CS 61BL Summer 2019 — Lab 16 — July 30, 2019

Name: SID:

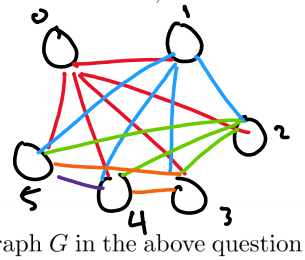
Please complete this worksheet during your lab, and turn it in to your TA by the end of your section. You are encouraged to work with your neighbors collaboratively.

Section Number: (01) (02) (03) (04) (05) (06) (07) (08) (09) (10) (11) (12)

1 Edge vs. Vertex Counts

1.1 Suppose that G is a directed graph with N vertices. What's the maximum number of edges that G can have? Assume that a vertex cannot have an edge pointing to itself, and that for each vertex u and v, there is at most one edge (u, v).

- ○ N
- ○ N^2
- ● $N(N-1)$
- ○ $\frac{N(N-1)}{2}$



$5+4+3+2+1$

↓ generalize to N

$(N-1)+(N-2)+\ldots+1 = \frac{N(N-1)}{2} = O(N^2)$

1.2 Now suppose the same graph G in the above question is an undirected graph. Again assume that no vertex is adjacent to itself, and at most one edge connects any pair of vertices. What's the maximum number of edges that G can have compared to the directed graph of G?

- ● half as many edges
- ○ the same number of edges
- ○ twice as many edges

undirected ⇒ every edge is 2 directed edges ⇒ $N(N-1)$

same logic but reverse

1.3 What's the minimum number of edges that a connected undirected graph with N vertices can have?

- ● $N-1$
- ○ N
- ○ N^2
- ○ $N(N-1)$
- ○ $\frac{N(N-1)}{2}$

↑ need all nodes to connect to something, no isolated nodes → at least $N-1$ connections

2 Trade Offs

2.1 Space

1. Which is most **space-efficient** if you have a lot of edges in your graph?
   - (•) Adjacency matrix
   - ( ) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

2. Which is most **space-efficient** if you have very few edges in your graph?
   - ( ) Adjacency matrix
   - (•) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

3. Which is most **time-efficient** for adding an edge if you have a lot of edges in your graph?
   - (•) Adjacency matrix
   - ( ) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

4. Which is most **time-efficient** for adding an edge if you have very few edges in your graph?
   - (•) Adjacency matrix
   - ( ) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

5. Which is most **time-efficient** for returning a list of edges from one node if you have very few edges in your graph?
   - ( ) Adjacency matrix
   - (•) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

6. Which is most **time-efficient** for returning a list of edges from one node if you have a lot of edges in your graph?
   - ( ) Adjacency matrix
   - (•) Adjacency lists
   - ( ) It depends
   - ( ) They are the same

2.2 Runtime

1. Using an adjacency matrix, how long in the worst case does it take to determine if vertex v is adjacent to vertex w? (Assume vertices are represented by integers.)

   - (●) constant time
   - ( ) time proportional to the number of neighbors of vertex v
   - ( ) time proportional to the number of vertices in the graph
   - ( ) time proportional to the number of edges in the graph

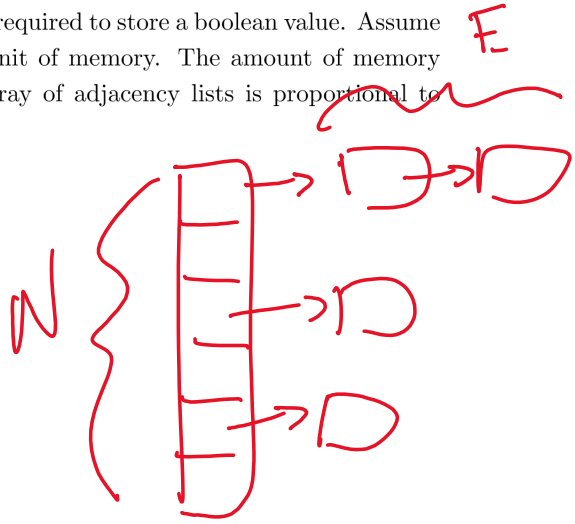
2. Using an array of adjacency lists, how long in the worst case does it take to determine if vertex v is adjacent to vertex w? (Assume vertices are represented by integers.)

   - ( ) constant time
   - (●) time proportional to the number of neighbors of vertex v
   - ( ) time proportional to the number of vertices in the graph
   - ( ) time proportional to the number of edges in the graph

3 Memory Use

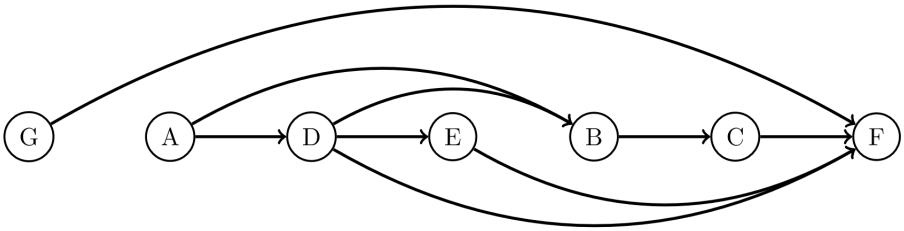
3.1 Suppose we are representing a graph with N vertices and E edges. There are N^2 booleans stored in an adjacency matrix, so the memory required to store an adjacency matrix is N^2 times the memory required to store a boolean value. Assume that references and integers each use 1 unit of memory. The amount of memory required to represent the graph as an array of adjacency lists is proportional to what?

- ( ) NE
- ( ) E^2
- (●) $N + E$
- ( ) E



4 Topological Sorting

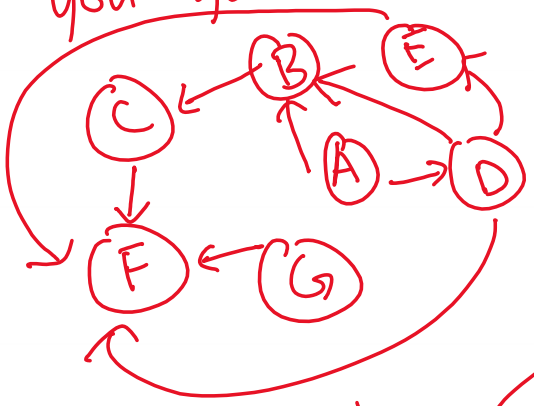
4.1 Give a valid topological sort of the graph below. For your reference, some orderings of the graph are provided below the graph.




```
DFS preorder: ABCFDE (G)
DFS postorder: FCBEDA (G)
BFS: ABDCEF (G)
```

GADEBCF

- normally you get it in some random form



⇒ Convert to graph above to get topo sorted version (arrows all going in 1 dir)

- many answers as long as only arrows pointing right