CAS CS 112 – Spring 2011, Assignment 6
Due at 10:00 pm on Tuesday, May 3

Part I: Problems 4.1.9, 4.1.12, 4.2.2 from the text (7 pts each).

Part II (79 pts): In this assignment you have to investigate the structure of a real world graph using some of the techniques that we discussed in class. The graph that you will use is a co-authorship graph from the DBLP database that contains information about articles published in a large number of computer science journals and conferences:

`http://www.informatik.uni-trier.de/~ley/db/`.

In a co-authorship graph, each vertex in the graph represents an author and an edge is present between two authors if they have published at least one paper together. Your task is to use simple techniques to analyze this graph. You will try to find the "importance" of an author based on a number of different measures including number of co-authors, neighborhood characteristics, and its "centrality" in this graph:

`http://en.wikipedia.org/wiki/Eigenvector_centrality#eigenvector_centrality`.

Building the Graph

(20 Points) You will be provided with a file (**graph.txt**) which contains the graph data using the format from the text described on p. 431. Each vertex in the graph is represented as an integer number between 1 and 326186, some of which refer to authors of single-authored papers (no incident edges). There are 1.2 million edges in the graph. Each line in this file contains two numbers, separated by a tab, that represents an edge between these two vertices. For example, the first five lines are:

1 2
1 43
1 60

and this means that there are three edges between vertex 1 and vertices 2, 43 and 60. The edges are undirected. Therefore, if an edge (a, b) is specified, so is (b,a). Since the graph is very sparse, use an adjacency list representation to store it. Your first task is to read in the **graph.txt** file line by line and add the corresponding edge to the graph representation. (Hint: do NOT start with this huge graph. Do everything on much smaller graphs from the booksite, like mediumG.txt.)

Ranking Vertices by Degree

(20 Points) The simplest way to determine the importance of a vertex is by its degree. For example, in a social interaction network the person who knows the most people may be important (but of course that is not always the case). Using our graph representation, it is very easy to determine the degree of each vertex (just get the size of its adjacency list). First, output a list of vertices sorted by decreasing degree, where each line contains the name

of the vertex and its degree. You should notice that most vertices have very low degree, but there are still a considerable number of hub vertices with high degree.

Ranking Vertices by Closeness Centrality

(20 Points) A second measure that signifies the "importance" of a vertex in a graph is the **closeness centrality** of this vertex. The idea behind this measure is that if a vertex is close to the center of the graph, it should be easy to reach any other vertex in the graph. Given a graph $G = (V, E)$ that is connected (that is, there is at least one path between every pair of vertices), the closeness centrality CLC of a vertex v is defined as follows:

$$CLC(v) = \frac{\sum_{u \in V'} d(v,u)}{|V|-1}$$

where $V' = V - \{v\}$ and $d(v, u)$ is the shortest distance between vertices v and u in the graph (the size of the shortest path). Thus, the $CLC(v)$ is the average distance of vertex v to every other vertex in the graph.

To calculate the $CLC(v)$ of a vertex v you can run a breadth-first search from v and keep a counter of the number of vertices with distance 1, 2, 3, and so on. After all vertices of the graph have been visited, you can use these counters to compute $CLC(v)$.

Compute the closeness centrality of the top-100 vertices ranked by degree and output those as a sorted list (sorted by increasing closeness centrality) as well.

Ranking Vertices by Clustering Coefficient

(20 Points) Another way to determine the importance of a vertex is by considering how dense is the immediate neighborhood of this vertex. One way to measure this is by considering what fraction of its neighbors are connected. This metric is known as the **clustering coefficient**. Formally, for undirected graph $G = (V; E)$, define the neighbor set of $v_i \in V$, denoted by N_{v_i} , to be the set of vertices connected to v_i:

$$N_{v_i} = \{v_j | (v_i, v_j) \in E\}.$$

Then the clustering coefficient of v_i, denoted by $CC(v_i)$, is the fraction of vertex pairs in N_{v_i} that share an edge:

$$CC(v_i) = \frac{|\{(v_j, v_k)\}|}{\binom{|N_{v_i}|}{2}} : v_j, v_k \in N_{v_i}, (v_j, v_k) \in E.$$

Compute the clustering coefficient of the top-100 vertices ranked by degree, and again output a sorted list (sorted by decreasing clustering coefficient).