2006

Exploring the topology of small-world networks

Min Hu

Department of Computer Science

Rochester Institute of Technology

Master's Project Report

Exploring the Topology of Small-World Networks

Student: Min Hu

Committee Members

Chairman: Prof. Stanisław P. Radziszowski

Reader: Prof. Roger S. Gaborski

Observer: Dr. Christopher M. Homan

Jan 5, 2004

Table of Contents

Exploring the Topology of Small-World Networks

Abstract

Complex networks have been studied for a long time in order to understand various real-world complex systems around us. Complex systems, such as the WWW, the movie-actor network, social networks and neural networks, are systems made of many non-identical elements connected by diverse interactions. The study of the network topology is one of important issues on the way of exploring such systems, because the structure always affects the system function. Traditionally, these systems have been modeled as either completely ordered graphs or completely random graphs. Until recently, some surprising empirical results in the field of complex networks, like 19 clicks of the web's diameter and 6 degrees of separation in social networks, show us the small-world phenomena existing in some large sparse networks. This finding motivates the interest in small-world networks. The objective of the project is to study the properties of small-world networks and the network evolution over time via experiments on a movie actor collaboration network; to find their different characteristics by comparing small-world networks with random networks; and to analyze the factors that result in such differences. The properties of small-world networks discussed here include small diameter, sparseness, clustering, giant component, power-law degree distribution and short path discovery. Also, four existing network models are studied in this project: Watts-Strogatz Small-world model, Erdös Rényi Random-graph model, A.-L. Barabási Scale-free model and Jon Kleinberg Small-world model.

1. Background

1.1 Introduction

Complex networks are ubiquitous in nature and society. The Internet is a network of routers and computers connected by physical links. The World Wide Web is a network of billions of web pages connected by hyperlinks. The nervous system is a network of nerve cells connected by axons. The cell is a network of chemical molecules linked by chemical reactions. The organization is a network of people linked by social interactions. Although we know about their existence, it's so hard for people to understand the topology of these networks due to their large size and inherent complexity. Intuitively, these systems seem to be disordered. Thus, these systems were traditionally thought of as random network models. But, the recent discovery of the small-world phenomena greatly changed the way in which people looked at these complex systems. The small-world phenomenon was studied as a social category, when Stanley Milgram, the social psychologist at Yale University in the US, declared the known claim "six degrees of separation"[10] in 1967. It means that one person can contact anybody else in this world by only six intermediate acquaintances on average. That is, we live in a "small world". The small-world phenomenon is not just one exception occurring in the social network. Surprisingly, it is proved that this feature also exists in several large sparse networks like movie-actor networks, www and the neural network of the nematode worm C. elegans. Is it possible that the small-world phenomenon is common to many sparse networks? Under what conditions can a network be the small-world network? If a system can be constructed as a small-world network, what does this imply about the dynamic behavior of this system? The project focuses on investigations of these questions. This includes doing experiments in the real movie-actor network, studying four proposed models (Watts-Strogatz Small-world model [5], Erdös Rényi Random-graph model [10], A.-L. Barabási Scale-free model [10] and Jon Kleinberg Small-world model [13]), analyzing empirical results against theoretic predictions in models, and investigating the properties of small-world networks such as small diameter, sparseness, clustering, giant component, power-law degree distribution and short path discovery.

1.2 Basic graph concepts

A **bipartite graph** *G* is a pair *(V, E)*, where the vertex set *V* is partitioned into two nonempty sets V_1 and V_2, and every edge in the edge set *E* joins a vertex in V_1 to a vertex in V_2.

A **complete graph** *G* is a pair *(V, E)*, where $|E| = \binom{|V|}{2}$; that is, any two vertices are adjacent.

A **sparse graph** *G* is a pair *(V, E)*, where $|E| = O(|V|)$.

The **distance** $D(i,j)$ is the minimum number of edges traversed from the vertex *i* to vertex *j*, $i, j \in V$.

The **characteristic path length** *L* of a graph $G(V,E)$ is the mean of all shortest path lengths among all pairs of vertices, $L = \sum_{i,j \in V} D(i,j) \Big/ \binom{|V|}{2}$.

The **diameter** of a graph *G*, $Diam(G)$, is the maximal distance between any pair of vertices in *G*. $Diam(G) = max_{(i,j)} D(i,j)$

The **neighborhood** of a vertex *v*, Γ_v, is a set of those vertices that are directly connected to *v*. $\Gamma_v = \{x | D(v,x) = 1\}$ $(v \notin \Gamma_v)$

The **local clustering coefficient**, C_v, is $C_v = |E(\Gamma_v)| \Big/ \binom{|\Gamma_v|}{2}$, where $|E()|$ is the number of edges in the induced subgraph whose vertex set is Γ_v, and $|\Gamma_v|$ is the number of elements in Γ_v.

The **clustering coefficient** *C* of a graph $G(V,E)$ is the average of C_v over all *v*.

$$C = \left(\sum_{v \in V} C_v \right) \Big/ |V|$$

The **degree** of a vertex *v*, d_v, is the number of edges incident on it. $d_v = |\Gamma_v|$

The **degree distribution** of a graph *G(V,E)* is the distribution of the discrete random variable, X_k, such that $P(X_{k=}r)$ denotes the probability that the number of nodes with degree $k \in \{1,2,...,|V|\}$ is equal to r, $r \in \{0,1,2,...,|V|-1\}$.

The **giant component** of a graph $G(V,E)$ is a component with $\succ |V|/2$ vertices.

A **random graph** $G(n,p)$ is a graph, in which connections between vertices are determined in a random way by the edge probability p and the number of nodes n, $|E| = \binom{n}{2} * p$.

2. Project Description

2.1 Properties of Small-world Networks

The large-scale networks, which fall in the small-world network set, have three significant properties in common: sparseness, small diameter and clustering. This project verifies these three properties by investigating the movie-actor network. Other two properties, threshold of the giant cluster appearance and degree distribution, are studied as well. The latter two properties are thought to be related to the network dynamics. With growth of the network over time, how they change is an interesting and important question hidden in the class of small-world networks. Besides them, this project discussed the search for short paths in Kleinberg model. The explanation of these properties is given as follows:

- Sparseness

  In a small-world network, the graph with n nodes has the number of edges closer to $O(n)$ than to $O(n^2)$. So, comparing with a complete graph with the same number of nodes, the small-world graph has much smaller number of edges.

- Small diameter

  A random graph with a large connection probability usually has a small diameter. Also, this characteristic is observed in small-world networks. Shortcuts in a small-world graph result in its small diameter. The diameters in small-world graphs increase logarithmically with the number of nodes.

- Clustering

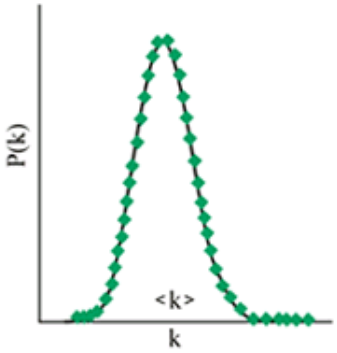
  Clustering coefficient *C* is the measure of how much a graph is clustered. In a small-world graph, there is much higher clustering coefficient than in a random graph.

- Threshold of the giant cluster appearance

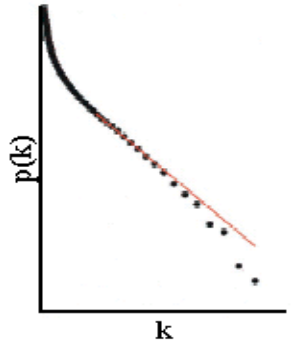
  The giant component emerges when the number of edges is greater than *(n/2)*[3] in random graphs. We will see if this property also appears in small-world graphs.

- Scale-free power-law degree distribution

  The degree distribution measures the network connectivity. According to the random graph theory, the degree distribution should be a Poisson distribution shown in Figure 1 a. For the Poisson distribution, *P(k)* is strongly peaked at $k = \langle k \rangle$, where $\langle k \rangle$ is a mean degree that depends on the edge probability *p*; then exponentially decays as *k* tends to be very large [10]. But, an unexpected degree distribution was found in small-world graphs. It is a power-law degree distribution shown in Figure 1 b. A power-law distribution shows that most vertices have similar low degree, but a small number of vertices have much higher degree than in random graphs [2].



a. Poisson Distribution [10]



b. Power-law Distribution [2]

Figure 1. : Two different kinds of degree distribution

- Short path discovery in Kleinberg model

  There exist short paths linking arbitrary pairs of nodes in the small-world networks. Jon Kleinberg model was designed to find short chains between nodes. This model puts each node in a metric space. It applies one greedy algorithm, relying on the geographical information on nodes, to find the best intermediate nodes that are closer to the target node.

2.2 Experiment environment

In this project, the study of small-world networks is closely related to understanding one real system, movie-actor collaboration network. The movie-actor network is selected as one example of complex systems based on each of the following requirements:

1. Be a large and sparse network
2. Possible to obtain the complete movie-actor database [11]
3. Network evolution over time

The movie-actor network is constructed as a unipartite graph in this way: there is only one type of vertex set in the graph: actors. Two corresponding actors are connected by an edge if they appear together in a movie.

This movie-actor database is released by Internet Movie Database Ltd (IMDb). IMDb is an international organization whose objective is to provide useful and up to date movie information freely available on-line. It currently covers over 150,000 movies with over 2,100,000 filmography entries and is expanding continuously. The database includes filmographies for actors, country of production, year of release, directors, writers, editors and other aspects. There are many *.list* files that provide the comprehensive movie information from ftp://ftp.imdb.com/pub/interfaces/. In this project, I just focus on the following aspects of movie-actor information: actor, country and year of release. So these three files were downloaded for this project: *actor.list, movies.list* and *countries.list*. The last modified time of these files obtained for this project is in May 2003.

2.3 Applied Models

A model, which takes the form of an algorithm to generate new graphs reflecting some properties of a real-world network, is an abstraction of a real network. So, studying the models proposed by other people is a good way to understand those properties of networks. In this project, four models are studied as follows:

- Erdös-Rényi random graph model [10]

A sample graph of the model is shown in Figure 2 [10].

Algorithm for generating sample networks:

Input: (n, p) where n is the number of vertices, and p ($0 \le p \le 1$) is the edge probability.

a) Start with n isolated nodes

b) Connect each pair of nodes with the same probability p.

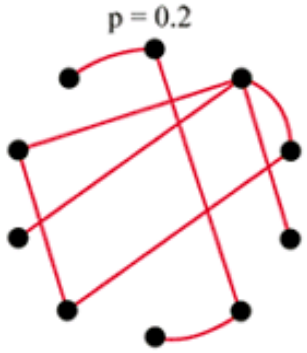


Figure 2. ([10]): This network is modeled as an Erdös-Rényi random-graph. It has $N = 10$ and $p = 0.2$. With these parameters, it is expected to have (and it has) 9 edges.

We can simulate networks for this model in two ways: Given a fixed-value n, generate different networks by changing the values of p; or setting p as some constant, just change the value of n to get different networks.

- Watts-Strogatz Small-world model [8]

A sample graph generated by this model is shown in Figure 3 [5]. Figure 4 [5] shows two important statistics in the small-world graphs.

Algorithm for generating sample networks:

Input: (n, k, p) where n is the number of vertices, k is the distance in which each vertex is connected initially to its neighbors by undirected edges, and p ($0 \le p \le 1$) is the probability of rewiring each edge.

a) Start with a ring lattice with n nodes, each has the *kth* nearest neighbors; that is, $V = \{0,1,\ldots,n-1\}$, $(i, j) \in E$ *iff* $(i \neq j) \cap (|i - j| \bmod n \le k)$. Thus, the degree of each vertex is $2k$ and the ring has (nk) edges.

b) Replace original edges by random ones based on the probability p. The following process is repeated for each vertex until all original edges are scanned: Pick up each vertex v from the ring in the clockwise direction; determine whether its edge connected to the *1st, 2nd,* …, or *kth* nearest neighbor in the clockwise direction still wasn't redrawn; if so, detach this edge and make a new edge from vertex v to a randomly chosen vertex w that is not directly connected to v before.

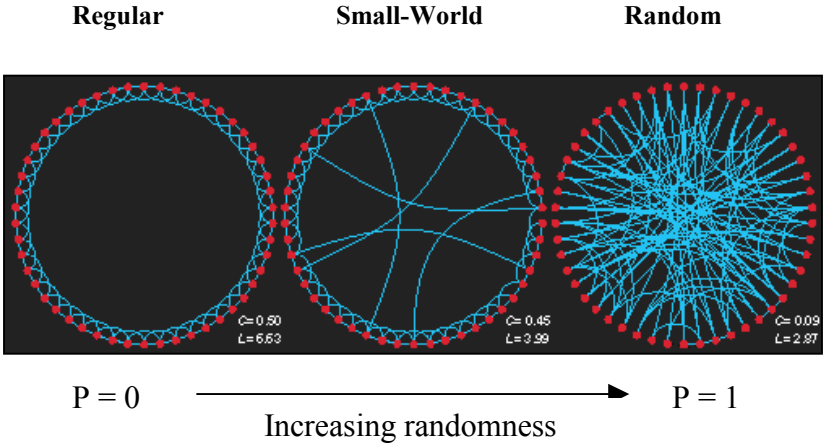


Figure 3([5]): Progression from regularity to randomness

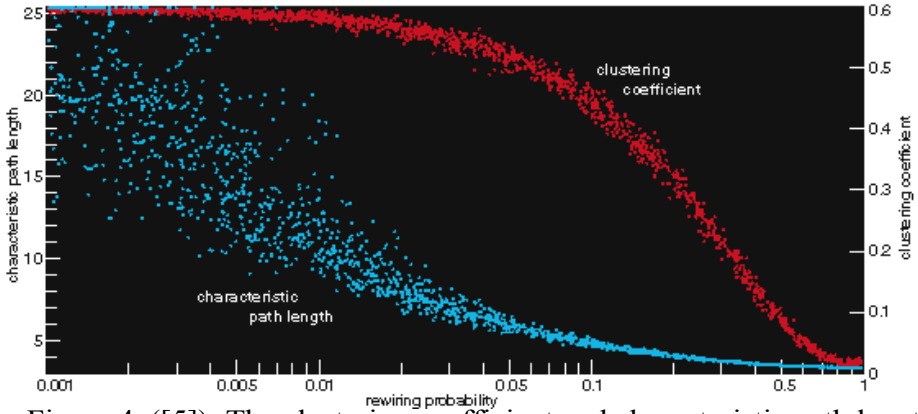


Figure 4. ([5]): The clustering coefficient and characteristic path length decline with p increasing.

