CS/ECE 374 A ✧ Fall 2019

Fake Midterm 2

November 11, 2019

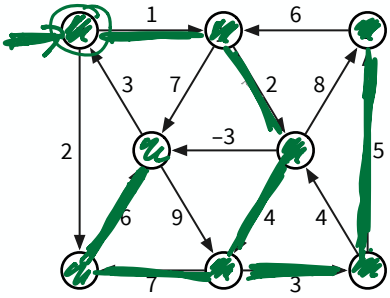
Real name:	Jeff E
NetID:	jeffe

Gradescope name:	same
Gradescope email:	

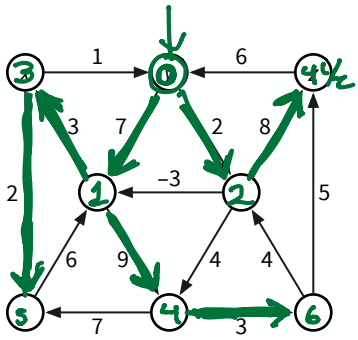
- ***Don't panic!***
- **All problems are described in more detail in a separate handout.** If any problem is unclear or ambiguous, please don't hesitate to ask us for clarification.
- If you brought anything except your writing implements, your **hand-written** double-sided 8½" × 11" cheat sheet, and your university ID, please put it away for the duration of the exam. In particular, please turn off and put away *all* medically unnecessary electronic devices.
- Please clearly print your real name, your university NetID, your Gradescope name, and your Gradescope email address in the boxes above. However, if you are using your real name and your university email address on Gradescope, you do ***not*** need to write everything twice. **We will not scan this page into Gradescope.**
- Please also print **only the name you are using on Gradescope** at the top of every page of the answer booklet, except this cover page. These are the pages we will scan into Gradescope.
- Please do not write outside the black boxes on each page; these indicate the area of the page that the scanner can actually see.
- If you run out of space for an answer, feel free to use the scratch pages at the back of the answer booklet, but **please clearly indicate where we should look**.
- Except for greedy algorithms, proofs are required for full credit if and only if we explicitly ask for them, using the word ***prove*** in bold italics.
- Please return ***all*** paper with your answer booklet: your question sheet, your cheat sheet, and all scratch paper.

CS/ECE 374 A ✧ Fall 2019 Fake Midterm 2 Problem 1	Gradescope name: Jeff E

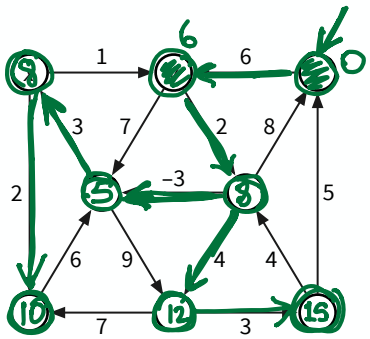
Clearly indicate the following structures in the directed graph below, or write NONE if the indicated structure does not exist. Don't be subtle; to indicate a collection of edges, draw a heavy black line along the entire length of each edge.



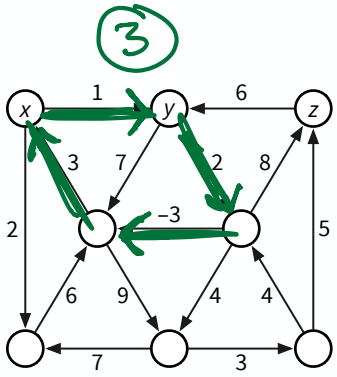
(a) A depth-first tree rooted at x.



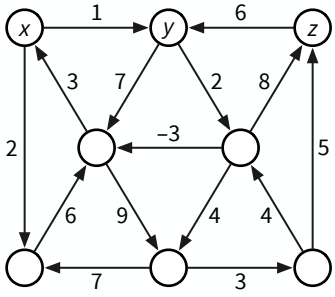
(b) A breadth-first tree rooted at y.



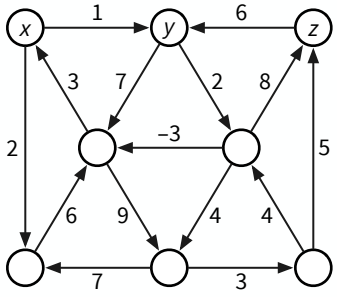
(c) A shortest-path tree rooted at z.



(d) The shortest directed cycle.



