CSCI 651 - Applied Graph Theory
Exam - Written and Coding Questions

November 14, 2021

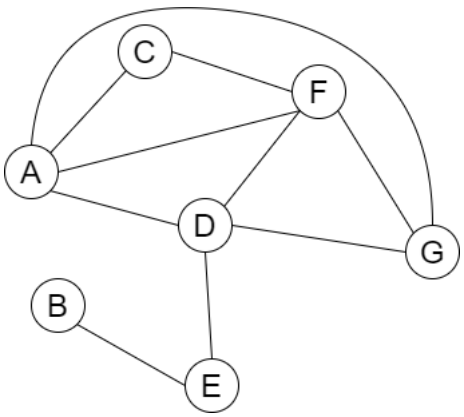
Solutions to the written portion of the exam along with relevant figures should be submitted via PDF to Blackboard. Make sure to **justify your answers**. Code should be written in a Python file called `exam.py` and submitted on Blackboard as well. The exam will close on **December 5th at 11:59 pm**. Note that the full exam is out of 100 points but there are 107 points available. These additional 7 points are extra credit.

You must complete solutions to these problems **individually**. You may ask and answer questions on Discord or Blackboard meant to clarify the problems but you may not discuss or post solutions. You are allowed to use your notes, material posted on Blackboard, NetworkX, matplotlib, and scikit-learn documentation, and the textbook (*Network Science*) during the exam. However, you **are not permitted** to use any other resources. This includes Google, Stack Overflow, and all other websites and documents available online.

Good luck!

Problems

1.



(a) (1 pts) Produce the adjacency list of this graph.

(b) (1 pts) Produce the adjacency matrix of this graph.

(c) (1 pts) Determine the clustering coefficient of each vertex.

(d) (1 pts) Determine the average degree of the graph.

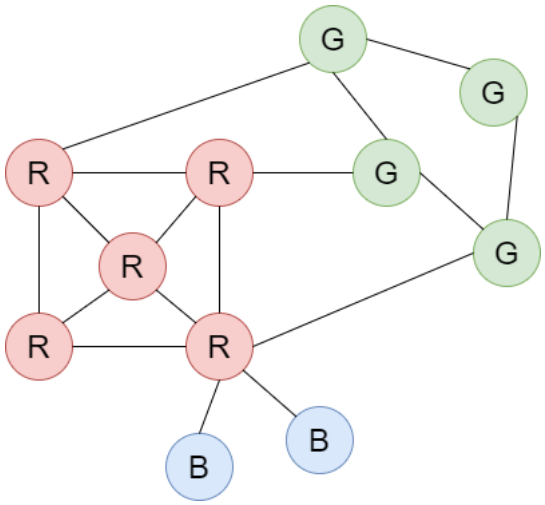
2. (a) (1 pts) Draw the graph associated with the adjacency matrix below.

$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

(b) (1 pts) Draw the graph associated with the adjacency list below.

```
1: [2]
2: [1, 3, 5, 6]
3: [2, 3, 5]
4: [7]
5: [2, 3, 5, 6]
6: [2, 5]
7: [4]
```

3. (4 pts) Determine the modularity of the graph with given communities below.



4. (10 pts) Let $n = 1000$, $m_0 = 5$, and $m = 3$. Using these parameters and starting with a complete graph, generate 20 Barabási-Albert random graphs. For each, collect the betweenness centrality of all nodes. Create a histogram showing this information as a distribution (show a density).

Collect the same information for nodes in 20 Erdös-Rényi random graphs $G_{1000,p}$ where $p = \frac{2m}{n-1}$. Show the distribution of betweenness centralities for nodes in these graphs on the same set of axes as for the Barabási-Albert graphs.

Don't forget to include a legend and appropriate labels in the figure. Collecting all of this information may take several minutes.

Consider the resulting figures and distributions carefully. What do you observe? Are there any differences between these distributions? If so, do these differences make sense given what you know about the Barabási-Albert model and Erdös-Rényi random graphs?

5. (10 pts) Consider the power grid network. Should we be surprised by the distribution of distances in this graph? Explain your reasoning.

6. (10 pts) Write a function `randomWalk(G, v, n, p, q)` that generates and returns a second-order random walk starting from the node v with parameters p and q as defined in *node2vec: Scalable Feature Learning for Networks* of length n. Include a small test to show that the function works as expected.

7. (20 pts) Use Node2Vec to generate an embedding of Zachary's Karate Club designed to identify community structure. Use k-means clustering to partition nodes into communities based on this embedding. Evaluate the quality of the communities you find based on the ground truth. Recall that this network is available through NetworkX with `nx.karate_club_graph()`. Calling this network `G`, the "ground truth" community of node v is available in `G.nodes(data=True)[v]['club']`.

   This is a fairly open-ended question. Here are a few things to think about.

   - What parameters should you use for Node2Vec? It might be worth trying out a few combinations.
   - Without knowing the number of communities in Zachary's Karate Club, what value should you choose for k? Again, it may be worth trying a few.
   - How should you measure the quality of communities? How should you pick the best k and the best set of communities? To start, consider using modularity as your metric.
   - Only compare the communities you find to the ground truth once at the very end when you are happy with the modularity value you achieve. Use normalized mutual information (you may use the function available in scikit-learn) to do this. This is meant to help avoid overfitting!