CS594/690 Homework 2

Spring 2025

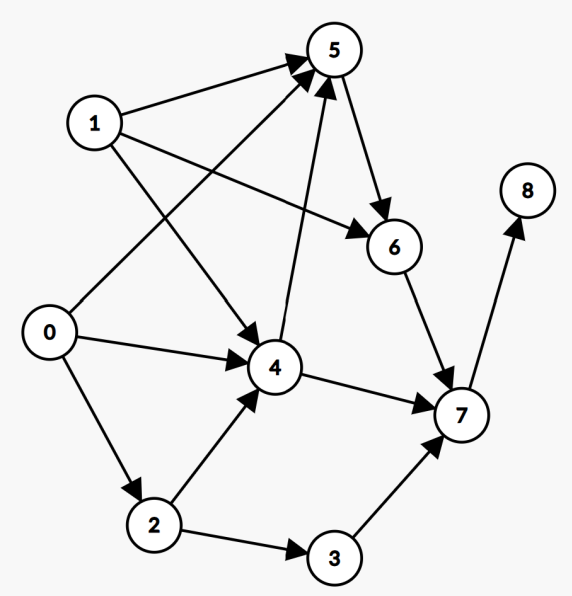
January 29, 2025

(Due 4:10pm, February 5, 2025)

Email homework assignments to ldojcsak@vols.utk.edu by the beginning of class time.

1. Let $\lambda(G)$ denote the edge-connectivity of a graph G. Show that if $\lambda(G) = k \geq 2$, then the deletion of k edges from G results in a graph with at most 2 components. Is there a similar result for vertex-connectivity?

2. Topological sort orders the vertices of a directed graph such that for every directed edge (u, v) from vertex u to vertex v, u comes before v in the ordering. Given the following graph, state the linear ordering of vertices by performing a topological sort using the conditions below. In every case where a choice needs to be made, choose the smallest vertex label.



   a. Using DFS
   b. Using BFS
   c. Which method has a better time complexity? Why?

3. Consider the following recursive algorithm. It takes as input a simple, undirected graph G and returns an output of True/False if the graph is connected.

   a. Does this algorithm work correctly for every simple, undirected graph of size $n > 0$?

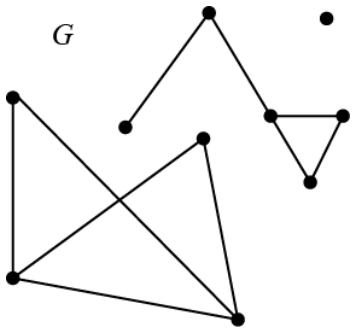
   b. On which graphs does this algorithm reach the final return statement?

   c. On what type(s) of graph(s) does the worst-case behavior occur? What about the best?

```
Input: Adjacency Matrix A[0...n-1, 0...n-1]
Output: True/False

define Connected(A[0...n-1, 0...n-1]):
    if n=1:
        return True
    else:
        if not Connected(A[0...n-2, 0...n-2]):
            return False
        else:
            for x=0 to x=n-2:
                if A[n-1][x] = 1:
                    return True
            return False
```

4. Write a program that takes as input a simple, undirected graph in modified DIMACS format and outputs the number of connected components. Input graphs will be labeled using 0-indexing. You may use any programming language, as long as it is supported on the EECS Linux machines. Codes should be documented with comments describing the algorithm implemented. Include a short README text file with instructions on how to invoke your program. The following example graph G should output the number 3:



Wikibook topics that may help you with this assignment:

- DFS
- BFS
- Topological sorting
- Connected components
- Edge connectivity
- Vertex connectivity
- Algorithms for 2-edge-connected components