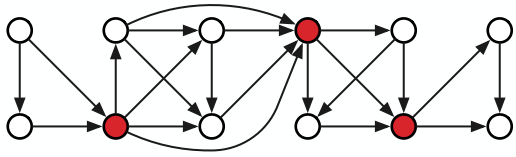
[scratch]



[scratch]

CS/ECE 374 A ✧ Fall 2019 Fake Midterm 2 Problem 2	Gradescope name:

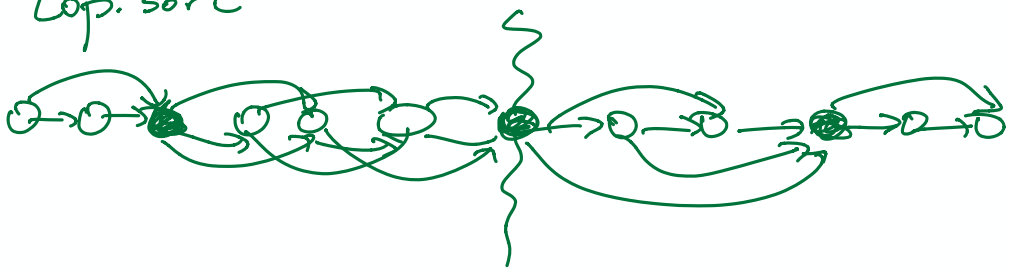
A vertex v in a (weakly) connected graph G is called a ***cut vertex*** if the subgraph $G - v$ is disconnected. For example, the following graph has three cut vertices, which are shaded in the figure.



Suppose you are given a (weakly) connected *dag* G with one source and one sink. Describe and analyze an algorithm that returns TRUE if G has a cut vertex and FALSE otherwise.

---

top. sort



cut vertex = no edge skip over in top order.

maxreach(i) = max{j | i→j edge}

↗ index in top order

For each v:
  is G\v connected?
O(V(E+V)) time

For all vertices i: maxr[i] ← i
  For all edges i→j
    maxr(i) ← max(maxr[i], j)

O(E)

mmr ← maxreach(1)
For i ← 2 to V-1
  if mmr > i
    cut[i] ← FALSE
  else
    cut[i] ← True
  mmr ← max{mmr, maxreach(i)}
O(V) time

For i ← 1 to V
  cut[i] ← True
  For h ← 1 to i-1
    if maxreach(h) > i
      cut[i] ← False

CS/ECE 374 A ✧ Fall 2019 Fake Midterm 2 Problem 3	Gradescope name:

The City Council of Sham-Poobanana needs to partition Purple Street into voting districts. A total of n people live on Purple Street, at consecutive addresses $1, 2, \ldots, n$. Each voting district must be a contiguous interval of addresses $i, i+1, \ldots, j$ for some $1 \le i < j \le n$. By law, each Purple Street address must lie in exactly one district, and the number of addresses in each district must be between k and $2k$, where k is some positive integer parameter.

Every election in Sham-Poobanana is between two rival factions: Oceania and Eurasia. A majority of the City Council are from Oceania, so they consider a district to be *good* if more than half the residents of that district voted for Oceania in the previous election. Naturally, the City Council has complete voting records for all n residents.

For example, the figure below shows a legal partition of 22 addresses into 4 good districts and 3 bad districts, where $k = 2$. Each O indicates a vote for Oceania, and each X indicates a vote for Eurasia.

Describe an algorithm to find the largest possible number of *good* districts in a legal partition. Your input consists of the integer k and a boolean array GoodVote$[1..n]$ indicating which residents previously voted for Oceania (True) or Eurasia (False). You can assume that a legal partition exists. Analyze the running time of your algorithm in terms of the parameters n and k.

---

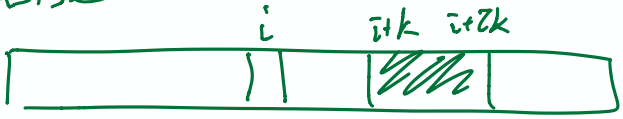
dynamic programming?