We can simulate networks for this model in three ways: a) With n and k as fixed-values, change the value of p to get different networks. This method allows us to see the role of p in different networks. b) In order to understand how k works, just change the value of k to obtain networks by setting n and p as some constant. c) setting k and p as some constant, change the value of n. This helps us analyze the network evolution.

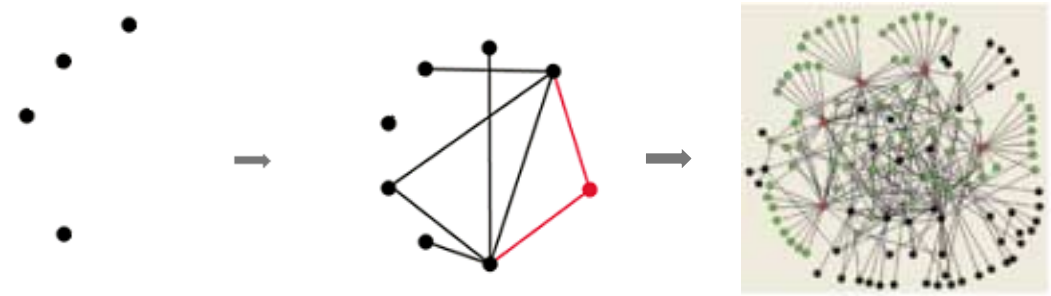
- A.-L. Barabási Scale-free model [1]

This model (an example shown in Figure 5 [10]) modified two attributes of the above two models. a) The number of vertices is fixed. b) The same probability p is applied to make new edges.

This scale-free model assumes that the network grows over time by addition of new nodes and new edges. When a new node is added to a scale-free graph, new edges are created from this node to some existing nodes. The probability that an existing vertex will receive one of such new edges is proportional to the current degree of the vertex. Hence, the more connectivity that a vertex already has, the higher probability that a vertex gains even more new edges, a phenomenon called preferential attachment.
Algorithm for generating sample networks:

Input: (n_0, m, t) where n_0 is the initial number of vertices, m ($m \le n_0$) is the number of added edges every time one new vertex is added to the graph, and t is the number of iterations.

a) Starting with n_0 isolated nodes.
b) Every time we add one new node v, m edges will be linked to the existing nodes from v. Each existing node u_i has a different probability $p(d_i)$ to receive the connection from v, such that $p(d_i) = d_i / \sum_j d_j$ [1]. Eventually, the graph has (n_0+t) nodes and (mt) edges.



(a) (b) (c)

Figure 5. ([10]): Scale-free graphs. (a) Start with 4 isolated nodes. (b) The 4th new node is added in the network. Two new edges were created from this new node to two existing nodes with more connections. (c) After the t^{th} iteration as (b), eventually we got such a scale-free graph.

We can simulate networks for this model in two ways: a) Make networks be in different sizes by changing the value of t and keeping n_0 and m constant. b) Make networks be in the same size by changing all three parameters.

- Jon Kleinberg Small-world model [13]

As we know, there exist short paths linking together arbitrary pairs of nodes in the small-world networks. But, one question comes up: how can we find these short paths. This model tries to answer this question. It uses ideas based on geographical intuition. The framework of this model can be put into a metric space, i.e. where distance is defined.

Algorithm for generating sample networks:

Input: *(k, n, p, q, r) where k* is the number of dimensions; n is the number of nodes in one dimension (that is, a grid of n^k points); p is lattice distance for constructing local connections; q ($q \geq 0$) is the number of long-range contacts for each node; and r ($r \geq 0$) is the clustering exponent that measures the distribution of long-range edges.

a) Begin with a set of nodes which are identified by geographical coordinates in the k-dimensional space, $\{(x_1, x_2, x_3, \ldots x_k): x_i \in \{1, 2, \ldots, n\}$ for $1 \leq i \leq k\}$

Create local-range edges based on a constant $p \geq 1$. The node u has a directed edge to every other node within lattice distance p. The lattice distance is defined as the number of steps separating two nodes: $d(u, v) = d((x_1, x_2, x_3, \ldots x_k), (y_1, y_2, y_3, \ldots y_k)) = |x_1 - y_1| + |x_2 - y_2| + \ldots + |x_k - y_k|$. It is also possible to use the Euclidean distance $d(u,v) = d((x_1, x_2, x_3, \ldots x_k), (y_1, y_2, y_3, \ldots y_k)) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_k - y_k)^2}$

b) Create long-range edges based on $q \geq 0$ and $r \geq 0$. Each node u is connected to q nodes v as the long-range contacts according to the probability that is $d(u,v)^{-r} / \sum_{v \neq u} d(u,v)^{-r}$ ($\sum_{v \neq u} d(u,v)^{-r}$ is the normalizing constant). Long-range edges are directed.

We can simulate networks for this model in two ways: a) generate different whole networks by passing the different values to parameters *k, n, p, q* and *r* in order to analyze common properties in other models. b) Obtains a part of a network instead of a whole network for searching the short path between nodes in this model by applying a decentralized algorithm with different values to parameters *k, n, p, q* and *r*.

2.4 The Evolution of Small-world Networks

It is known that most of large-scale real systems, which belong to the small-world network set, dynamically grow up over time. However, it is only hypothesized whether these underlying characteristics, which determine the topologies of networks, have impact on the dynamics of these networks. If so, how does it act? In this project, these questions are investigated. Since the study of the evolution, which includes variety of mechanisms like preferential attachment, local events, growth constraints [1] and so on, seems to be very complicated, it's almost impossible to cover all of them. This project concentrates only on some of them: In particular, I extracted the sub-networks from the entire movie-actor network according to consecutive years (or decades) of movies,

compare the empirical data of network properties against the models described in section 2.3.

3. Implementation

3.1 Software tools

The following software tools were implemented in java1.4:

1) A parser program to extract data from the movie-actor database to construct a compact network structure.

This parser program is called *ExtractMovieActorDB*. It reads one or several files about movies and actors from www.imdb.com to generate actor network data. The extracted files included *actors.list*, *movies.list* and *countries.list*.

Each line in *actors.list* corresponds to one actor. The data format:

actor name -> movie name (year) [optional info] -> movie name (year) [optional info] -> …

Each line in *movies.list* corresponds to one movie. The data format:

movie name (year) -> year

Each line in *countries.list* corresponds to one movie. The data format:

movie name (year) [optional info] -> country

The parser encodes each actor and each movie in a unique number. It generates a compact graph format and writes data into a new file. This parser can generate three kinds of graph files at one time: movie-actor bipartite graph, actor unipartite graph, and movie unipartite graph. Their format is: each line represents the connectivity of one node. The first column is the node's ID; the rest of columns are other nodes' IDs, to which this node connects. This program can generate different sub-graphs by parameter controls on

released movie year or countries by which movies were issued. The option parameters include:

Usage: ExtractMovieActorDB [-di] [-am] [-maxY <maximum extracted year>] [-minY <minimum extracted year>] [-c <country limit>] [-mc <maximum movie #>] [-ac <maximum actor #>] [-out <output file name>]

-di: control dropping isolated nodes. By default, drop isolated nodes without this option.
-am: control adding the movies, which appear in *actors.list* but not in *movies.list* or *countries.list* to the movie set. By default, add those movies without this option.
-maxY < maximum extracted year >: set the maximum extracted movie year.
-minY <minimum extracted year>: set the minimum extracted movie year.
-c <country>: only extract the movies issued by the country.
-mc <maximum movie #>: set the maximum number of the movie set.
-ac <maximum actor #>: set the maximum number of the actor set.
-out <output file name>: set the output file name. By default, the name is *outputDB*.

2) A topology generator to create networks according to the proposed models.

This generator includes the implementation of four models: Erdös-Rényi random graph model, Watts-Strogatz Small-world model, A. -L. Barabási Scale-free model and geographical Small-world model by Jon Kleinberg. Each model can generate different graphs by construction parameter controls including the number of nodes, the number of edges, probabilities and so on. The generator will output each generated graph into a file in the following format:

 # parameter lines
node_1 linked_ node_1 linked_ node_2 …
node_2 linked_node_1 linked_node_2 …
…

a) The class *RandomGraph* implements the Erdös-Rényi random graph model. It includes options as follows:

Usage: RandomGraph <the number of nodes> <probability> [output file name]

b) The class *SmallWorldGraph* implements the Watts-Strogatz Small-world model. It includes options as follows:

Usage: SmallWorldGraph <the number of nodes> <neighbor distance> <rewiring probability> [output file name]

c) The class *ScaleFreeGraph* implements the Barabási Scale-free model. It includes options as follows:

Usage: ScaleFreeGraph <the number of initial nodes> <increased edge # at every time> <repeated times> [output file name]

d) The class *GeographicalGraph* implements the model proposed by Jon Kleinberg. It includes options as follows:

Usage: GeographicalGraph <k> <n > <p > <q > <r > [output file name]

k: represents dimensions; only 2 or 3 dimensions are supported.
n: the number of nodes in one dimension.
p: lattice distance for constructing local connections.
q: the number of long-range contacts for each node.
r: clustering exponent. It measures the density of long-range connections.
An alteration of the above class, *GeographicalGraphInEuclidean*, uses Euclidean distances to link two nodes instead of lattice distances. Assume that points are randomly placed on a sphere with the radius of 1.Two random variables are defined to generate one random point: the latitude (theta) and the longitude (phi), x=cos(theta)cos(phi), y=cos(theta)sin(phi), z=sin(theta).
Usage: GeographicalGraphInEuclidean <dimensions> <the number of nodes> <neighbor distance rate (%)> <long-range contacts> <clustering exponent> [output file name]

3) An analyzer program to analyze properties based on the modeling topologies and the actual network structure.

This analyzer program called *Analyzer* reads the specific files created by the program 1) and 2). It calculates the statistics of each given graph, such as the number of edges, a graph diameter, clustering coefficient, degree distribution and the size of a giant component. It includes options as follows:

Usage: Analyzer <input file name> [-di] [-sd <n pairs of nodes>]

-di: indicate that the graph is a directed graph. By default the graph is undirected.
-sd <n pairs of nodes>: indicate that the graph diameter is calculated by sampling distances between n pairs of nodes. These n pairs of nodes are selected randomly from the giant component. In the section 3.3, n is set to 20 in various models for calculating sampled diameters. If this option is not set, the distances between all pairs of nodes are calculated. This option is very useful for large graphs with 10000+ nodes since for such sizes computing exact diameter becomes too expensive.

3.2 Algorithms

The following algorithms were applied in this project for analyzing the properties of networks.

1) Depth-first search algorithm

   To check whether a graph is connected or what the size of its giant component is, we can use the DFS algorithm. The DFS algorithm is implemented in non-recursive way in Java as follows:

```
DfsGraph (int rootNode,  Vector visitedList)
    int visitedCount = 0
    create an empty stack for storing those nodes waiting for scanning.
    stack ← rootNode   //initially put the root node to the stack.

    while ( stack is not empty )
       do   u ← pop up the top node from stack
            if ( u was not visited )
            then   put the node u to the visited list.
                   visitedCount ← visitedCount + 1
                   nlist  ← getNeighbors(u)    // get neighbors of the node u
                   for  each v  ∈  nlist
```

```
                    do if ( v was not visited )
                        then stack ← v // put v into stack for scanning later

    return visitedCount
```

2) Breadth-first search algorithm

The BFS algorithm is applied to calculate the distance of any pair of nodes. Thus we can get the diameter of a graph. The implementation of this algorithm is as follows:

```
bfsGraph(int root, int dest)
    color = int[nodesNum]  // indicate the status of each node
    d = int[nodesNum]     // storing the distances from the root node to other nodes
    p = int[nodesNum]    // storing the parent node of a node
    Vector queue    // the waiting list for being visited

    // initialization
    for (each node i ∈ the graph vertices )
       do if (i is not equal to root)
            then color[i] ← WHITE_COLOR;  // mark as not visited yet
                 d[i] ← the maximum integer
                 p[i] ← -1
    color[root] ← GRAY_COLOR   // mark as the waiting node for being visited.
    d[root]  ← 0
    p[root] ← -1
    put the node root into queue

    while ( queue has some element  )
       do u ← remove the first element from queue
          nlist ← getNeighbors(u)
          for  ( each v ∈ nlist )
             do  if ( color[v] is not WHITE_COLOR )
                   then  color[v] ← GRAY_COLOR  // mark as being visited
                         d[v]  ← d[u] + 1
                         if ( v is equal to dest )
                         then  return d
                         p[v] ←  u
                         add the node v to queue
          color[u] ←   BLACK_COLOR   // marked as visited already
return d
```

3) Decentralized algorithm

This decentralized algorithm is addressed by Jon Kleinberg [13]. It is applied on a network generated according to Jon Kleinberg small-world model. The decentralized algorithm is used to find a short path between one arbitrary pair of nodes operating with purely local information. The mechanism is that in each step, the current message holder

u chooses a contact that is as close to the target *t* as possible according to lattice distance. The current message holder has knowledge of

(i) the set of local contacts among all nodes

(ii) the location of the target

(iii) the locations and long-range contacts that have come in contact with the message.

We can even let the algorithm make use of less information than is allowed. Assume that the current message holder doesn't know (iii). The *expected delivery time* is the primary measure for such an algorithm, which represents the expected number of steps needed to forward a message from a random source to a random target. The implementation of this algorithm is as follows:

```
decentralizedAlg(int src, int des )
    bestNode ← src
    Vector path   // store all of intermedial nodes including the start node
    add bestNode to path

    while ( bestNode is not equal to des )
        do  nlist ← getNeighbors(bestNode)
            minDis ←    the maximum integer
            for ( each v ∈ nlist )
               do distance ←    get the lattice distance from  des to v
                  if ( distance  is less than minDis )
                  then   minDis ←    distance
                         bestNode ←     v
                         if ( distance is equal to 0 )
                         then jump out of the "for" loop
             add bestNode to path
    return path
```

4. Experiment

4.1 Movie-Actor graphs

Extracted movie-actor sub-graphs from different criteria like size, time and location to see if these graphs still keep the same properties and how close the measured data for properties is.

