the right.



$c : E \to \mathbb{Z}^+$ (edge capacities)



$f : E \to \mathbb{Z}^+$ (current flow)

Your task is to run one iteration of the Ford-Fulkerson algorithm:

(a) Specify the augmenting path used to update the current flow.

(b) Specify the value of the updated flow.

(c) Specify a minimum cut and the capacity of the minimum cut.

(d) Use the updated flow to justify that the cut described in part (c) is a minimum cut.

Problem 2 [10 marks]

Consider the Partition problem: we are given a set I of n positive integers and want to decide if there is a partition of I into two sets I_1 and I_2 such that the sum of I_1 is equal to the sum of I_2. Consider the following greedy algorithm: initialize I_1 and I_2 to be empty; iterate through I in ascending order and place the current integer in whichever of I_1 and I_2 is lower. We break ties in favor of I_1. If at the end, the sum of I_1 equals the sum of I_2, we return "Yes" otherwise we return "No". In the following, we use w_i to denote the i-th smallest integer, i.e. $w_1 \leq w_2 \leq \ldots \leq w_n$.

Algorithm 1 Greedy

function $\textsc{Greedy}(I)$
    $I_1 \leftarrow \emptyset$ and $I_2 \leftarrow \emptyset$
    for $i \leftarrow 1; i \leq n; i++$ **do**
        if sum of I_1 is at most sum of I_2 **then**
            $I_1 \leftarrow I_1 \cup \{w_i\}$
        end if
    end for
    if sum of I_1 is equal to sum of I_2 **then**
        return Yes
    else
        return No
    end if
end function

Your task is to show that there exists a set I of positive integers such that there is a partition of I into sets J_1 and J_2 such that the sum of J_1 is equal to the sum of J_2 but the greedy algorithm returns No.

(a) Describe the set I.

(b) Describe the sets I_1 and I_2 at the end of the greedy algorithm.

(c) Describe a partition of I into two equal-sum sets J_1 and J_2.

Problem 3 [20 marks]

Let $T = (V, E)$ be a tree with positive edge weights $w(e)$. Recall that a matching M is a subset of edges such that no two edges in M are incident on the same vertex. The maximum weight matching problem is to find a matching M maximizing the sum of the weights of the edges in M, that is, maximize $\sum_{e \in M} w(e)$.

Your task is to design a dynamic programming algorithm that computes the weight of the maximum weight matching. (Note that your algorithm does not need to output the actual maximum weight matching, just its weight.) For full marks your algorithm should run in $O(n)$ time where n is the number of vertices in T.

(a) Clearly define the DP subproblems.

(b) State and justify the recurrence and base cases.

(c) Analyze its time complexity.

Problem 4 [20 marks]

We are given an undirected graph $G = (V, E)$ and we want to decide if there exists a spanning tree T of G where every vertex has degree at most 4. In particular, each vertex should be incident to at most 4 edges in T.

Your task is to prove that the above problem is NP-complete.

(a) Show that the problem belongs to the class NP.

(b) Prove that the problem is NP-hard. [Hint: Reduce from the Hamiltonian Path problem: Given a graph $H = (V', E')$, decide if there is a simple path that visits all vertices.]