MaxGood(i) = max # of good districts for voters i thru n

$$\text{MaxGood}(i) = \begin{cases} 0 & \text{if } i > n \\ -\infty & \text{if } n - i + 1 < k \\ \max\left\{ \text{GoodDistrict}(i,j) + \text{MaxGood}(j+1) \;\middle|\; \begin{array}{l} j \ge i+k-1 \\ j \le i+2k-1 \\ j \le n \end{array} \right\} & \text{otherwise} \end{cases}$$

→ GD(i,j) = 1 iff majority of voters i..j are good.
0 otherwise

We need MaxGood(1)



n subprobs × O(k²) time (brute force) = O(nk²) time

k possible values of j × O(k) time for GD

See p. 6

CS/ECE 374 A ✧ Fall 2019 Fake Midterm 2 Problem 4	Gradescope name:

After graduation, you accept a job with Aviophiles-Я-Us, the leading traveling agency for people who love to fly. Your job is to build a system to help customers plan airplane trips from one city to another. Your customers love flying, but they absolutely despise airports. You know all the departure and arrival times of all the flights on the planet. ← Input

Suppose one of your customers wants to fly from city X to city Y. Describe an algorithm to find a sequence of flights that *minimizes the total time spent in airports*. Assume (unrealistically) that your customer can enter the starting airport immediately before the first flight leaves X, that they can leave the final airport at Y immediately after the final flight arrives at Y. → output

---

shortest path [Added after the review session]

State of the traveler = airport × time

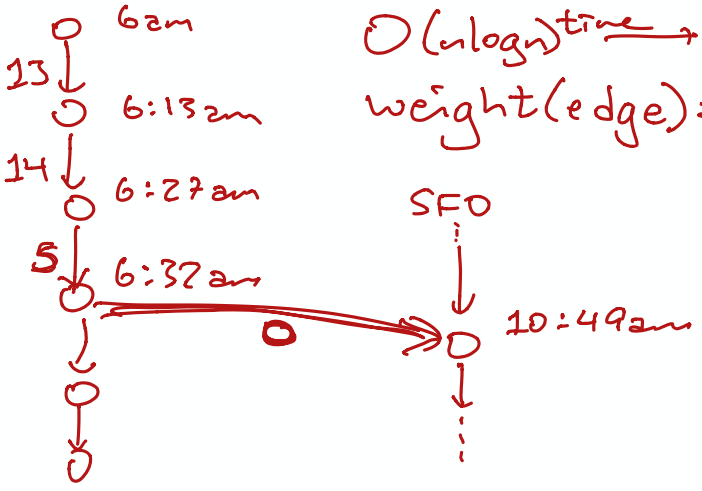
Define a directed graph $G = (V, E)$

V = set of all arrival + departure times at each airport
(airport, time) pairs (SFO, 840am)

E = Flights and stays at airports —
connect consecutive times in sorted order at each airport

$O(n \log n)$ time

weight(edge) = { time for waiting edges
0 For flying edges



add new node X = "in X parking lot"
edges X → (X, t) weight 0

Shortest path from any node (X, t) to any node (Y, t')

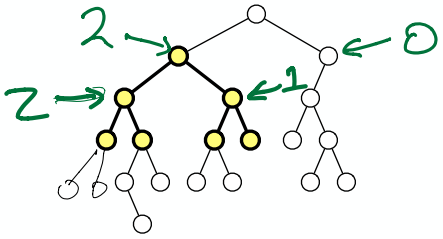
Run Dijkstra from new node X.
Time = $O(E \log V)$

see page 6

CS/ECE 374 A ✧ Fall 2019 Fake Midterm 2 Problem 5	Gradescope name:

For this problem, a *subtree* of a binary tree means any connected subgraph. A binary tree is *complete* if every internal node has two children, and every leaf has exactly the same depth.

Describe and analyze a recursive algorithm to compute the ***largest complete subtree*** of a given binary tree. Your algorithm should return both the root and the depth of this subtree. For example, given the following tree T as input, your algorithm should return the left child of the root of T and the integer 2.



Recursion

$LCS(v)$ = depth of largest complete subtree rooted at v

we need $\max_v LCS(v)$

$$LCS(v) = \begin{cases} 0 & \text{if } left(v) = NULL \text{ or } right(v) = NULL \\ 1 + \min\{LCS(left(v)), LCS(right(v))\} & \end{cases}$$

$O(v)$ time by ~~recursive~~ postorder traversal

#3 Faster?

#4 continued

n = input size

$\#\text{ vertices} = 2 \cdot \#\text{flights} = O(n)$

$\#\text{ edges} \leq 2 \cdot \#\text{vertices} = O(n)$

Running time $= O(E \log V) = \boxed{O(n \log n)}$