Case 1: Varying the number of actor nodes

Actor nodes	Edges	Nodes in the giant component	Diameter	Clustering coefficient	Degree sequences
100000	904340	83476	4.5	0.62677	am-a
50000	213144	35255	4.9	0.48730	am-b
10000	8081	3062	6.4	0.15718	am-c

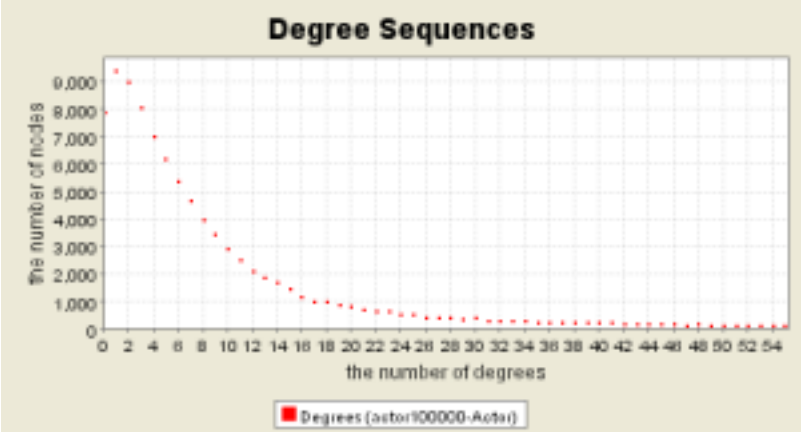


Fig. am-a (1)

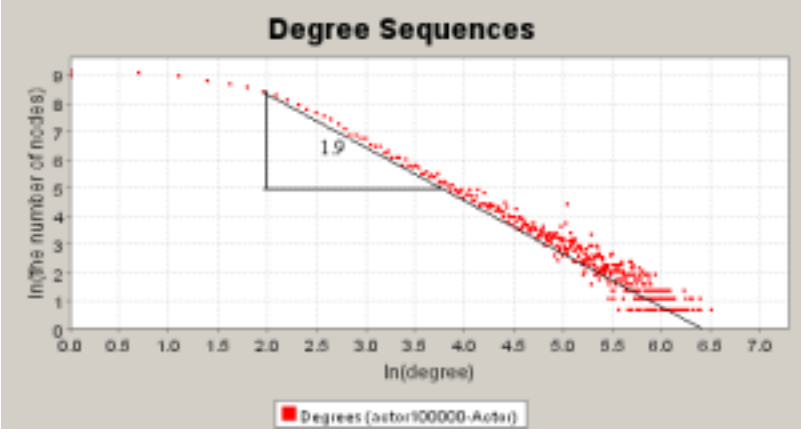


Fig. am-a (2)
Log-log plot of degree sequence. This plot shows that there is power law degree distribution with the exponent $r \approx 1.9$ decay.

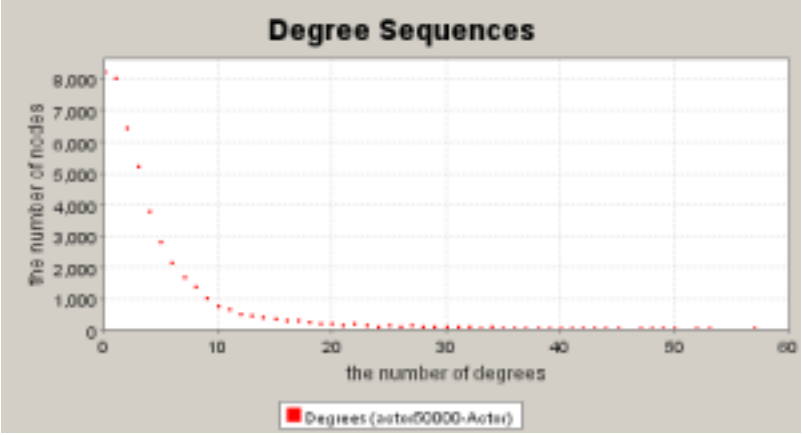


Fig. am-b

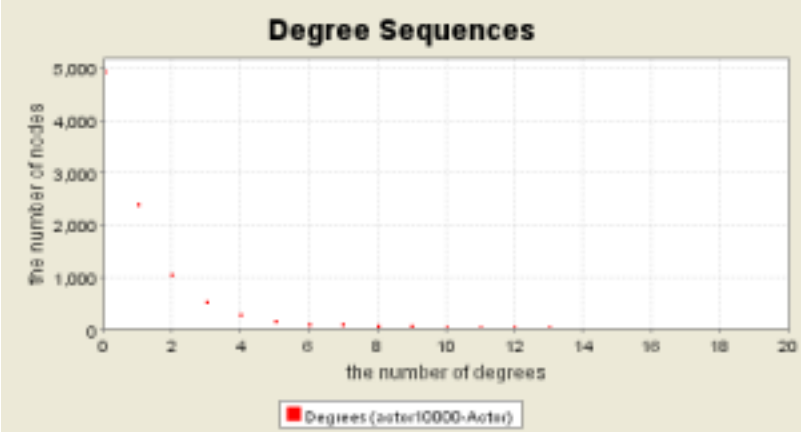


Fig. am-c

Case 2: Varying time

Although the following graphs are retrieved from different periods, the measured values for properties in each graph come out very close.

Years	Nodes	Edges	Nodes in the giant component	Diameter	Clustering coefficient	Degree sequences
1970s	76892	1868118	73100	3.5	0.76236	am-d
1980s	116286	2914598	111440	4.2	0.78199	am-e
1990s	179868	4638204	170343	?	0.79073	am-f
1990	26213	376060	23396	4.8	0.87033	
1991	25577	349787	22674	4.9	0.86889	
1992	27999	447901	24521	5.3	0.87572	
1993	29037	418654	25441	5.6	0.87239	
1994	32003	465337	28026	4.9	0.87205	
1995	33329	500940	29260	5.1	0.86769	
1996	35307	497457	30764	4.9	0.86524	
1997	37917	577386	33062	5.3	0.86560	
1998	39926	626864	34579	4.6	0.866885	
1999	41873	596297	36093	4.8	0.862576	

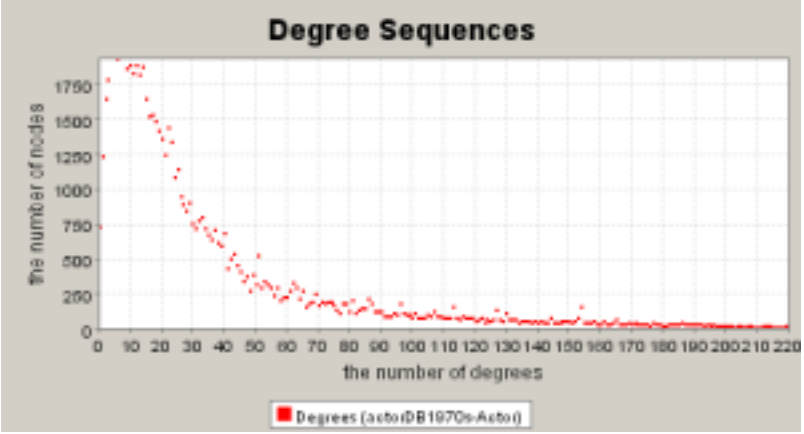


Fig. am-d

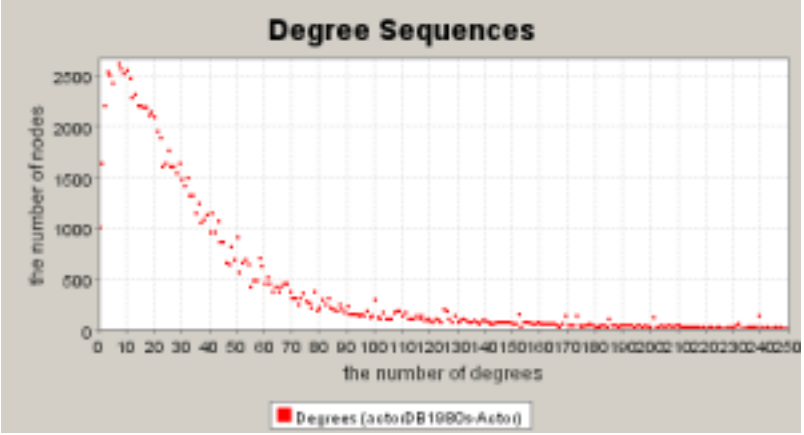


Fig. am-e

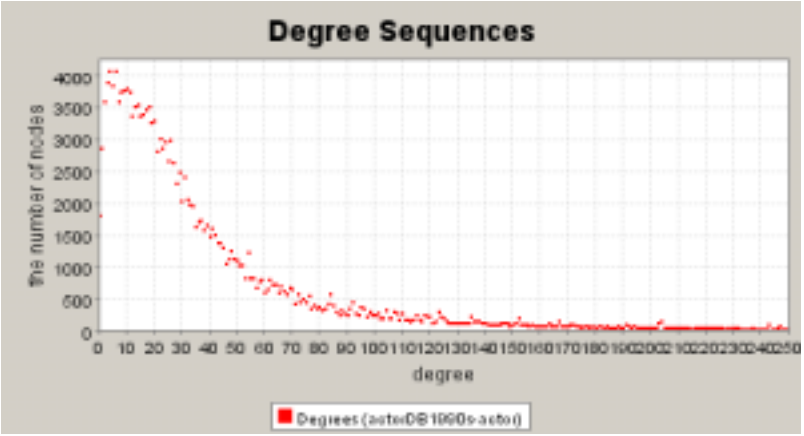


Fig. am-f

Case 3: Varying countries

Country	Nodes	Edges	Nodes in the giant component	Diameter	Clustering coefficient	Degree sequences
France	54807	1436436	52831	3.5	0.83004	am-g
India	5272	74714	4884	3.5	0.77392	am-h
Italy	31787	722225	30790	3.8	0.81522	am-i

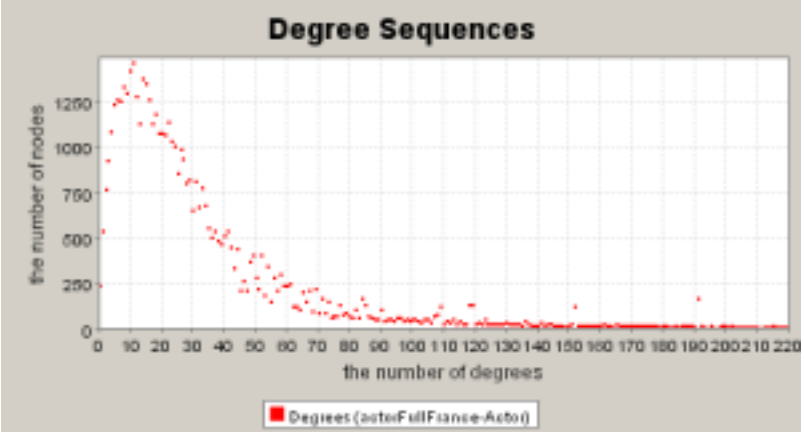


Fig. am-g

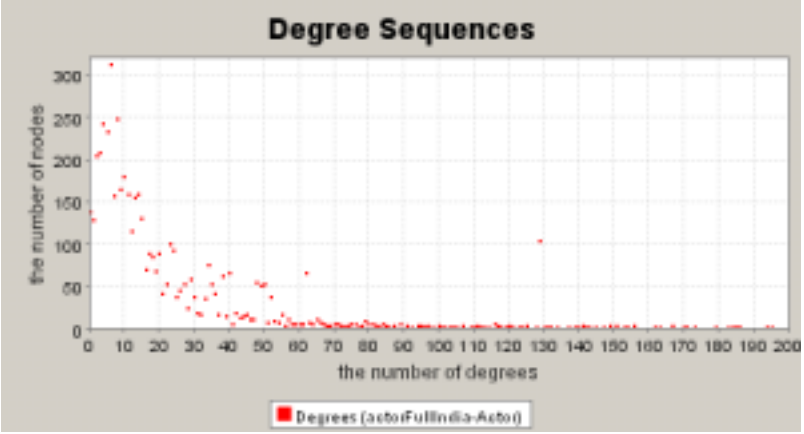


Fig. am-h

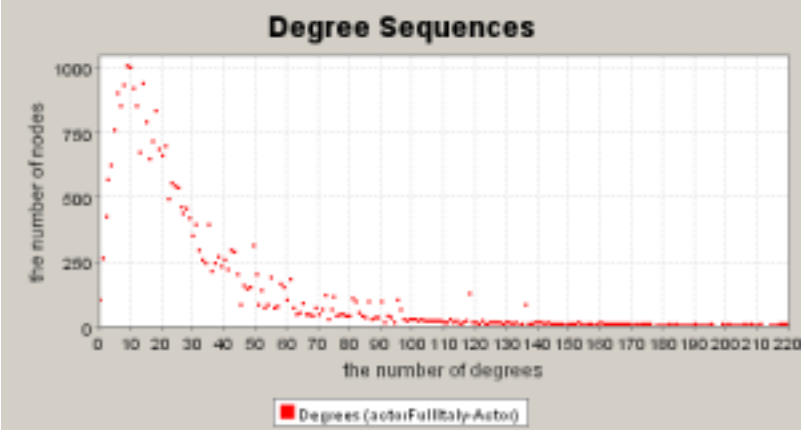


Fig. am-i

The above figures of degree sequences show that movie-actor networks have a power-law distribution in network connectivity no matter how these sub-networks are extracted.

4.2 Erdös-Rényi random graphs

Three scale networks were simulated for this mode. The networks in the same scale have different probabilities. Compared one network with another in the same table, we can find how the probability affects properties. Also, Compared one network with another from the different table and with the same probability, we can find whether properties make big changes with network growth.

Case 1: The total number of nodes is 1000.

Probability	Edges	Nodes in the giant component	Diameter *(exact/ sampled)*	Clustering coefficient	**CPU time for diameter** *(exact / sampled)*	Degree sequences
0.5	249750	1000	1.5/2.0	0.50012	13845 *s* / 20 *ms*	r-a
0.3	149850	1000	1.7	0.30033		r-b
0.15	74925	1000	1.8	0.15003		
0.1	49950	1000	1.9	0.10013		r-c
0.05	24975	1000	2.0/2.1	0.05016	13*s* /20 *ms*	
0.01	4995	1000	3.3	0.01052		
0.005	2497	993	4.4/4.3	0.00378	3 *s* /20 *ms*	

Case 2: The total number of nodes is 500.

Probability	Edges	Nodes in the giant component	Diameter	Clustering coefficient
0.5	62375	500	1.5	0.50056
0.3	37425	500	1.7	0.29982
0.15	18712	500	1.8	0.14944
0.1	12475	500	1.9	0.10127
0.05	6237	500	2.2	0.04926
0.01	1247	497	4.0	0.00903
0.005	623	449	6.4	0.00414

Case 3: The total number of nodes is 50.

