MATH 255	Graph Theory Spring '18	Lab 3

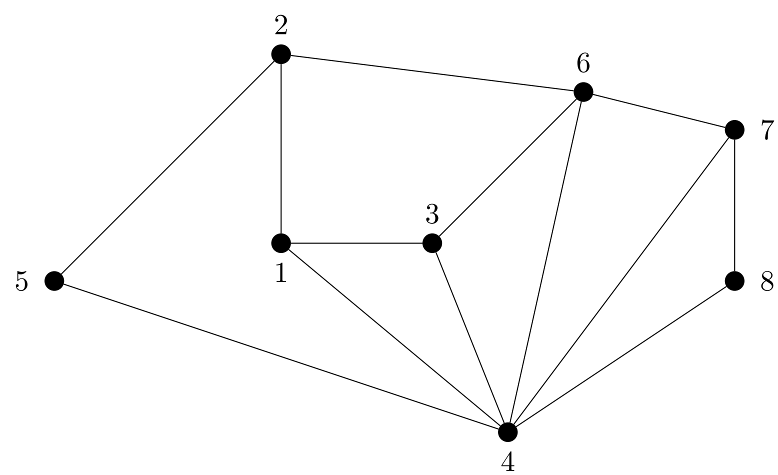
Random Walks, Rich-Get-Richer

Pair Names: ______________________________

1 Uniform Random Walks

A *Random Walk* in G starts from an initial *current node*, and builds a walk according to a random procedure. In each time step a neighbor of the current node is chosen at random, and this neighbor becomes the next node in the walk (and becomes the new "current node").

For example, a *5-step random walk* traverses 5 consecutive edges in the graph before stopping. In the graph below, for initial node 2, one possible random walk of length 5 is $(2, 1, 3, 6, 2, 5)$.
A different possible random walk of length 5 starting from node 2 is $(2, 6, 7, 4, 3, 4)$.



We say a random walk is *Uniform* if every neighbor of the current node x has an equal chance of being selected as the next node in the walk (aka the probability distribution from which the next node is chosen is uniform over the neighbors of the current node).

An important implicit assumption is that the *random choices that define the walk are independent*. That is, the next step of the walk depends only on the *current node*, and not at all on what nodes preceded the current node.

1. The Matlab command `randi(8)` produces a uniform random integer between 1 and 8. Use this command to select a random initial node.

2. The initial node you chose has some number of neighbors (e.g. node 5 has two neighbors), so use `randi` to choose one of those neighbors uniformly at random.

   For example, if node 5 is the current node, and has neighbors $\{2, 4\}$, I might consider node 2 to be the first neighbor of node 5, and node 4 to be the 2nd neighbor of node 5.

   Using `randi`, construct 2 random walks of length 10 starting from a random initial node:

   Report the final destination of your 2 random walks to the professor!

3. Now you will create function with inputs `A`, `steps`, and `x` (where `x` is the initial node) that starts a random walk of length `steps` from node `x` in the graph with adjacency matrix `A`.

   Carefully comment on what the lines of the following code are doing:

```
function [destination] = randwalk(A,steps,x)

order=length(A);
current=x;

for time=1:steps
    neighcurrent=[];
    for node=1:order
        if (A(current, node)==1)
            neighcurrent=[neighcurrent, node];
        end
    end
    numneighbors=length(neighcurrent);
    chosen=randi(numneighbors);
    current=neighcurrent(chosen);
end

destination=current;
```

4. **Make an adjacency matrix for the graph on page 1.**
   Run `randwalk` on this graph starting from node 2 several times for a 20-step random walk.

   Record the final destination for each trial you conduct:

5. Consider the two possible random walks of length 5 mentioned earlier: $(2, 1, 3, 6, 2, 5)$, and $(2, 6, 7, 4, 3, 4)$. **Find the exact probability of each of these walks being realized**.

   Hint: recall that each step is an independent event.

6. Suppose that your initial node is 2. What is the **exact probability** that after a random walk of length 2, the final destination will be 4?

7. **(Distribution of Destinations)** For the graph given on page 1, write a script that builds 100 random walks each of length 20. Each random walk should choose a random initial node, and use `randwalk` from above to arrive at a destination after 20 steps. Your script should record the final destination of each random walk.

   Print your script, with comments about the lines of code,
   and print a histogram of final destinations reached for the random 100 walks.

8. Fix a particular initial node of your choice, and create a histogram of destinations for 100 random walks of length 20 as in 7. **Print your histogram, and comment in 1-2 sentences about how it compares** to what you got when each initial node was chosen randomly as in 7.

9. **(Distribution of Visits)** How often is each node visited on a long random walk?

   Again, we'll work with the graph on page 1.

   Generalizing the code from `randwalk`, create a script that prints a histogram showing the number of times each node is visited on a random walk of length 1000.

   Try running this for a few different choices of initial node.
   Does this seem to make a difference in the final histogram?

   Print your script, with comments about the lines of code,
   and print a histogram showing how often each node was visited.

10. **Reflection:** Carefully consider the evidence you've gathered in 7, 8, and 9. Write a paragraph about how your observations seem to relate to G, including any conjectures you have.

2 Rich Get Richer

Now you'll consider a new style of random graph in which nodes gradually join a network forming links to the most popular existing nodes. You'll build an adjacency matrix gradually.

The graph will have a total of 200 nodes. First create a core of 10 early nodes, $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ where every node is connected to every other node.

Then add the remaining 190 nodes to the graph one at a time. When node i joins, i will

form edges to **exactly 8** randomly-chosen nodes in $\{1, 2, 3, ..., i-1\}$, and

i **will be strictly more likely to form edges to nodes of high degree than to low-degree nodes**

i has a non-zero chance of connecting to each node in $\{1, 2, 3, ..., i-1\}$

1. Create a **script that meets these three requirements** *(there are many possibilities!)*. Creating this code may take some significant creative thought. **Write comments about what your code is doing** and **explain in writing why you CAN BE SURE** that all three requirements hold. **Print your script.**

2. Print a histogram of the degrees for your Rich-Get-Richer Random Graph.

3. Repeat exercises 7 and 9 from the previous section for your 200 node graph, **printing the histograms in both cases. Write 2-3 sentences discussing each of the histograms** and how they relate to the degree distribution you observed above for your rich-get-richer random graph.

3 Reading/Writing

Download and read the first two pages of this research paper:

`http://research.microsoft.com/en-us/um/people/borgs/papers/fit.pdf`

Don't worry if you can't follow all of the details/citations: concentrate on the connections to the random graph that you just constructed. **Write three thoughts or questions** you have based on either this article or Lab 3 more generally.

By the way, one of the "static" random graph models mentioned in the introduction of this article is the $G(n, p)$ *that you met in Lab 1.*