

Modeling and Simulation of a Complex Social System: A Case Study

Yongqin Gao
Computer Science and Engineering Dept.
University of Notre Dame
Notre Dame, IN 46556
ygao1@nd.edu

Greg Madey
Computer Science and Engineering Dept.
University of Notre Dame
Notre Dame, IN 46556
gmadey@nd.edu

Vincent Freeh
Department of Computer Science
North Carolina State University
Raleigh, NC 27685
vin@ncsu.edu

Abstract

The Open Source Software (OSS) development movement is a classic example of a social network. It is also a prototype of a complex evolving network. After over two years of collecting developer and project information from SourceForge, we have sufficient data to study the dynamic and structural mechanisms that govern the evolution and topology of this complex system. First, we collected empirical data from SourceForge and analyzed the statistics of the topological information of the OSS developer social network. By inspecting the network's evolution over time, we were able to gain knowledge about the appearance of the complex network and characteristics of the system. Then we generated a model to depict the evolution of this network based on the statistical results we got from the empirical data. Finally, using Java/Swarm, we simulated the evolution of the OSS developer network using our model to compare against the empirical data. In simulation, we used several different models to replicate the empirical data: random network theory, scale-free network, scale-free network with constant fitness and scale-free networks with dynamic fitness. From these simulation iterations, we were able to verify and validate our model for this network and to increase our understanding of the OSS developer collaboration network.

1 Introduction

1.1 Open source software phenomenon

Open source¹ software development, despite usually consisting of volunteers dispersed worldwide, now competes with commercial software firms. This competition is due in part to closed-source proprietary software being associated with risks to users. These risks, often referred to as a “software crisis,” can be summarized as systems that take too long to develop, are too costly and do not work very well when delivered. Open Source Software (OSS) may finally present a good solution.

The purpose of Open Source Software was not to ensure distributing software to the end user without cost, but to ensure that the end user could use the software freely (no cost, no limitation, source code available and modifiable). Essentially, as a superset of free software, OSS exists in countless varieties today. Also they all have their own unique histories [8], many of them achieved big successes, for example Linux, Apache.

However, we still don’t know exactly what is going on in the OSS community. The success of OSS software, such as Linux and Apache, lead us to wonder how these developers collaborated with each other to develop software in such a fast, efficient and low-cost way. Understanding the mechanism of the OSS community may help us conceive a new software development model or at least address the software crisis.

SourceForge² is used as the data source of our research since it is one of the largest and most famous OSS hosting web sites, offering features like bug tracking, project management, forum service, mailing list distribution, CVS and much more. As of April 2003, SourceForge hosted more than eighty thousand developers and fifty thousand projects. Thus the study of this community can let us understand more about the Open Source Software phenomenon. Furthermore, by obtaining an inside view and, hopefully, discovering the mechanisms producing the topology and evolution of the OSS collaboration network, we may predict the future of the OSS through modeling and simulation.

1.2 Complex networks can model this problem

Many well-studied networks have apparent generating principles, which means the topology of the network is deterministic by the given generating principle. These kind of networks are called simple networks. But there are also

¹“Open source” is a certification mark owned by the Open Source Initiative (<http://www.opensource.org>).

²<http://sourceforge.net>

large-scale networks to which we can't attribute apparent generating principles, like friendship networks, power grids and natural neural networks. These networks have a mixture of randomness and structure and are called complex networks. Graph theory is the common method to study these networks. The graph used to study complex networks is called a random graph. Complex networks are decentralized and self-organized. The control and order in this kind of network is emergent rather than predetermined. We will use complex network models to study the Open Source Software Community.

During those studies of complex networks, there were several important properties, which were often studied to characterize the topology of the complex networks. These properties are:

1. Degree distribution. The degree of a node, k , equals the total number of other nodes to which it is connected, while $P(k)$ is the distribution of the degree k throughout the network.
2. Diameter. The diameter of a network is the maximum distance (number of hops or edges) between any pair of vertices. The diameter can also be defined as the average length of the shortest paths between any pair of nodes in the network. In our research, the diameter of the network characterizes efficiency because the time to propagate information is proportional to network diameter. Since the average value is more suitable for our purposes, we used the second definition in our research.
3. Clustering coefficient. The neighborhood of a node consists of the set of nodes to which it is connected. The clustering coefficient of a node is the ratio of the number of links to the total possible number of links among the nodes in its neighborhood. The clustering coefficient of a graph is the average of all the clustering coefficients of the nodes.