Probability	Edges	Nodes in the giant component	Diameter	Clustering coefficient
0.5	612	50	1.5	0.51816
0.3	367	50	1.7	0.28793
0.15	183	50	2.1	0.14125
0.1	122	50	2.6	0.10879
0.05	61	45	3.8	0.01133
0.01	12	4	*1.7	0.0
0.005	6	3	*1.3	0.0

*Note: * The values are obtained from the maximum connected component. Since the size of the connected component is very small, the values cannot reflect the diameter of a graph.*

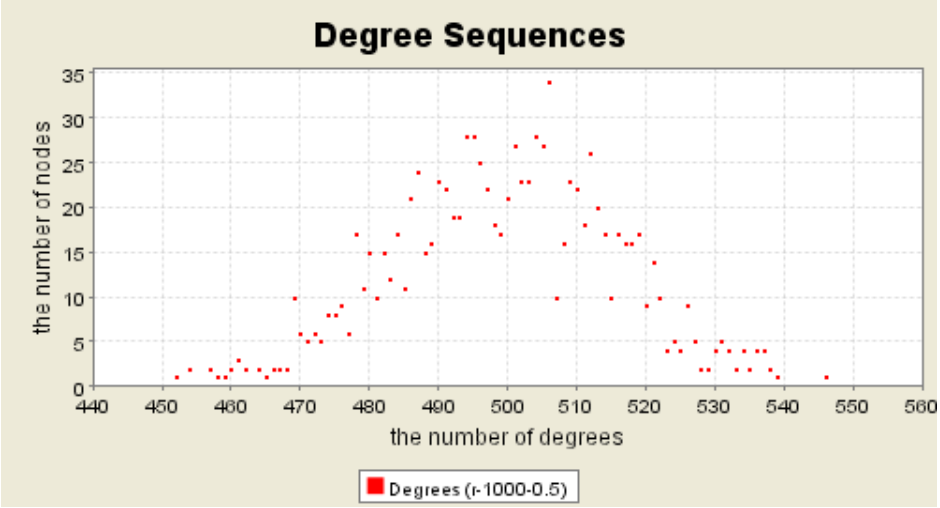


Fig. r-a

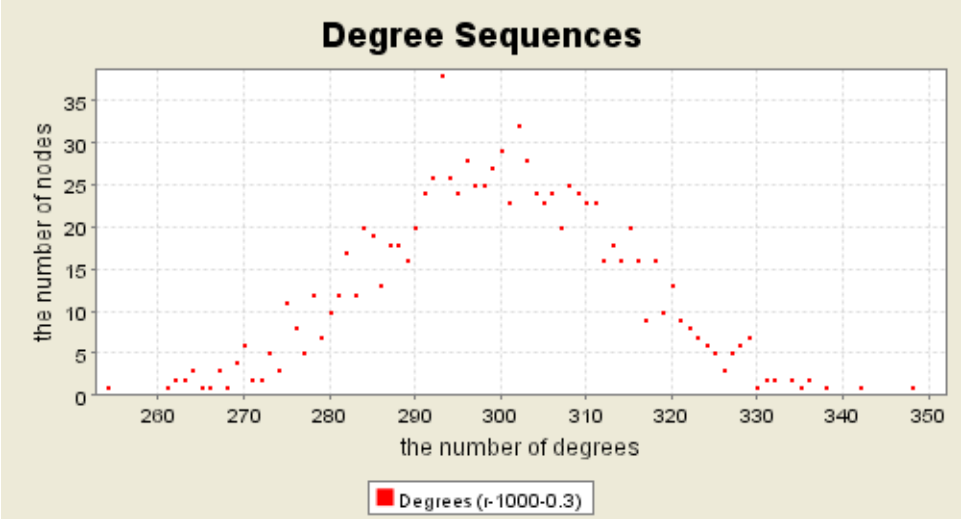


Fig. r-b

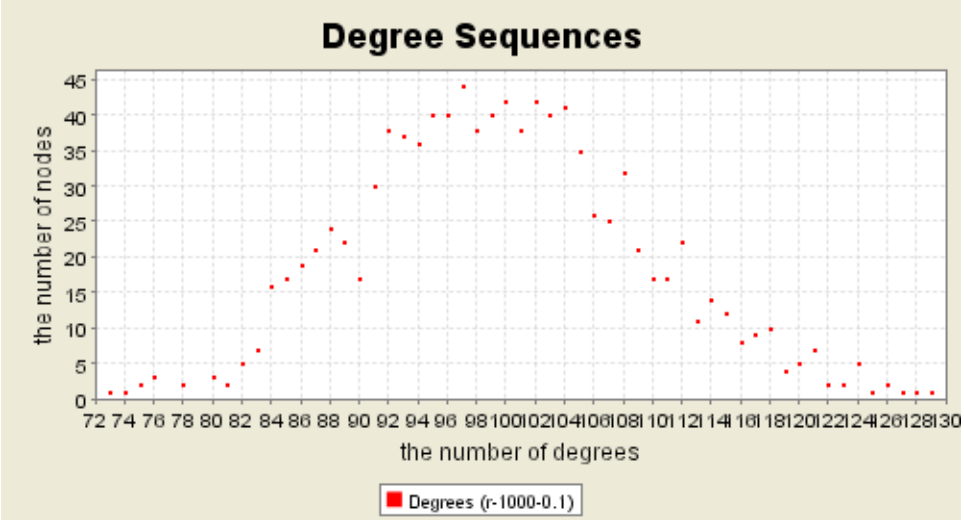


Fig. r-c

The above figures of degree sequences indicate that Erdös-Rényi random model has a different degree distribution from movie-actor networks. This model has a Poisson distribution, but movie-actor networks have a power law distribution.

4.3 Watts-Strogatz Small-world graphs

Case 1: Each graph has 1000 total node; the size of the giant component is also 1000. This table can allow us to investigate: 1) with the same neighbor distance, how the sampling networks vary with increasing probability. 2) with the same probability, how the sampling networks vary by changing the value of neighbor distance.

Neighbor distance	Probability	Edges	Diameter *(exact/ sampled)*	Clustering coefficient	**CPU time for diameter** *(exact / sampled)*	Degree sequences
2	0.0	2000	125.4	0.5		
2	0.005	2000	51.2/54.7	0.49336	2 *s* / 20 *ms*	
2	0.01	2000	29.5	0.48753		
2	0.05	2000	12.5/11.1	0.44083	2 *s* / 20 *ms*	
2	0.1	2000	9.2	0.39453		s-a
2	0.15	2000	7.6	0.32767		
2	0.3	2000	6.2	0.18408		
2	0.5	2000	5.6 / 5.6	0.07029	693*s* /40*ms*	s-b
2	1.0	2000	5.3	0.00634		s-c
3	0.0	3000	83.7	0.6		
3	0.005	3000	23.6	0.59019		
3	0.01	3000	16.7	0.58672		
3	0.05	3000	7.8	0.51199		
3	0.1	3000	6.2	0.44437		
3	0.15	3000	5.5	0.37192		
3	0.3	3000	4.7	0.22643		
3	0.5	3000	4.3	0.08019		s-d
3	1.0	3000	4.1	0.00461		
4	0.0	4000	62.9	0.64285		
4	0.005	4000	12.4	0.62866		
4	0.01	4000	10.6	0.62148		
4	0.05	4000	6.1	0.55211		
4	0.1	4000	4.9	0.46924		
4	0.15	4000	4.6	0.39908		
4	0.3	4000	4.0	0.22623		
4	0.5	4000	3.7	0.08972		s-e
4	1.0	4000	3.5	0.00761		

Case 2: Each graph has 500 total nodes and 1000 edges; the size of the giant component is also 500. Compare this with the above to learn how properties of the networks are related to the size of networks.

Neighbor distance	Probability	Diameter	Clustering coefficient
2	0.0	62.9	0.5
2	0.005	26.6	0.487
2	0.01	24.7	0.4862
2	0.05	10.4	0.4314
2	0.1	7.9	0.37466
2	0.15	6.7	0.33387
2	0.3	5.5	0.18868
2	0.5	4.9	0.08323
2	1.0	4.8	0.02257

Case 3: Each graph has 50 total nodes and 100 edges; the size of the giant component is also 50.

Neighbor distance	Probability	Diameter	Clustering coefficient
2	0.0	6.6	0.5
2	0.005	5.9	0.49266
2	0.01	5.9	0.47333
2	0.05	4.0	0.41333
2	0.1	3.9	0.36
2	0.15	3.5	0.344
2	0.3	3.2	0.262
2	0.5	3.1	0.220 57
2	1.0	2.9	0.03066

