CSE 6010 Assignment 3: Graph Analytics

Due Dates:

- Part 1 due: 10:00 AM, Friday, October 23, 2015
- Part 1 Revision Due (optional): 11:59 PM, Monday October 26, 2015
- Part 2 due: 10:00 AM, Friday, November 6, 2015
- Part 2 Revision Due (optional): 11:59 PM, Monday November 9, 2015
- No late submissions will be accepted

1. Background

This assignment involves generating synthetic network graphs and performing analyses on them. Graphs arise in many applications in science, engineering, and the social sciences. A graph consists of a set of vertices (nodes) and edges (links) where each edge connects two vertices. Here, we assume undirected edges and there are no edges from a vertex to itself. The number of edges attached to a vertex is referred to as the vertex's *degree*.

The first part of this assignment involves analyzing random graphs. The second part considers a class of graphs known as scale-free networks. You will develop code to answer some specific questions concerning these graphs.

You will work in teams of two persons each to attack this problem. For each team one person will write a program to generate a network graph and write it into a file as well as conduct a modest literature search. The second person will write a program that reads the graph file and perform an analysis of the graph. The two programs should be completely separate "stand-alone" programs. The interface between the two programs will be the format of the file specifying the graph. You and your partner must define and document a common file format for the graph.

2. Part 1: Random Graph Generation and Analysis

Consider an undirected graph. A connected component (or just component) is defined as a subgraph of the original graph where there is a path between every pair of vertices in the subgraph, and there are no edges between a vertex in the component and any other vertex not in the component. If the graph represented a communication network with vertices representing computers and edges representing communication links, computers in a component can communicate with each other, but a computer in the component cannot communicate with any other computer not in the component. Here, we are interested in the size of the largest component of a graph, i.e., the maximum number of vertices of any component in the graph.

We are specifically interested in random graphs constructed using the Erdős-Rényi model (first introduced by Gilbert in 1959). This model has two parameters: N is the number of vertices in the graph, and P is the probability there is a link between any two vertices in the graph. P is the same for all pairs of vertices in the graph. It is clear that if P is 0.0, the graph contains N unconnected vertices, and the largest component of this graph has size 1. Further, if P is 1.0, there is an edge between every pair of vertices, and the largest component is the entire graph, so the largest component is of size N. For the first part of this assignment your goal is to examine how the size of the largest component of random graphs changes as P is increased from 0.0 to 1.0.

There two pieces to the first part of the assignment: (1) write a graph generation program and conduct a literature search to determine what result you should expect from this analysis, and (2) write a program to analyze a graph in order to determine the size of the largest component. One person in your team should take responsibility for each part.

Graph generation and literature search. The graph generation program should take as a command line input (1) an integer N that indicates the number of vertices in the graph, (2) the probability parameter P ($0<P<1$), and (3) the name of the file that will hold the resulting graph. For example:

```
% graphgen1 100 0.5 topology
```

will execute your program called `graphgen1` and produce a graph with 100 vertices and probability parameter 0.5 and store the graph into a file called "topology." Hint: you may find it convenient to use the C function `sscanf()` to extract a value from a string.

In addition, conduct a literature search to determine what results you should expect in the graph analysis portion of this assignment.

Graph analysis. The graph analysis program reads the graph file produced by the graph generation program and computes the size of the largest component. To perform this computation you should use a *breadth first search algorithm* to identify components in the graph, and determine the size of the largest one. You will need a FIFO queue to implement the breadth first search. You may reuse or adapt software developed earlier in the semester (software developed by your partner or included in the solution provided by the TA is fine) to implement the FIFO queue, but be sure to cite the source of this software.

You should write your analysis to be computationally efficient. For example, a brute force approach is to invoke your breadth-first search function on every node of the graph. You should be able to do something more efficient!

Report. The report for this part of the assignment should include the results of your literature survey, and plot the size of the largest component for different values of P. Use as large a value of N as your programs can process in a reasonable amount of time. Compare the results observed by your program with the expected results derived from your literature search.