These three properties will be significant in later discussion.

1.3 What is the challenge

The SourceForge community is one of the largest OSS communities. Our recent research is focused on the exploration of the internal mechanism of the OSS community [14, 15, 16]. The collaboration network of SourceForge is different from the traditional collaboration networks frequently studied (e.g., movie actor networks, or scientist collaboration networks). One of the big differences is the collaboration relationships they address. Traditional, well-studied col-

laboration networks merely discuss the existence of the relationship, while it is the activities within these relationship networks that we are studying. In the collaboration network of SourceForge there were sometimes deletions of edges. In traditional well-studied collaboration networks, once an edge exists, it will remain persistent. For example, in movie actor networks, once an actor acts in a movie, he or she will always have a link to that movie. The link is persistent. The situation is different in the OSS collaboration network. The links between vertices can change. We notice an edge detachment process since a developer can quit the project at any time. Through our research, we also identified SourceForge as a collaboration network.

After obtaining a good theoretical model for the community, the next step was to use simulations to verify and validate it. By having the correct model and parameters, we were able to reproduce the evolution of the network. From this interior view of the network, we could understand, even predict and adjust, the evolution and future developments of the network. .

1.4 Paper layout

The remainder of this paper is structured as follows. Section 2 defines the state of art of related research and existing toolkits, focusing especially on their limitations. Section 3 discusses the data preparation and modeling. Section 4 delves into the simulation procedures and the results. Finally, we give a conclusion of our work in section 5.

2 Related works

2.1 Similar real networks in research

Various real systems, ranging from communication networks to ecological webs, have proven to be complex networks [3]. A good understanding of the topology and evolution of these complex networks can aid in analyzing and understanding other real systems, such as the Open Source Software community. These real networks include WWW, Internet, collaboration network, ecological networks, power grid and phone call network. These real systems exist in different domains, but the networks are similar and they shares the similar properties like “small world” and “power law”³.

³These ideas will be discussed later in this section

2.2 Simulation of social networks

Simulation research about network related system is not a new topic, like photonic interconnection network [7], Peer-to-Peer networks [24], Active networks [20] and Dynamic structure heterogeneous flow system [5].

Social networks have caught researchers' attention recently. Among them, Axelrod has done a good deal of work in simulation of social networks. Part of his research [4] included the replications of eight popular simulation models using Swarm (a simulation toolkit) and comparing the results to the original. All the models are selected for their simplicity, diversity and reasonable run times. As result, Axelrod concluded – “In most cases, the results were so close that we probably attained distributional equivalence for all the models.” In his paper, he also discussed the docking of simulation (Docking is an alignment process and experiment for verifying simulations. detailed definition can be found in [4]).

Prietula, Carley and Gasser [19] discussed the importance of computational modeling and simulation in organizational design, analysis and reengineering, especially in large scale transnational organizations. They also discussed the existing methods in modeling and simulation of social networks like organizations.

2.3 Open Source Software community

The Open Source Software community can be described as a collaboration network. In this network, the nodes are developers and projects. There is a link between a developer and a project if the developer joins the project. This network is similar to the two collaboration networks mentioned earlier (movie actor and scientist collaboration network) because of common features like the bipartite property, small world phenomenon and high clustering coefficient. But the OSS network also differs from these traditional networks. In those two traditional collaboration networks, the edge (link, relationship) is persistent, which means it will never be removed from the network after it was added. This is not the case in the Open Source Software community, where the edge can be removed if the developer (node) quits the project (node). Thus the collaboration network in OSS is based on active relationships instead of once-present relationships found in many collaboration networks. The detachment property is one way that makes the complex network for Open Source Software community different.