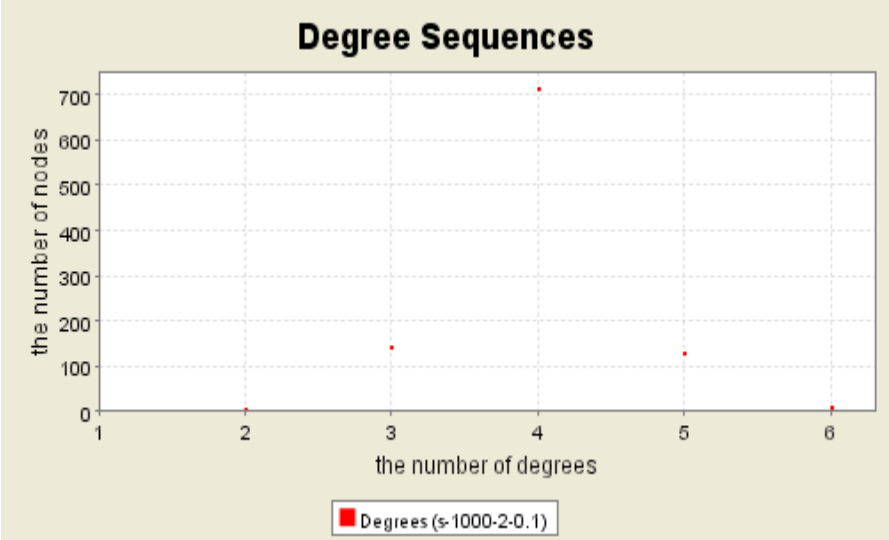


Fig. s-a

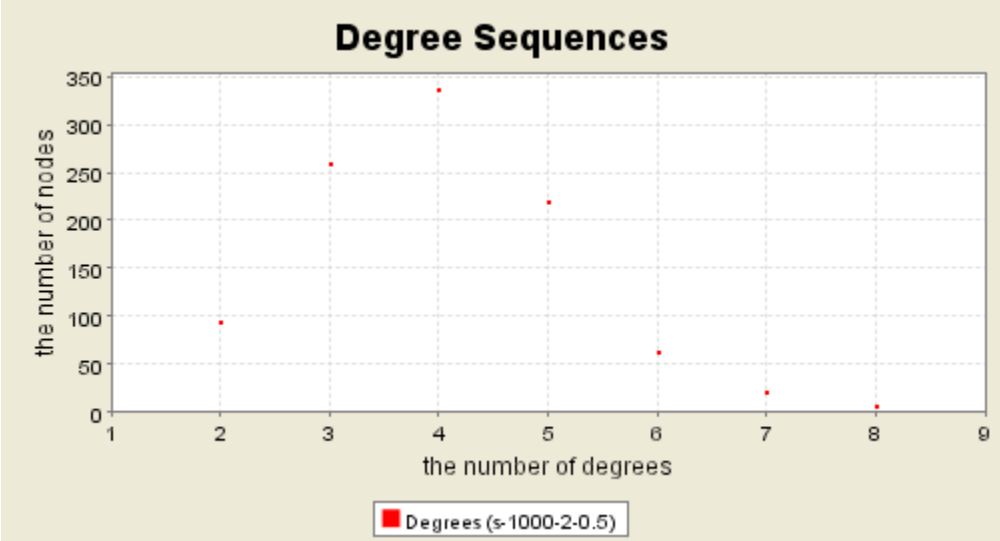


Fig. s-b

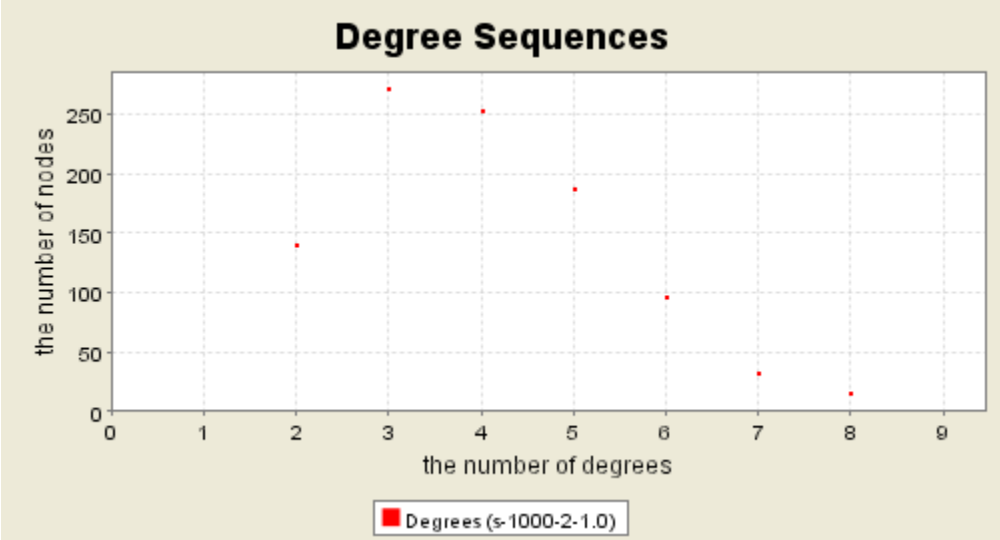


Fig. s-c

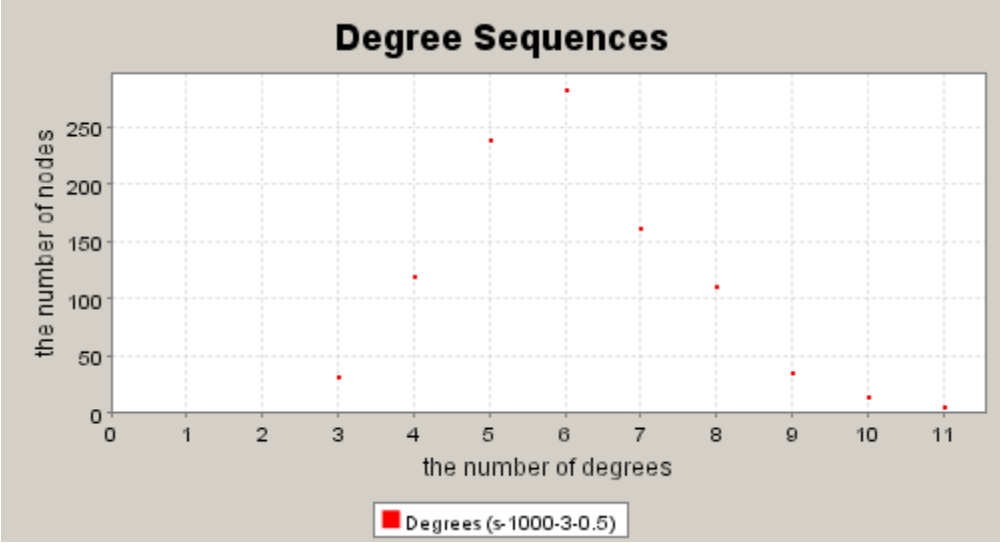


Fig. s-d

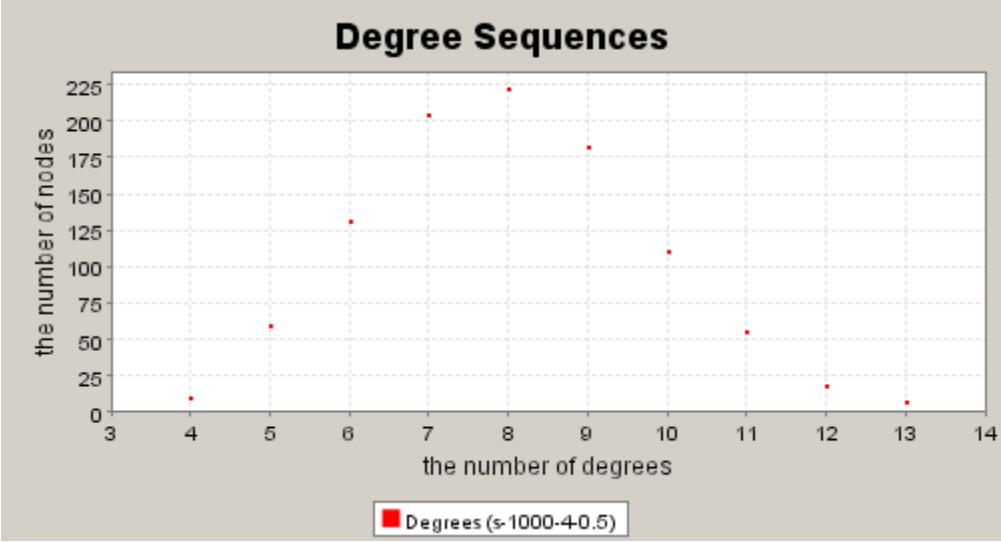


Fig. s-e

The above figures of degree sequences indicate that Watts-Strogatz model has a Poisson distribution in network connectivity, instead of a power law distribution that occurs in Movie-actor networks.

4.4 A.-L. Barabási Scale-free graphs

Case 1: Each graph has 1000 nodes.

Initial nodes (n_0)	Added edges (m)	Repeated times (t)	Edges	Nodes in the giant component	Diameter *(exact/ sampled)*	Clustering coefficient	**CPU time for diameter** *(exact / sampled)*	Degree sequences
15	10	985	9850	1000	2.5 / 2.4	0.06124	847*s* / 20 *ms*	sf-a
20	8	980	7840	999	2.7	0.04827		sf-b
8	5	992	4960	1000	3.0 / 3.0	0.0352	1893*s* / 40 *ms*	sf-c
5	3	995	2985	1000	3.5 / 3.6	0.02794	4*s*/20*ms*	

Case 2: Each graph has 500 nodes; the size of the giant component is also 500.

Initial nodes	Added edges	Repeated times	Edges	Diameter	Clustering coefficient
15	10	485	4850	2.4	0.09261
20	8	480	3840	2.5	0.07514
8	5	492	2460	2.7	0.06372
5	3	495	1485	3.3	0.04224

Case 3: Each graph has 50 nodes; the size of the giant component is also 50.

Initial nodes	Added edges	Repeated times	Edges	Diameter	Clustering coefficient
15	10	35	350	1.7	0.43112
20	8	30	240	1.99	0.34541
8	5	42	210	2.03	0.25882
5	3	45	135	2.3	0.24992

From the above tables, we can see that the parameters *m* and *t* determines the number of edges for a network with the specified number of nodes. And the number of edges seems to affect the diameter and clustering coefficient. For the same size networks, the more edges that networks have, the smaller diameter and the a little higher clustering coefficient that networks possess.

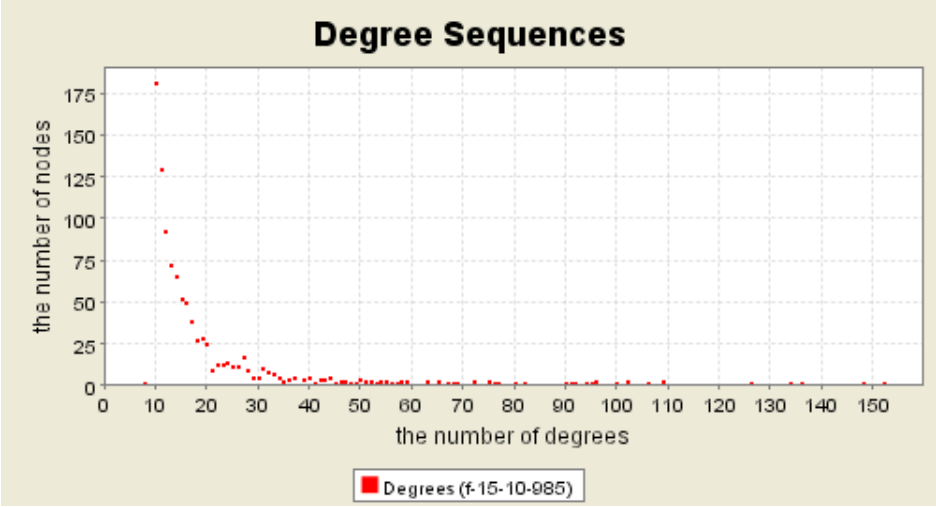


Fig. sf-a

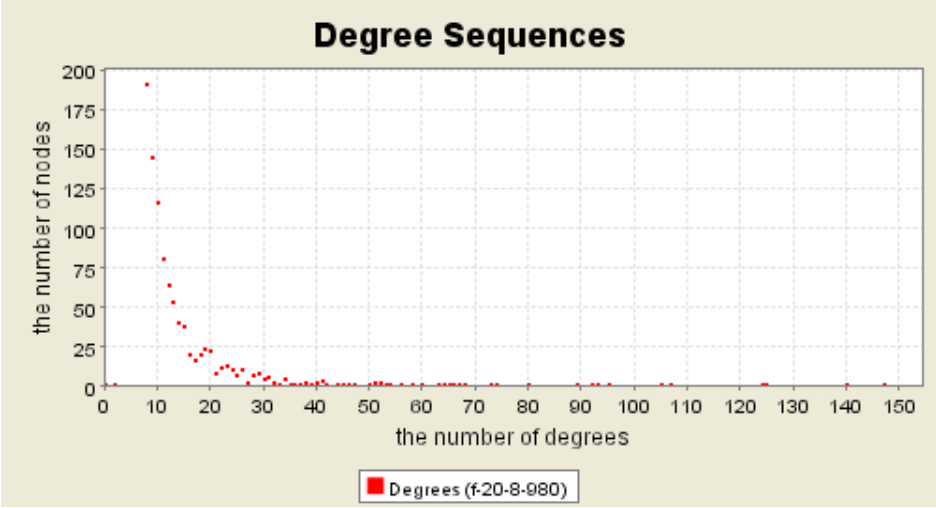


Fig. sf-b

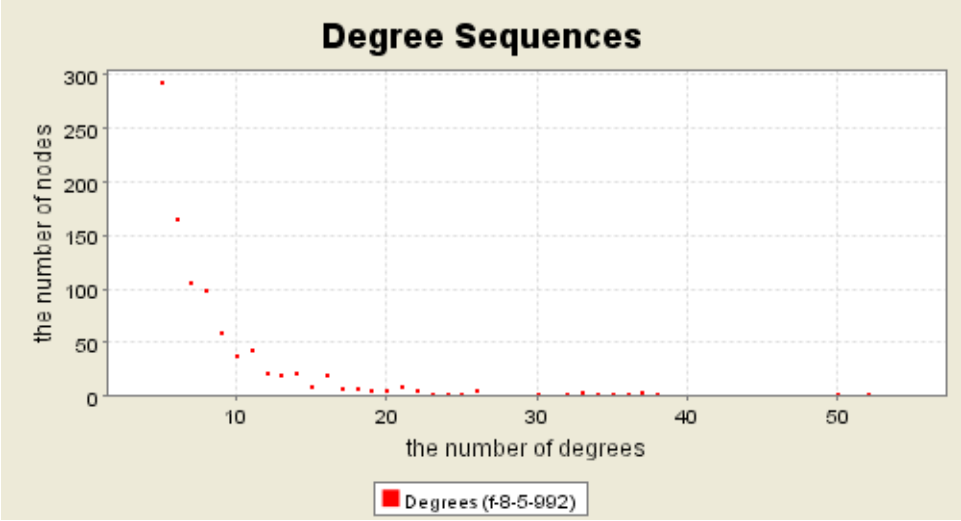


Fig. sf-c

The above figures of degree sequences indicate that A.-L. Barabási model has a power law distribution in network connectivity; witch is the same as movie-actor networks.

4.5 Jon Kleinberg graphs

a. The common properties in other models

In two dimensions:

Case 1: (k, n, p, q) = (2, 32, 1, 1); 2 dimension, lattice distance of 1 and long contacts of 1
Each graph in the following table is a directed graph with 1024 nodes and the size of the giant component in each graph is also 1024.

Clustering exponent	Edges	Diameter	Clustering coefficient	Degree sequences
0.5	4646	6.1	0.00352	d[2]=1, d[3]=42 d[4]=387, d[5]=594
1.0	4592	6.24	0.00646	d[2]=1, d[3]=52 d[4]=421, d[5]=550
1.5	4572	6.44	0.01794	d[2]=1, d[3]=51 d[4]=443, d[5]=529
1.8	4548	6.6	0.02710	d[2]=1, d[3]=73 d[4]=423, d[5]=527
1.9	4506	7.5	0.02846	d[2]=1, d[3]=60 d[4]=491, d[5]=472
2.0	4465	7.7	0.03065	d[2]=3, d[3]=54 d[4]=538, d[5]=429
2.1	4463	8.1	0.03536	d[2]=2, d[3]=62 d[4]=527, d[5]=433
2.2	4424	7.7	0.02918	d[2]=2, d[3]=69 d[4]=552, d[5]=401
2.5	4379	8.8	0.03641	d[2]=2, d[3]=85 d[4]=559, d[5]=376
3.0	4294	11.3	0.04269	d[2]=2, d[3]=83 d[4]=651, d[5]=287

Case 2: (k, n, p, q) = (2, 22, 1, 1); 2 dimension, lattice distance of 1 and long contacts of 1
Each graph in the following table is a directed graph with 484 nodes and the size of the giant component in each graph is also 484.

Clustering exponent	Edges	Diameter	Clustering coefficient
0.5	2143	5.7	0.00851
1.0	2151	5.6	0.01581
1.5	2126	6.18	0.02686
1.8	2101	6.2	0.03422
1.9	2076	7.2	0.03150
2.0	2071	6.5	0.03798
2.1	2062	6.7	0.03643
2.2	2044	7.4	0.03629
2.5	2027	8.5	0.04029
3.0	1994	9.4	0.03904

Case 3: (k, n, p, q) = (2, 30, 1, 1); 2 dimension, lattice distance of 1 and long contacts of 1
Each graph in the following table is a directed graph with 900 nodes and the size of the giant component in each graph is also 900. Each row in the table contains data of directed and undirected graphs. The first line in a row represents the directed graph data and the second line is for the undirected graph.

Clustering exponent	Edges	Diameter	Clustering coefficient
1.0	4380	12.7	0.01179
	2189	10.2	0.05774
2.0	4380	11.7	0.02607
	2183	8.2	0.09166
3.0	4380	9.9	0.10499
	2131	9.5	0.11025

Case 4: (k, n, p, q) = (2, 30, 1, 2); 2 dimension, lattice distance of 1 and long contacts of 2
The number of nodes is 900 and the size of the giant component is also 900. Each row in the table contains data of directed and undirected graphs. The first line in a row represents the directed graph data and the second line is for the undirected graph.

Clustering exponent	Edges	Diameter	Clustering coefficient
1.0	5280	10.4	0.02557
	2640	9.1	0.09751
2.0	5280	8.5	0.05024
	2640	6.1	0.13203
3.0	5280	6.1	0.16314
	2640	6.5	0.16774

In three dimensions:

Case 1: (k, n, p, q) = (3, 10, 1, 1); 3 dimension, lattice distance of 1 and long contacts of 1
Each graph in the following table is a directed graph with 1000 nodes and the size of the giant component in each graph is also 1000.

Clustering exponent	Edges	Diameter	Clustering coefficient	Degree sequences
0.5	6047	5.1	0.00302	d[3]=3, d[4]=39, d[5]=202 d[6]=420, d[7]=336
1.0	6025	5.5	0.00491	d[3]=3, d[4]=51, d[5]=185 d[6]=440, d[7]=321
1.5	5966	5.1	0.00942	d[3]=3, d[4]=44, d[5]=236 d[6]=418, d[7]=299
2.0	5950	5.6	0.01367	d[3]=3, d[4]=47, d[5]=235 d[6]=427, d[7]=288
2.5	5867	5.8	0.01898	d[3]=5, d[4]=57, d[5]=249 d[6]=444, d[7]=245
2.8	5814	6.6	0.02704	d[3]=5, d[4]=58, d[5]=256 d[6]=480, d[7]=201
3.0	5783	6.3	0.02712	d[3]=7, d[4]=55, d[5]=271 d[6]=482, d[7]=185
3.1	5760	6.4	0.02506	d[3]=6, d[4]=61, d[5]=291 d[6]=451, d[7]=191
3.3	5729	6.3	0.02403	d[3]=7, d[4]=70, d[5]=281 d[6]=471, d[7]=171
4.5	5565	8.7	0.01744	d[3]=7, d[4]=83, d[5]=351 d[6]=456, d[7]=103

Case 2: (k, n, p, q) = (3, 8, 1, 1); 3 dimension, lattice distance of 1 and long contacts of 1
Each graph in the following table is a directed graph with 512 nodes and the size of the giant component in each graph is also 512.

