Department of Electrical Engineering
National Taiwan University
Algorithms, Spring 2024

Quiz #3

Student Name:__________ Student ID: ____________ Class: ☐ Sun / ☐ Jiang

I pledge to follow the honor code of NTU and do not cheat in the exam.

Signature: ____________________

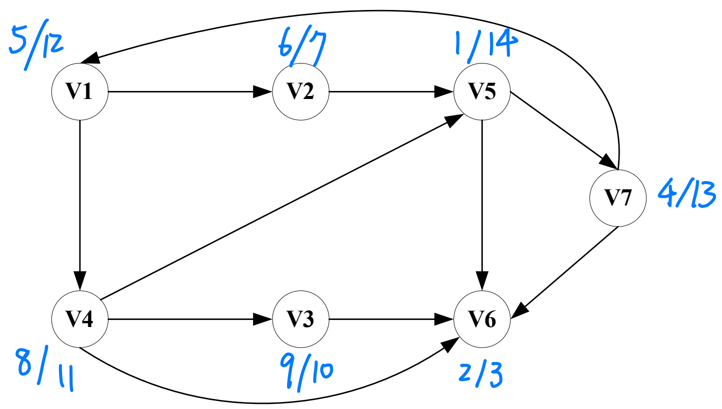
Problem 1. (20 pts)

In following sub-problems, **break ties by numerical order, if any**. (For example, choose vertex V5 first if there exist three unexplored vertices V5, V6, V7.)

(a) (10 pts) Perform a depth-first search on the following directed graph. Please start from V5. Please indicate the discovery and finishing times of each vertex.

(b) (10 pts) Continued from (a). Find strongly connected components, starting from V5.

(a)



(b) run DFS on G^T

$\{1, 2, 4, 5, 7\}$

$\{3\}$

$\{6\}$

Department of Electrical Engineering
National Taiwan University
Algorithms, Spring 2024

Problem 2. (40 pts)

One of the basic motivations behind the Minimum Spanning Tree Problem is the goal of designing a spanning network for a set of nodes with **minimum total cost**. Here we explore another type of objective: designing a spanning network for which the **most expensive edge** is as cheap as possible.

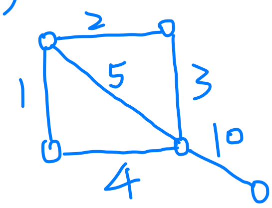
Specifically, let $G = (V, E)$ be a connected graph with n vertices, m edges, and positive edge costs that you may assume are all distinct. Let $T = (V, E')$ be a spanning tree of G; we define the **bottleneck edge** of T to be the edge of T with the greatest cost.

A spanning tree T of G is a **minimum-bottleneck spanning tree** if there is no spanning tree T' of G with a cheaper bottleneck edge.

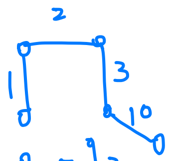
(a) (20 pts) Is every minimum-bottleneck tree of G a minimum spanning tree of G? Prove or give a counterexample.

(b) (20 pts) Is every minimum spanning tree of G a minimum-bottleneck tree of G? Prove or give a counterexample.

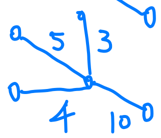
(a) False



MST :



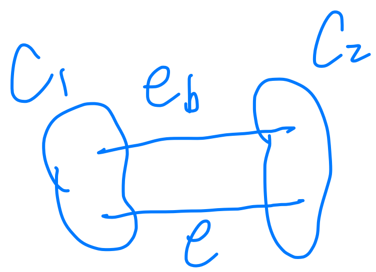
MBST :



(b) True

Assume T is a MST of G

e_b is bottleneck edge of T



Because T is MST so there is no other cheaper edge that can connect C_1 and C_2.

$\Rightarrow$ There is no other spanning tree T' of G with a cheaper bottleneck edge.

Department of Electrical Engineering
National Taiwan University
Algorithms, Spring 2024

Problem 3. (40 pts)

Each of the following problems suggests a greedy algorithm for a specified task.
Prove that the greedy algorithm is optimal, or give a counterexample to show that it is not.

(a) (20 pts)

Problem: You are given a set of n jobs. Job i starts at a fixed time s_i, ends at a fixed time e_i, and results in a profit q_i. Only one job may run at a time. The goal is to choose a subset of the available jobs to maximize total profit. Assume all start times s_i and end times e_i are distinct.

Algorithm: First, sort the jobs so that $q_1 \geq q_2 \geq \ldots \geq q_n$. Then, add each job to the schedule in turn. If job i does not conflict with any jobs scheduled so far, add it to the schedule; otherwise, discard it.

(b) (20 pts)

Problem: You are given a set of n jobs, each of which runs in unit time. Job i has an integer-valued deadline time $d_i \geq 0$ and a real-valued penalty $p_i \geq 0$. Jobs may be scheduled to start at any integer time (0, 1, 2, etc.), and only one job may run at a time. If job i completes at or before time di, then it incurs no penalty; otherwise, it incurs penalty p_i. The goal is to schedule all jobs so as to minimize the total penalty incurred.

Algorithm: Define slot k in the schedule to run from time $k-1$ to time k. First, sort the jobs so that $p_1 \geq p_2 \geq \ldots \geq p_n$. Then, add each job to the schedule in this order. When adding job i, if any time slot between 1 and d_i is available, then schedule i in the latest such slot. Otherwise, schedule job i in the latest available slot $\leq n$.

b) not optimal, Let $n=3$,
$s_1=0, f_1=2, q_1=2$
$S_2=3, f_2=5, q_2=4$
$S_3=1, f_3=4, q_3=5$

Algo will choose job 3 and profit is 5
but optimal solution is that choose job 1 and 3 and profit is 6.

(b) Optimal

Assume the Algo. is not optimal which means that there exists at least one job j_k that can replace one scheduled job j_r and reduce penalty. But in this way, penalty will increase by $\underbrace{P_r - P_k}_{\geq 0}$ $\rightarrow\times\leftarrow$ Algo is optimal.