Previous research on the complex networks was often based on snapshots [22, 17, 3], which focused on the topology of the network at a given point in time. Since we have collected the data over two years, we were able to look

into the evolution of the network. This helped us understand the evolution of the network topology, the development patterns of any single object in the network and the impact of the interaction among objects to the evolution of the overall network system. The papers related to this research have been published in proceedings of NAACSOS (North American Association for Computational Social and Organizational Science) 2003 [10, 11] and SwarmFest 2003 [12].

2.4 Complex network models

Complex networks are not a new topic in scientific research. There are many existing models used to depict complex networks. In this section, we will review these models to aid in understanding and conceptualizing our work.

2.4.1 Random Network theory

Random network theory, proposed by Erdős and Rényi is the first model to depict complex networks [9]. We will call this the “ER model” in the remainder of this paper. We will go through some relevant terminology and definitions, and a few significant results of this model.

Erdős and Rényi define the random graph as N labeled nodes connected by m edges, which are chosen randomly from the $N(N - 1)/2$ possible edges [9]. There are two similar definitions for random graphs by Bollobás [6].

Definition 2.1 $G(N, m)$ is a labeled graph⁴ with vertex set $V(G) = \{1, 2, \dots, N\}$, having m randomly chosen edges (where m usually depends on N). $G(N, m)$ is frequently abbreviated as G_m .

Definition 2.2 $G(N, p)$ is a labeled graph with vertex set $V(G) = \{1, 2, \dots, N\}$, in which every one of the possible $\binom{N}{2}$ edges exists with probability $0 < p < 1$, independent of any other edges. $G(N, p)$ is frequently abbreviated as G_p .

These two definitions are similar in most cases, and they are practically interchangeable when $m \approx pN$. G_p uses probability p as a parameter and G_m uses m , the number of edges, as a parameter. Since p has a single value of the network, single value for the whole network is much easier to deal with than considering all the edges. We will just use G_p in the following discussions.

⁴labeled graph means a graph with each node labeled differently but arbitrarily.

In a random graph with connection probability p , the degree k_i of a node i follows a binomial distribution with parameters $N - 1$ and p :

$$P(k_i = k) = C_k^{N-1} p^k (1-p)^{N-1-k} \quad (1)$$

Réka Albert and Barabási [3] proved the degree distribution of random network is as:

$$P(k) \approx e^{-pN} \frac{(pN)^k}{k!} = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!} \quad (2)$$

where $\langle k \rangle$ is the average degree of all vertices in the network. Thus with a good approximation the degree distribution of a random graph is a binomial distribution.

And the general conclusion for the diameter D of a connected random graph is that for most values of p , almost all graphs with the same N and p have precisely the same diameter [1], which is given by:

$$D = \frac{\ln(N)}{\ln(pN)} = \frac{\ln(N)}{\ln(\langle k \rangle)}. \quad (3)$$

The diameter of the network is logarithmic to the system size. This means diameter will increase with respect to the size of the network.

Cluster coefficients are another important property. If we consider a node in a random graph along with its nearest neighbors, the probability that two of these neighbors are connected is equal to the probability that two randomly selected nodes are connected [3]. Consequently the clustering coefficient of a random graph is

$$C_{rand} = p = \frac{\langle k \rangle}{N} \quad (4)$$

Random network theory does not fit the real networks in diameter, clustering coefficient and degree distribution.

2.4.2 Small world

Watts and Strogatz [23] found the cluster coefficients of real networks similar to large ordered lattices. Based on this observation, they proposed their complex network model, called the “WS model” in the rest of this paper. A much-studied variant of the WS model was proposed by Newman and Watts [18], in which edges were added between

randomly chosen pairs of nodes, but no edges were removed from the regular lattices.

The diameter of the WS model is dependent on the system size. The diameter is logarithmic to the system size N . This is called the small world phenomenon, which has become so familiar recently [21].

WS models also have a relatively high clustering coefficients (CC). The related equation of clustering coefficient for the WS model is

$$CC(p) = \frac{3K(K-1)}{2K(2K-1) + 8pK^2 + 4p^2K^2} \quad (5)$$

where K is the degree of each node in the original lattice and p is the probability of rewiring edges [3]. This clustering coefficient is independent of the system size and it is relatively high, which matches the results obtained from the real networks.