Clustering exponent	Edges	Diameter	Clustering coefficient
0.5	3015	4.6	0.00299
1.0	3009	4.8	0.00981
1.5	2997	4.6	0.01144
2.0	2970	4.8	0.01916
2.5	2916	5.3	0.01841
2.8	2887	5.6	0.02329
3.0	2870	5.6	0.02369
3.1	2855	5.9	0.02355
3.3	2848	5.8	0.02643
4.5	2757	7.9	0.01487

Case 3: (k, n, p, q) = (3, 10, 1, 2); 3 dimension, lattice distance of 1 and long contacts of 2
The number of nodes is 1000 and the size of the giant component is also 1000. Each row in the table contains data of directed and undirected graphs. The first line in a row represents the directed graph data and the second line is for the undirected graph.

Clustering exponent	Edges	Diameter	Clustering coefficient
1.5	7400	6.3	0.01939
	3698	4.9	0.09260
3.0	7400	4.8	0.0752
	3659	5.4	0.09910
4.5	7400	5.9	0.14985
	3332	5.9	0.10565

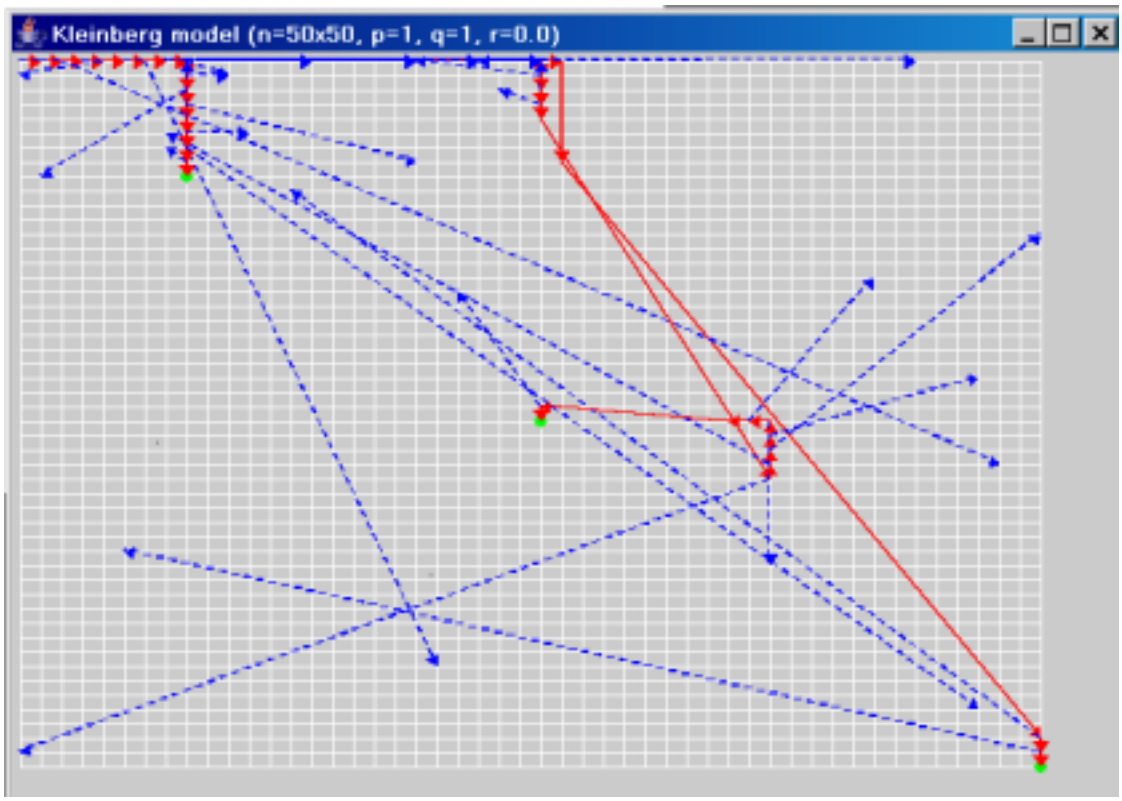
Case 4: (k, n, p, q) = (3, 10, 3, 2); 3 dimension, lattice distance of 3 and long contacts of 2
The number of nodes is 1000 and the size of the giant component is also 1000. Each row in the table contains data of directed and undirected graphs. The first line in a row represents the directed graph data and the second line is for the undirected graph.

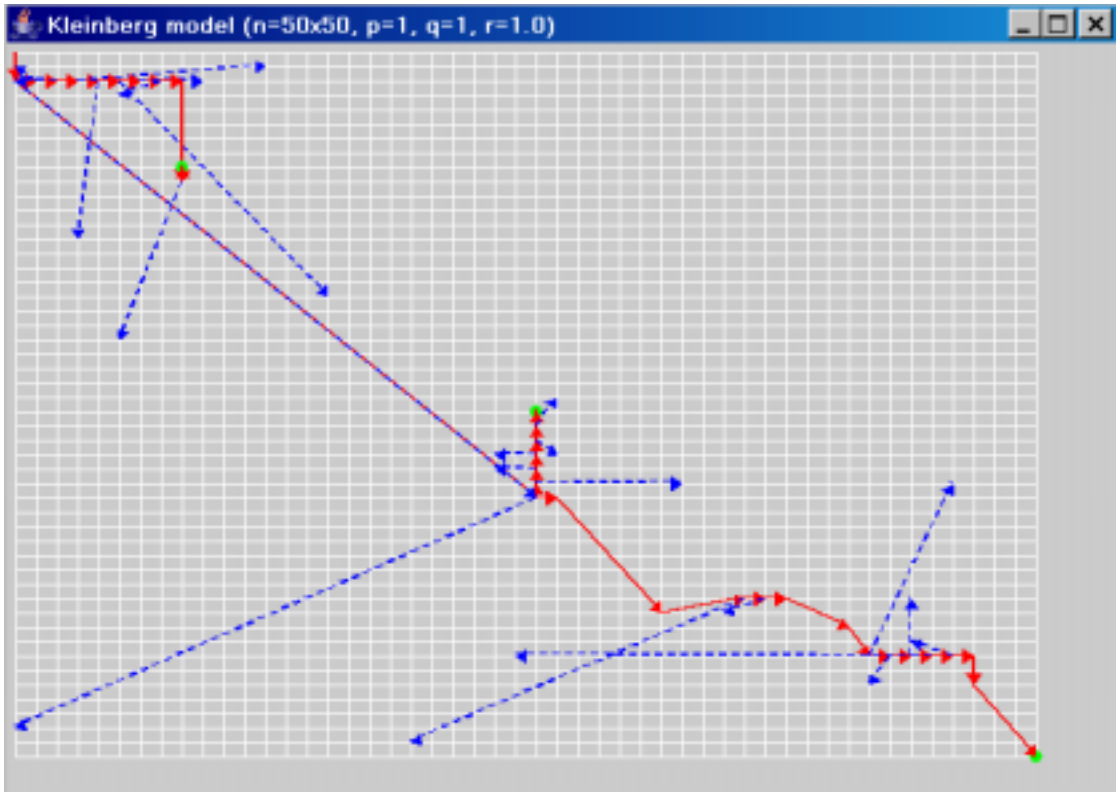
Clustering exponent	Edges	Diameter	Clustering coefficient
1.5	49232	2.7	0.42533
	24606	3.1	0.43220
3.0	49232	3.1	0.43510
	24448	2.8	0.44097
4.5	49232	2.7	0.44280
	23807	3.5	0.45587

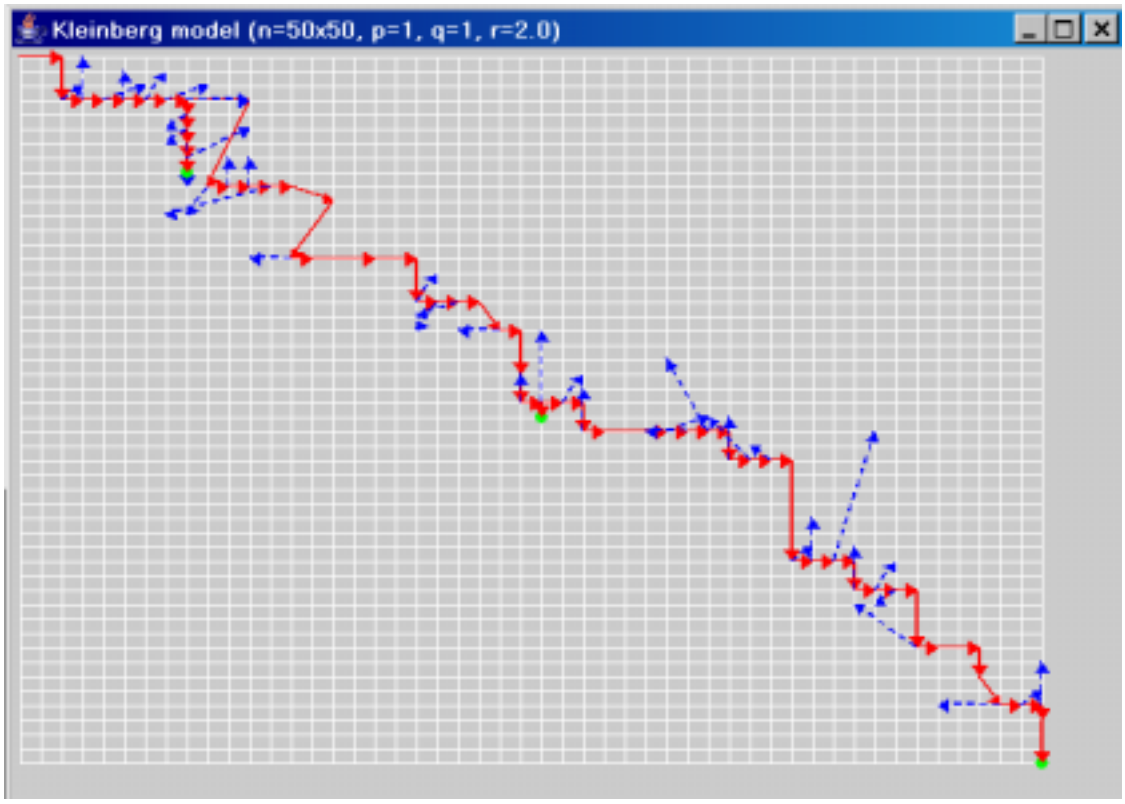
b. Delivery Time

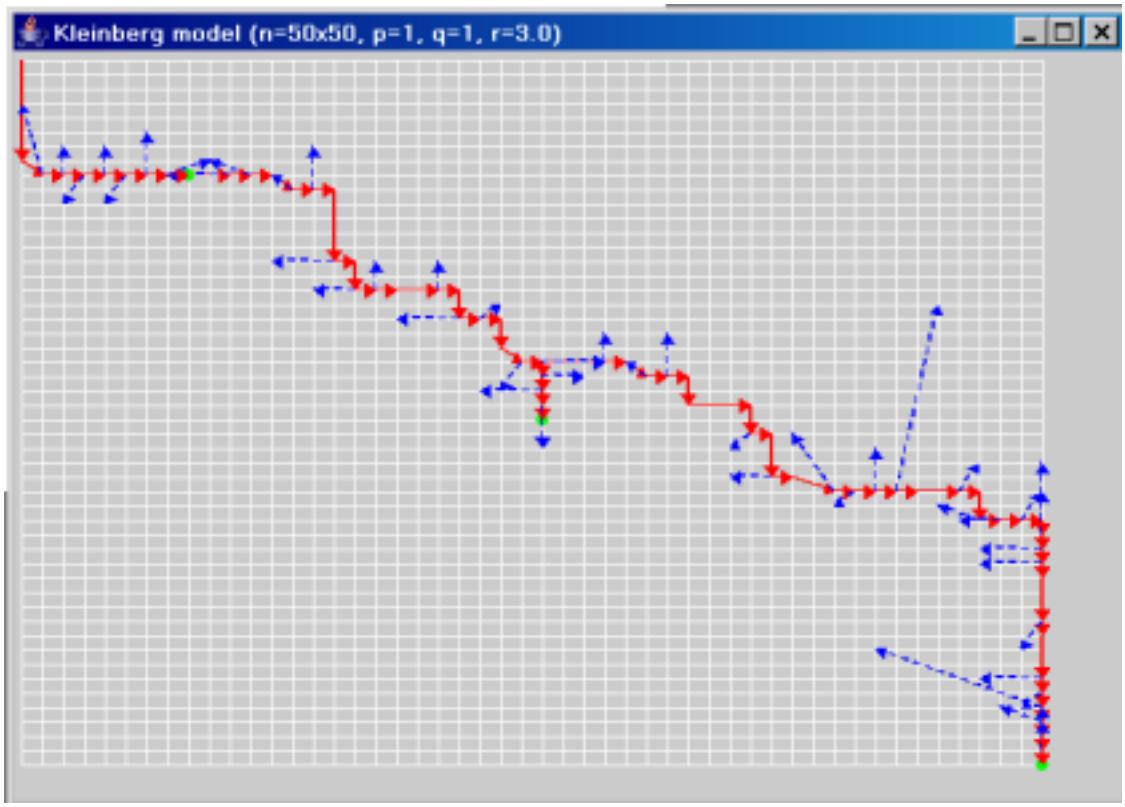
The purpose of following experiments is to investigate how the clustering exponents affect the short path finding (which we call the delivery time).