In addition to documenting your software you will need to define the format of the file containing the graph. This file format defines the interface between the graph generation and the graph analysis programs. You should also include the results of some test runs demonstrating your programs produce correct results. You should include some test cases so the TA can test if your program executes correctly. Be sure to include proper bibliographic citations where appropriate.

3. Part 2: Scale Free Network Generation and Analysis

Here, we are interested in a class of graphs known as *scale-free networks*. An important characteristic of a scale-free network is they include some number of *hub* nodes that contain a relatively large number of edges, but the vast majority of nodes contain relatively few edges. Technically, a scale-free network is defined as a network whose node degree distribution follows a power-law, however, you need not be concerned with this aspect to complete this assignment. Scale free networks are interesting because there is empirical evidence that they arise in many applications of current interest.

Here, we are interested in the Barábasi-Albert model, also called the preferential attachment model to construct a scale-free network. Each edge of the resulting graph should be assigned a weight drawn from a random variable that is uniformly distributed between 0.0 and 1.0.

We are interested in understanding how the path length between vertices grows as the size of the scale-free network grows. The metric of interest here is called the network *diameter*. Diameter is defined as follows. A path from a source to a destination vertex is a sequence of edges leading from the source to the destination. The length of this path is the sum of the weights assigned to the edges making up the path. The *distance* between any two vertices in a graph is the length of the *shortest* path between those two vertices. The diameter of the graph is defined as the maximum distance among all pairs of vertices in the graph. The objective of this part of the assignment is to understand how the graph diameter grows as N is increased in a scale free network with N vertices.

The two parts of this assignment are: (1) conduct a literature search to learn how to construct a scale-free graph and write a program to construct one, and (2) write a program to analyze a graph produced by the graph generation program to determine the graph's diameter. One person in your team should take responsibility for each part. The person that wrote the graph generation for the first part of the assignment must write the graph diameter program, and the person who wrote the graph analysis program in the first part must write the graph generation program and conduct the literature survey in the second part.

Graph generation and literature search. The graph generation program should take as a command line input (1) an integer N that indicates the number of vertices in the graph, (2) the name of the file that will hold the resulting graph. For example:

```
% graphgen2 100 topology
```

will execute your program called `graphgen2` and produce a scale free graph with 100 vertices and store the graph into a file called "topology." In addition to specifying the vertices and edges, the file should also specify the edge weights.

The network topology is constructed using a well-known principle called *preferential attachment*. The key idea behind preferential attachment is when a vertex is added to an existing network, it is more likely to establish an edge to another vertex that already has a high degree compared to vertices with low degree. This construction mechanism is also referred to as "the rich get richer" model. Your literature search should locate sources that define more precisely how to create a scale-free network and you should also identify at least one real-world application where scale-free networks arise.

To construct a scale network we recommend you start with a core with a small number (e.g., 5) of vertices where each vertex has an edge to every other vertex (i.e., a fully connected network), and then add new vertices one by one, where each new vertex adds an edge to another vertex that is already part of the network. Note that this guarantees the network consists of a single connected component, i.e., there is a path between every pair of vertices in the graph.

Graph analysis. The graph analysis program reads the graph produced by the graph generation program and computes the diameter of the graph. To compute the diameter, use Dijkstra's algorithm to compute the minimum length path between a source vertex s and every other vertex in the network. Try to be as efficient as possible. You will need a priority queue. You may use

the one you or your partner developed in the previous assignment, or one provided in the solution set for that assignment but be sure to cite the source of this software.

Report. The report for this part of the assignment should include the results of your literature survey, describe the graph generation algorithm, and plot the size of the network diameter for different sized networks. Use as large a value of N as your programs can process in a reasonable amount of time (e.g., a few minutes). Are your results consistent with information you found in your literature search concerning scale-free networks? Explain your results. Your report should also briefly describe at least one application where scale-free networks arise in practice by specifying what the nodes and links represent, and what kinds of questions a graph analysis might answer.

To complete this part, turn in (1) all source code for your programs, (2) example test files demonstrating your program executes correctly, and (3) a brief report documenting your results verifying both correctness, and reporting and explaining the results you obtained. You should include some test cases so the TA can test if your program executes correctly. Be sure to include proper bibliographic citations where appropriate.