CONFIDENTIAL EXAM PAPER

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

Computer Science

EXAMINATION

Semester 1 - Main, 2021

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.

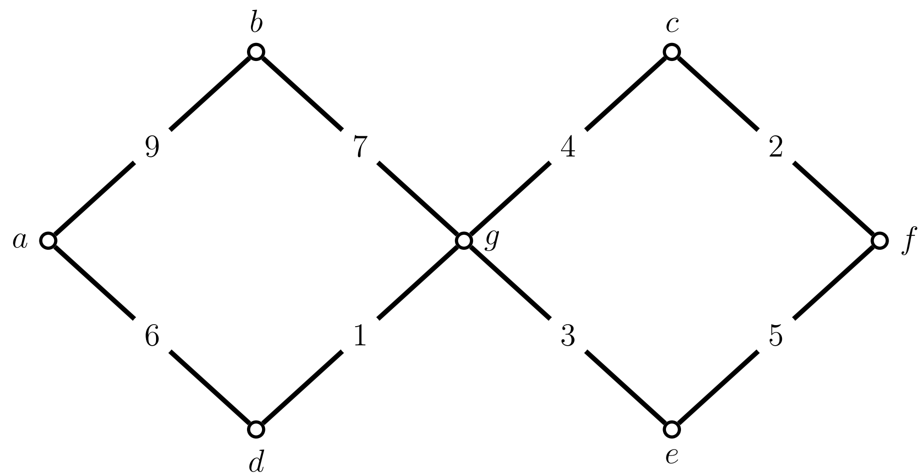
Start by typing your student ID at the top of the first page of your submission. Do **not** type your name.

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 following edge weighted undirected graph G:



Your task is to compute the minimum weight spanning tree T of G using Prim's algorithm starting at a:

(a) State the edges of T, in the order that the algorithm adds them to the solution.

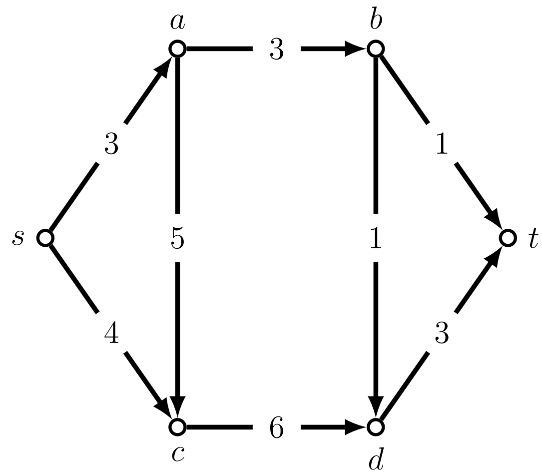
Other potential questions:

Draw a minimum spanning tree (MST) of G using Kruskal's algorithm. Label each node to indicate the order in which they were added to the MST.

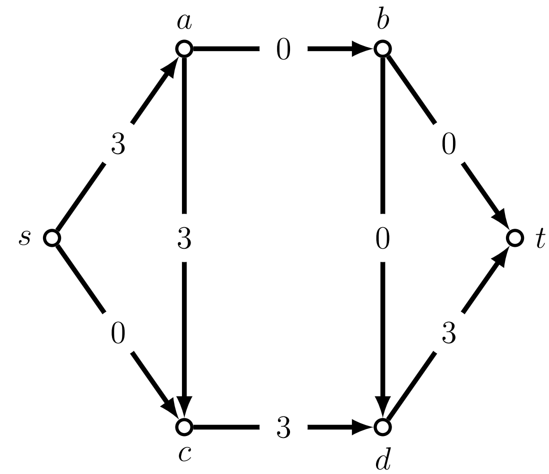
Run Dijkstra's Algorithm on G starting from node a, to calculate the length of the shortest path between a and each other node. Write down the order in which each vertex was extracted from the priority

Problem 2 [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)

(a) What is the value of this flow?

(b) Is this a maximum $s - t$ flow in this graph? If not, find a maximum $s - t$ flow.

(c) Find a minimum $s - t$ cut. (Specify which vertices belong to the sets of the cut.)

Other potential questions:

Draw the residual graph with respect to this flow.

Find an augmenting path in the residual graph.

Draw the updated flow.

Problem 3 [10 marks]

The product of two $n \times n$ matrices X and Y is a third $n \times n$ matrix $Z = XY$, where the (i, j) entry of Z is $Z_{ij} = \sum_{k=1}^{n} X_{ik}Y_{kj}$. This definition immediately leads to an $O(n^3)$ time algorithm for matrix multiplication. Here we explore the option of designing an alternative algorithm using divide and conquer. Suppose that X and Y are divided into four $n/2 \times n/2$ blocks each:

$$X = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \text{ and } Y = \begin{bmatrix} E & F \\ G & H \end{bmatrix}.$$

Using this block notation we can express the product of X and Y as follows

$$XY = \begin{bmatrix} AE + BG & AF + BH \\ CE + DG & CF + DH \end{bmatrix}.$$

In this way, one multiplication of $n \times n$ matrices can be expressed in terms of 8 multiplications and 4 additions that involve $n/2 \times n/2$ matrices. It is straightforward to translate this insight into a divide and conquer algorithm for matrix multiplication; unfortunately, this new algorithm's time complexity is again $O(n^3)$.

Suppose that instead of 8 recursive multiplications of $n/2 \times n/2$ matrices, we could compute the product using only 7 such matrix multiplications and a constant number of matrix addition operations. Let $T(n)$ be the time complexity of multiplying two $n \times n$ matrices using this improved recursive algorithm. **Your task** is to

(a) Derive the recurrence for $T(n)$. (Assume that adding two $k \times k$ matrices takes $O(k^2)$ time.)

(b) Solve the recurrence by unrolling it.

Other potential questions: Provide a counter-example which shows that a given algorithm does not produce a correct solution.

Problem 4 [20 marks]

Given an array A holding n objects, a *majority* element is an object that appears in more than $n/2$ positions of A. Assume we can test equality of two objects in $O(1)$ time, but we cannot use a dictionary indexed by the objects. **Your task** is to design an $O(n \log n)$ time algorithm for determining if there is a majority element.

(a) Describe your algorithm in plain English.

(b) Justify the correctness of your algorithm.

(c) Analyse the time complexity of your algorithm.

Problem 5 [20 marks]

Consider the problem of finding change using as few coins as possible. Formally, we are given as input a non-negative integer n, and we have unlimited quantities of 3 types of coins: 1c, 2c, 5c. The goal is to determine the minimum number of coins that add up to nc.

Your task is to design an $O(n)$ time algorithm for solving this problem using dynamic programming:

(a) Clearly define the DP subproblems.

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

(c) Analyze its time complexity.

Problem 6 [20 marks]

Consider the *3-partition* problem. Given integers $a_1, \ldots, a_n$, we want to determine whether it is possible to partition $\{1, \ldots, n\}$ into three disjoint sets I, J and K such that

$$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k = \frac{1}{3} \sum_{\ell=1}^{n} a_i.$$

Your task is to show that 3-partition is NP-complete:

(a) Show that 3-partition belongs to the class NP.

(b) Prove that 3-partition is NP-hard by showing that the *subset-sum* problem can be reduced in polynomial time to 3-partition; that is, prove that

$$\text{subset-sum} \leq_P \text{3-partition}.$$