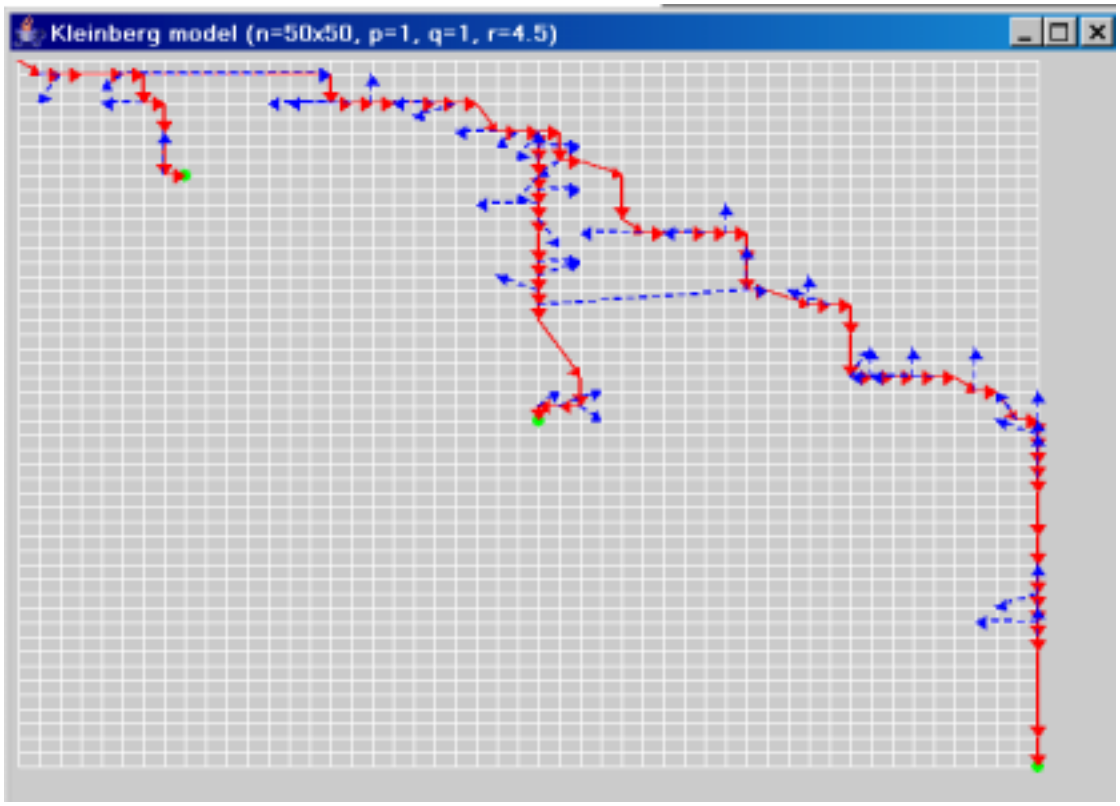
(k, n, p, q) = (2, 50, 1, 1 < or 2>) 2 dimension, lattice distance of 1 and long contacts of 1 or 2. Each graph is a directed graph with 50 x 50 nodes. The following figures show how the decentralized algorithm finds the shorter paths with the various clustering exponents. The source node is the most left upper point. Three target nodes are marked green. The red solid lines are the paths from the source node to each target node. The blue dash lines indicate the long contacts.

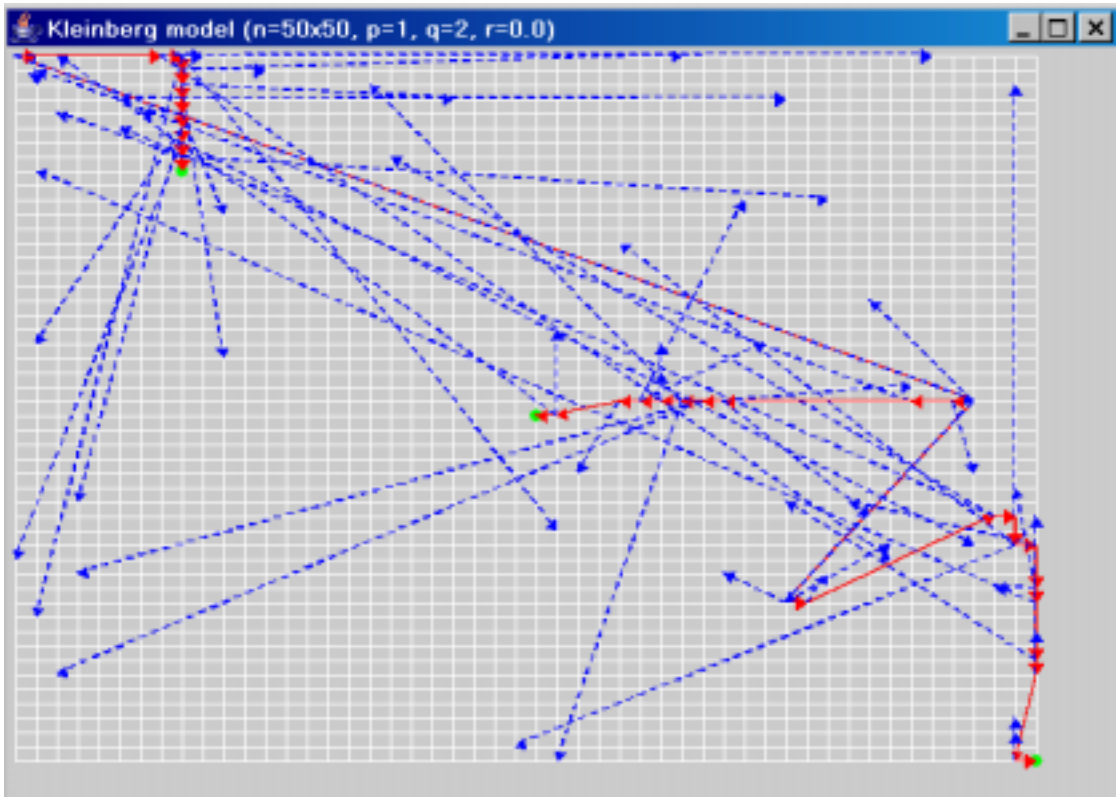

Kleinberg model (n=50x50, p=1, q=1, r=0.0)

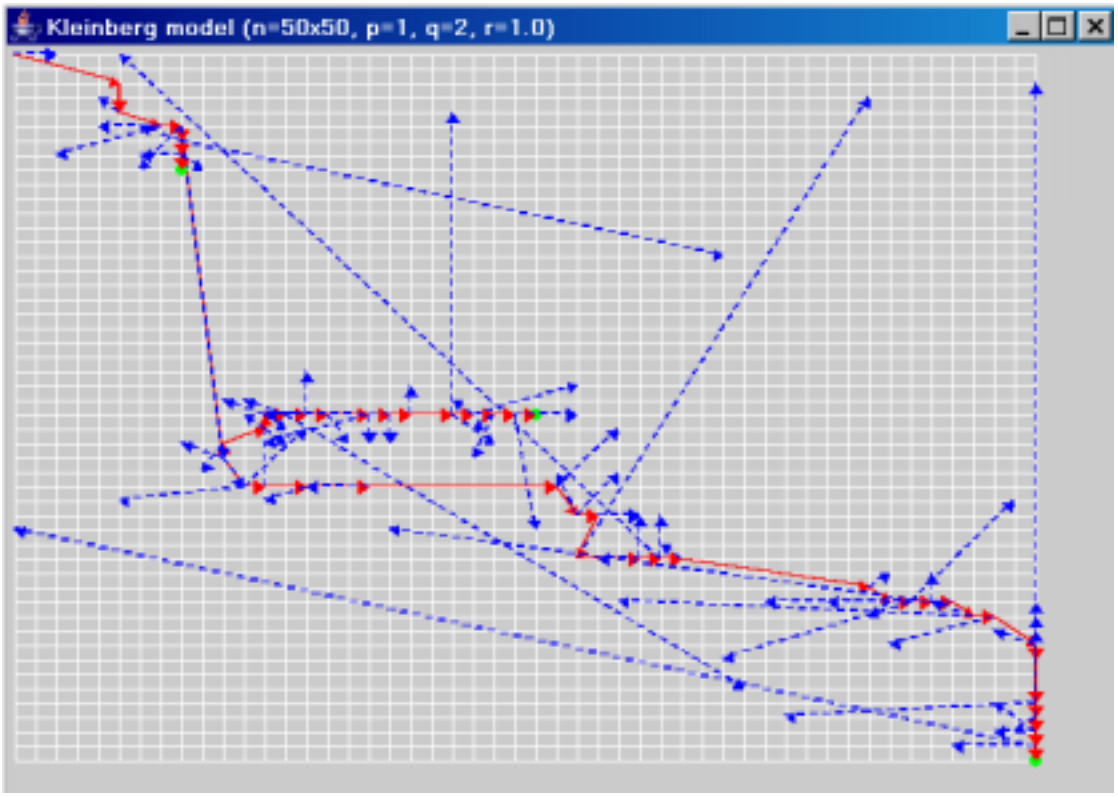

Kleinberg model (n=50x50, p=1, q=1, r=1.0)

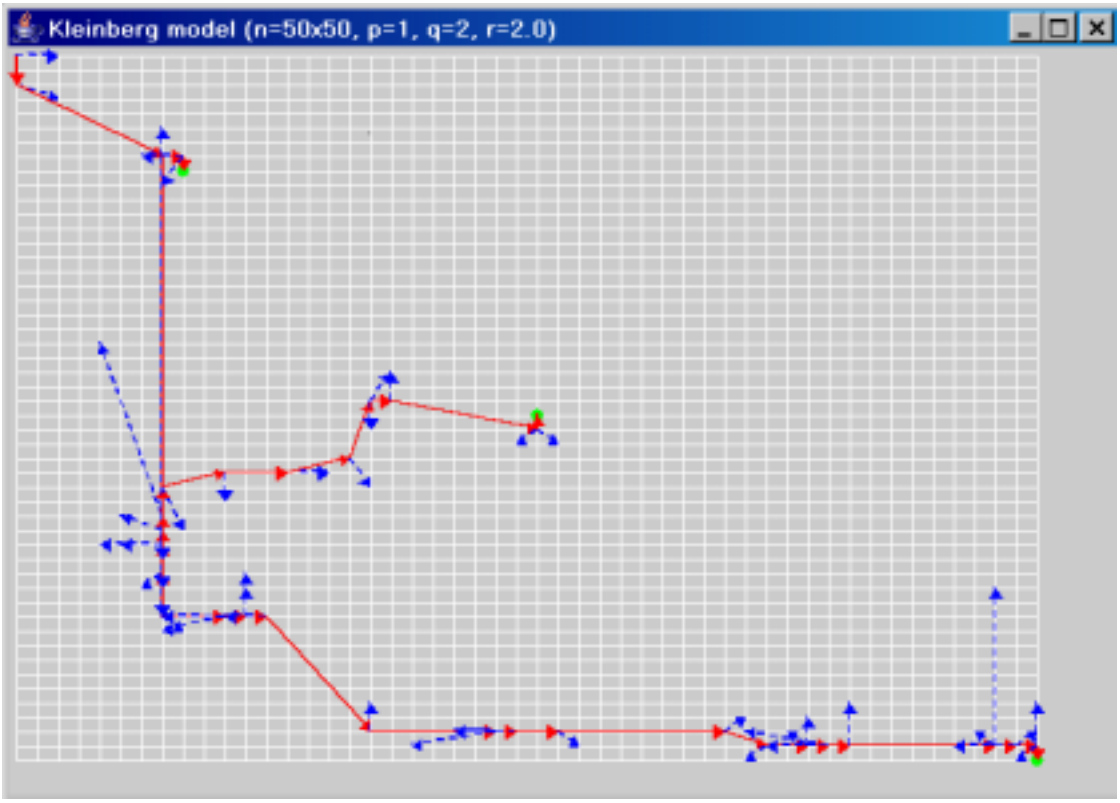

Kleinberg model (n=50x50, p=1, q=1, r=2.0)

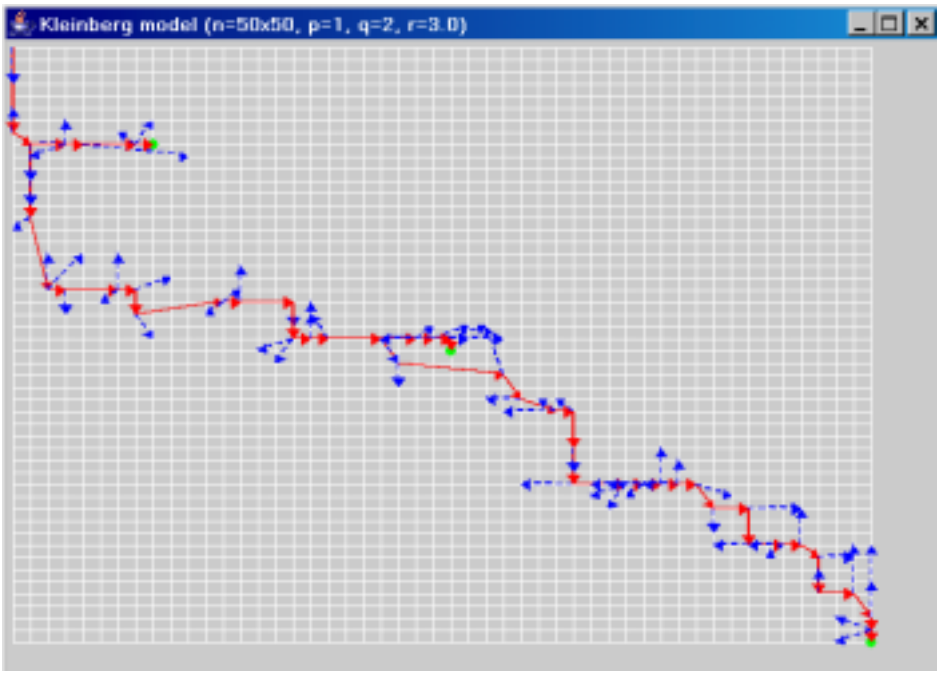

Kleinberg model (n=50x50, p=1, q=1, r=3.0)


Kleinberg model (n=50x50, p=1, q=1, r=4.5)


Kleinberg model (n=50x50, p=1, q=2, r=0.0)


Kleinberg model (n=50x50, p=1, q=2, r=1.0)


Kleinberg model (n=50x50, p=1, q=2, r=2.0)



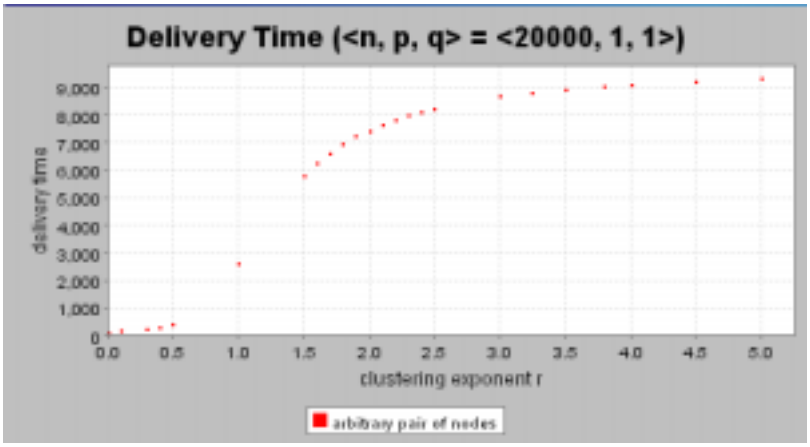
According to the above figures, we can see that deliver time gets longer with *r* increase no matter where the target node is. This is because the long-range contacts of a node are inclined to fall in the nearby partitions. Thus, it's hard to find one neighbor node that is far away from the current node, but closer to the target node when *r* is so big. The following table shows how partition distribution is.

