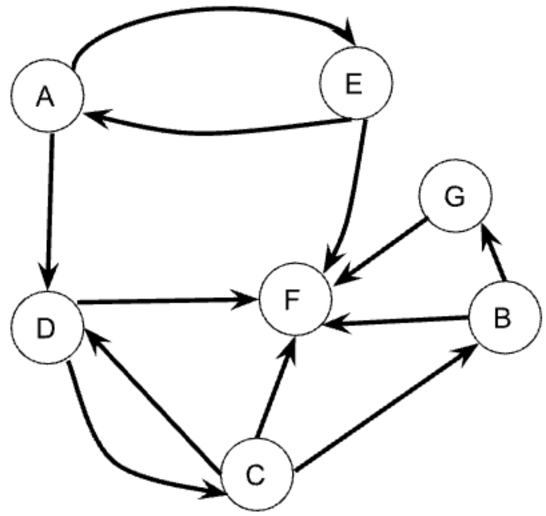
CS3000: Algorithms & Data — Summer 2025 — Laney Strange

Recitation 4
Date: June 3rd, 2025

Name:

- Recitation problems are for practice only. We'll go over the solutions during your scheduled recitation on Tuesday!
- We will provide `.tex` starter files for recitations, just as we do for homeworks. For most recitations, we encourage you to work out your solution in LaTeXto practice with typesetting.
- Collaboration is strongly encouraged during recitation!

Problem 1. *BFS*



(a) How would you represent the graph above using an adjacency matrix? (We've started the table for you.

vertex	A	B	C	D	E	F	G
A	0	0	0	1	1	0	0
B							
C							
D							
E							
F							
G							

Solution:

vertex	A	B	C	D	E	F	G
A	0	0	0	1	1	0	0
B	0	0	0	0	0	1	1
C	0	1	0	1	0	1	0
D	0	0	1	0	0	1	0
E	1	0	0	0	0	1	0
F	0	0	0	0	0	0	0
G	0	0	0	0	0	1	0

(b) In pseudocode, we use $G.A[v][u]$ to index into the adjacency matrix. For the example above, give a short pseudocode snippet that would print out all of E's neighbors.

Solution:

for each vertex $u \in G.V$
    if $G.A[E][u] == 1$
        PRINT(u)

Problem 2. *Counting Degrees*

In a directed graph, the *out-degree* of a vertex is its total outgoing edges, and the *in-degree* of a vertex is its total incoming edges. (For this problem, note that $G.adj[v]$ is the notation we use to access the adjacency list of vertex v.)

(a) Describe (in English) how you would count the out-degree of every vertex in a directed graph, given an adjacency-list representation. What would be the run-time?

Solution: For each $v \in G.V$, count the length of its adjacency list. This would take $\Theta(V + E)$ time.

(b) Give pseudocode for an algorithm that would compute the in-degree of every vertex for a directed graph, given an adjacency-list representation. Your algorithm should take G as its only parameter, and assume every vertex v has an attribute $v.indeg$.

Solution:

```
COUNT-DEG(G)
for each vertex u ∈ G.V
    u.indeg = 0
for each vertex u ∈ G.V
    for each vertex v ∈ G.adj[u]
        v.indeg = v.indeg + 1
```

Problem 3. *Sinks*

In a directed, unweighted graph, a vertex k is a "universal sink" if and only if k has in-degree $|G.V| - 1$ and out-degree 0. Can k be a universal sink if...

(a) ...for any vertex $i \neq k$, $G.A[k][i] == 1$?

Solution:

No, if k has an outgoing edge, then its out-degree is at least 1 but needs to be zero by definition.

(b) ...$G.A[k][k] == 1$?

Solution:

No, if k has a self-loop, it cannot be a universal sink.

(c) ...for all vertices $i \neq k$, $G.A[i][k] == 1$?

Solution:

Yes, if there exists an edge from all vertices to k (other than edge ferom k itself), then k would be a universal sink.

(d) ...for any vertices $i \neq k$, $G.A[i][k] == 0$?

Solution:

No, if there no edge from i to k then k's in degree can't be $|G.V| - 1$ so it can't be a universal sink.