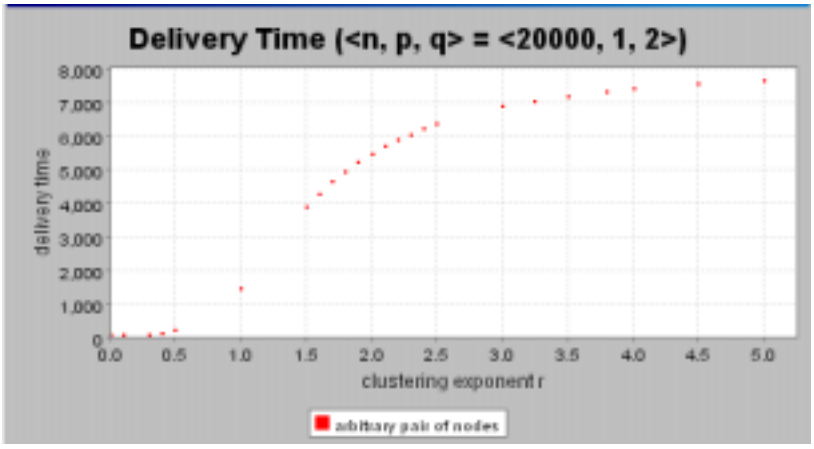
Partition distribution during the formation of each graph. (The parameter n determines the maximum number of partitions. For n = 50, the maximum partition number is 6)

Exponent	A1	A2	A3	A4	A5	A6
0.0	7	5	6	5	7	3
0.1	11	10	8	11	9	7
0.3	19	16	13	4	8	2
0.4	8	10	7	4	1	4
0.5	5	9	4	5	4	5
1.0	14	5	5	4	2	0
1.5	34	19	5	1	0	0
1.8	51	13	1	0	1	0
2.0	50	9	2	0	0	0
2.3	18	7	0	1	0	1
2.5	22	5	0	1	1	0
3.0	59	5	2	0	0	0
3.25	61	10	0	0	0	0
3.8	85	4	0	0	0	0
4.0	59	5	0	0	0	0
4.5	73	2	2	0	0	0
5.0	90	2	0	0	0	0
20.0	76	0	0	0	0	0

(k, n, p, q) = (2, 20,000, 1, 1 < or 2>) 2 dimension, lattice distance of 1 and long contacts of 1 or 2. Each graph in the following table is a directed graph with 20,000 x 20,000 nodes. Each data of delivery time is the average of 1000 runs.

Clustering exponent	Delivery time *q=1* / *q=2*
0.0	152 / 86
0.1	163 / 93
0.3	238 / 128
0.4	314 / 174
0.5	447 / 244
1.0	2622 / 1493
1.5	5822 / 3907
1.6	6243 / 4313
1.7	6619 / 4661
1.8	6937 / 4972
1.9	7221 / 5259
2.0	7451 / 5505
2.1	7658 / 5729
2.2	7832 / 5916
2.3	7994 / 6085
2.4	8133 / 6239
2.5	8257 / 6385
3.0	8695 / 6896
3.25	8839 / 7071
3.5	8955 / 7219
3.8	9067 / 7353
4.0	9125 / 7430
4.5	9241 / 7566
5.0	9311 / 7664
20.0	9483 / 7881
200.0	9483 / 7881





4.6 Movie-Actor networks vs. Models

Each network in the following table has 50000 nodes.

Network	Edges	Nodes in the giant	Diameter	Clustering coefficient
Movie-Actor	213144	35255	4.9	0.48730
Small-world	200000	50000	11.0	0.55463
Random	211866	49989	5.4	1.7E-4

5. Analysis of Results

According to the tables and figures in section 4, we can find some properties presented by the models and movie-actor networks and which real networks are best simulated by each model.

5.1 Movie-Actor Network

The tables and figures in section 4.1 show that movie-actor network has these properties of the small-world network set that are discussed in section 2.1:sparseness, small diameter, clustering, power-law degree distribution and the emergence of a giant component. These properties hold in each extracted movie-actor sub-graph even though these networks have differences in the network size, network formation time and location. Especially, the log-log plot of degree sequence in Fig. am-a(2) suggests that the data are consistent with a power law decay with the exponent $r \approx 1.9$. The apparent exponent in my experiments is a little different from $r \approx 2.3 \pm 0.1$ reported by Amaral in [2].

5.2 Models

1) Erdös-Rényi random model

   For sampling graphs with the same size, their properties are totally determined by the connection probability. With the increment of the probability *p*, the number of edges and clustering coefficient are getting bigger and bigger, but the diameter is getting smaller. The number of edges varies with the expectation value $|E| = p\binom{|V|}{2}$. When $p=1.0$, the graph is a complete graph. Consequently the diameter reaches the smallest value 1 and the clustering coefficient reaches the largest value 1.0. The networks generated by this model usually have a very small diameter even though they are quite sparse for small *p*. The graphs have a Poisson degree distribution. The mean degree $<k>$ in the degree distribution is dependent on the number of vertices *n* and the probability *p*. For graphs with the same size, $<k>$ increases with the probability *p* increasing. The tables in section

4.2 show that both the diameter and clustering coefficient are quite independent of the number of vertices. I think that this is because every other node *v* gets a connection from a given node *u* only according to a uniform probability *p*. From the third table in section 4.2, we can see that the giant component doesn't emerge when the number of edges is less than half the number of nodes, which is consistent with [3] and what Brian Hayes said in [5].

2) Watts-Strogatz Small-world model

Since the rewiring probability introduces the random connections to a regular graph, big changes happen to the properties of graphs when compared with the original graph. The diameter sharply decreases at the beginning of this probability increment, but later the decline gets smooth. However, the clustering coefficient changes in another way with the probability increasing, it's getting smaller a little bit at the beginning, and then decreases much more. The neighbor distance *(k)* and the number of nodes determine the number of edges, $|E| = k|V| \geq |V|$. Thus, the giant component can always emerge in sampling networks generated by this model. In this model, I found that the rewiring probability *p* is a key parameter to generate a sampling network that can simulate a real network well. We know this fact that the clustering coefficient and diameter of a network are mainly determined by *p* and the speed of their changes is quite different with *p* increasing. Therefore, what is the range of *p* to balance them better? That is, how can we obtain not only higher clustering coefficient, but also lower diameter in a sampling network? From the tables shown in section 4.3, we can find that the range of *p* from 0.05 to 0.15 is good for both. Of course, the neighbor distance plays a little role in both properties as well. Based on the same other parameters, the bigger the neighbor distance is, the smaller the diameter is and the bigger the clustering coefficient is. Like the random model, this model also has the property of Poisson degree distribution. But, from figures in section 4.3, we can see that the mean degree $<k>$ is only dependent on the neighbor distance *k*, which is $<k> = 2*k$, independent of the number of vertices *n* and the rewiring probability *p*. In section 4.3, the sampling networks are generated in three different sizes. I found

that the clustering coefficient is almost independent of the number of vertices. This is because the base graph is a regular graph and each node gets the rewiring connections in a uniform way.

3) A.-L. Barabási Scale-free model

   This model can generate the networks with the power-law distribution no matter what parameters are passed. This characteristic is quite different from the behavior of the random model and small-world model. But it is similar to the degree distribution of the movie-actor networks. Three tables in section 4.4 show us the growth of the networks. The diameter increases slowly and the clustering coefficient declines a little faster while the network evolves. The clustering coefficient is very small in this model, though it is a little bigger than the one in Erdös-Rényi random model based on the similar scale of networks.

4) Jon Kleinberg model

   This model was investigated by generating directed/undirected graphs, but only directed graphs in the original model [13], in two/three dimensions. Since each node has its own position, we can use this information to find the short path between any pair of nodes. A decentralized algorithm [13] was applied to do this: In each step, the current node *u* chooses the next node that is as close to the target node *t* as possible according to the lattice distance. But it's not guaranteed to find the shortest path between any pair of nodes by applying this algorithm. I couldn't prove the theorem about lower bound as Jon Kleinberg said in [13]: when the clustering exponent *r* is equal to the value of the dimension, the graph should have a very small delivery time bounded by a function proportional to $(logN)^2$; For every other exponent, an asymptotically much larger delivery time is required. According to my experiments, the path becomes longer without changing other parameters except clustering exponent *r* increment. This is because *r* is anti-proportion to the width of long contacts. In other words, nodes becomes clustering and the number of short-cut edges becomes smaller and smaller with *r* increasing. We can see this from the partition distribution table.

When $r = 0.0$, each partition gets the number of long contacts very close. But with r increasing, the partitions closer to the node have higher possibility to obtain long contacts. This model doesn't have the Power-law or Poisson degree distribution.

5.3 Movie Actor Network vs. Models

Thus far, the movie-actor networks and each model are separately studied. How does each model simulate the real movie-actor networks? Only the A.-L. Barabási Scale-free model has the power-law degree distribution as movie-actor networks do. Other models don't have this kind of distribution. The table in section 4.6 shows us that the network generated by the small-world model has a clustering coefficient close to the one in the movie-actor network, but there is a much smaller clustering coefficient in the network generated by the random model than the one in the movie-actor network. Since the scale-free model has the similar clustering coefficient to the random model, we can say that the scale-free model also cannot simulate this property. Also, the table represents that the diameter in the random model is close to the one in the movie-actor network. Although the diameter in the small-world model is almost twice as large as the one in the movie-actor, it is possible that the small-world model can obtain a comparable diameter for this movie-actor network by changing the parameters: neighbor distance and rewiring probability. Therefore, there is not one model from the above four models, which can best simulate all small-world properties hidden in movie-actor networks.

5.4 Conclusions

The study of Movie-Actor networks tells us that: the real complex networks are not totally random as they look. In fact, these networks have generic organizing characteristics so that they can be categorized in some kind of the network set. These characteristics determine the topologies of networks and impact on the evolution of networks. It is these underlying properties that provide cues for us to know system functions of networks and to predict the dynamic changes in networks. This field study is wide and valuable. Lots of mysteries behind complex networks need to be uncovered.

Thus, it gives some open challenges. This project is just the beginning of exploring complex networks.

6. Problem Encountered

6.1 DFS Implementation

The DFS algorithm is used in this project for investigating the giant component of networks. I used the recursive method to implement DFS at first. It worked fine when the network size is not big. Later, I found it's impossible to run on large networks by this method. I got the out of stack error. So I changed the recursive method to the loop method for DFS implementation and it worked well.

6.2 Diameter of Large Graphs

In order to get the diameter of a graph, we need to calculate the shortest paths in all pairs of nodes by applying the BFS algorithm. This computation is very expensive in time. So, I adopted the sampling method to do this for large graphs with 10000+ nodes. In order to prove this method feasible, I compared the sampled diameter with the exact diameter by testing on some graphs generated by various models (see section 4). Thus it was found that the sampled diameter is close to the exact diameter and, on the other hand, the sampling method is much faster than the exact method.

6.3 Directed vs. Undirected Graphs

The graphs generated by the Jon Kleinberg model are directed graphs. This is different from other three models, which generate undirected graphs. In order to make Jon Kleinberg model comparable to other models, I gave an option for undirected graphs in this model generator. Also, the analyzer program supports character analysis for both kinds of graphs. Compared directed graphs with undirected graphs generated by

Kleinberg model from experiments in section 4, directed graphs mostly have a little higher clustering coefficient than undirected graphs.

6.4 Movie-Actor Network Analysis

The main problem in movie-actor network analysis is the network size. Due to the limitation on huge computation for properties, I only did analysis on a small part of full-size networks extracted by different time or different countries. The sampling data may not be the same as the real data. But there is not a big gap between them because they come from the original networks.

7. Future Enhancement

7.1 Optimization algorithm

Although I have changed some algorithms for analysis to improve the performance several times, I still feel that it can be improved further. It's expensive to analyze the giant networks with 100000+ nodes.

7.2 Experiments on much larger Movie-Actor graphs

Doing experiments on much larger movie-actor graphs can give us more accurate data so that we can correctly investigate the properties hidden in movie-actor networks.

7.3 Experiments on other real networks

Studying on other real networks and doing comparison among them can make us more understand the small-world network set. Only studying on one type of a network like Movie-Actor network is insufficient to prove that the properties obtained from such a network are the same ones as the small-world networks are provided with. We can study some real networks such as www, organization, cell, etc.

7.4 Graphic User Interface

The analyzer only supports GUI display for the degree sequence right now. It's friendly to make the analyzer as a window program. All operations such as loading file, writing file, changing options and etc. are triggered in the window mode.

8. References

[1] Reka Albert, and Albert-László Barabási, Statistical mechanics of complex networks, Reviews of modern physics 74, http://www.nd.edu/~networks/PDF/rmp.pdf.

[2] L.A.N. Amaral, A. Scala, M. Barthelemy, and H. E. Stanley, 2000, Classes of Small-world networks, http://polymer.bu.edu/~amaral/Papers/pnas00a.pdf.

[3] Bela Bollobás, 1985, Random Graphs, Academic press.

[4] James Case, 2001, The Continuing Appeal of Small-world Networks, SIAM News, Volume 34, Number 9.

[5] Brian Hayes, 2000, Graph theory in practice Part I/ Part II, American Scientist.

[6] Petter Holme, 2001, Characteristics of Small World NetWorks, http://www.tp.umu.se/~holme/seminars/swn.pdf.

[7] Christopher P. Mawata, 1997, Graph Theory Lessons, http://www.utc.edu/~cpmawata/petersen/index.htm.

[8] Steven H, Strogatz, 2001, Exploring complex networks, Nature 410:268-276.

[9] Ducan J, Watts, and Steven H, Strogatz, 1998, Collective dynamics of 'Small-world' networks, Nature 393:440-442.

[10] Albert-László Barabási, July 2001, The physics of the Web, Physics Web http://www.physicsweb.org/article/world/14/7/09.

[11] Movie Actor database: ftp://ftp.imdb.com/pub/interfaces/.

[12] Study of Self-Organized Networks at Notre Dame, http://www.nd.edu/~networks/.

[13] Jon Kleinberg, 1999, The Small-World Phenomenon: An Algorithmic Perspective, Cornell Computer Science Technical Report 99-1776.

[14] Jon Kleinberg, 2001, Small-World Phenomena and the Dynamics of Information, Advances in Neural Information Processing Systems (NIPS) 14, 2001.

[15] Christopher M. Homan and Gabriel Istrate, Small Worlds, Locality, and Flooding on Landscapes, University of Rochester, Department of Computer Science Technical Report TR-2003-796.

[16] Jon M Kleinberg, 24 August 2000, Navigation in a small world, Nature, Vol 406.