Next, consider the degree distribution, which contains a discrepancy between the WS model and the real-world networks. As Albert and Barabási [3] discussed, in the WS model for $p = 0$, each node has the same degree K . Thus the degree distribution is a delta function centered at K . After randomization, the degree distribution maintains the average degree equal to K . They have also calculated the distribution as

$$P(k) = \sum_{n=0}^{f(k,K)} C_{K/2}^n (1-p)^n p^{K/2-n} \frac{(pK/2)^{k-K/2-n}}{(k-K/2-n)!} e^{-pK/2} \quad (6)$$

for $k \geq K/2$, where $f(k, K) = \min(k - K/2, K/2)$.

The shape of the degree distribution is similar to that of a random graph. It has a pronounced peak at $\langle k \rangle = K$ and decays exponentially for large k . The WS model does not fit the real networks in degree distribution.

2.4.3 Scale free network

From inspection of several real networks, Barabási found that many large networks are scale free, that is, their degree distribution follows a power law for large k . The ER model and WS model do not reproduce this feature.

The power law degree distributions observed in networks were investigated by Barabási and Albert [2]. They proposed a new network model, called the BA model in the remainder of this paper. They proposed this new model based on the idea that the scale-free nature of the real networks is rooted in two generic mechanisms – growth and preferential attachment.

Numerical simulations indicated that this network evolved into a scale-invariant state with the probability that a node had k edges, which followed a power law with exponent $\gamma_{BA} = 3$. This is called the “power law distribution,” which exists in many real networks but not in any networks generated by the ER or WR model.

Barabási and Albert also proved that the diameter of the networks generated by their model was smaller than the diameter of a network generated by random network theory with the same size and average degree.

As to the clustering coefficient, there is no analytical prediction in the BA model. Barabási and Albert [3] compared a BA model network with average degree $\langle K \rangle = 4$ to the random network with clustering coefficient $CC_{rand} = \langle k \rangle / N$. They found that the clustering coefficient of a scale free network was about five times higher than that of a random graph, and this factor slowly increased with the number of nodes.

3 Empirical data processing

3.1 Empirical data preparation

To complete our research, we collected empirical data from SourceForge. We collected data from SourceForge every month ever since January 2001. Every month we collected dataset in pair of DeveloperID and ProjectID. As of October 2003, we collected over 2 millions records.

3.2 Statistical observation in empirical data

3.2.1 Average degree is increasing

Average degree $\langle k \rangle$, which gives the average number of links per developer is a good quantitative measurement for the connectivity of a graph. There is an approximately linear increase of $\langle k \rangle$ with time in the empirical data. The increase of $\langle k \rangle$ may come from two changing parts in the network. First, the number of nodes in the network increased with time due to the arrival of new developers. Second, the total number of edges also increased through the collaborations made by new developers with old ones and by new collaborations made by old developers. These two changes act together to increase the $\langle k \rangle$.

3.2.2 Cluster distribution also follows the power law

A cluster is defined as the maximal subset of connected nodes (a path exists between any pair of nodes). It is important to realize that the developer network is composed of many clusters. There are two reasons for this. First, there are developers that do not collaborate at all, i.e., each of them is the only developer of a certain project. We defined these as isolated clusters, and the percentage of isolated clusters in the whole network is not negligible (about 30%). Second, the links between developers are connected by projects. As long as a project is still active, developers may still collaborate with more newcomers to the project. But the life of a project is finite (about 65% of the new projects in July 2001 died in less than 9 months), so the collaborating relationship between developers changed with time.

Since the cluster is an important property of a developer network, investigation of the cluster is helpful in our understanding of the network. We found that the cluster distribution after log-log transformation fits a straight line quite well without considering the biggest cluster (which will be called the major cluster in latter discussion).

3.2.3 Diameter decays with time

The diameter of the developer network is a good measure of network communication ability. A shorter diameter results in fewer average steps needed for one developer to spread a message to another developer and less time needed for an idea to spread through the network. The developer network has a small diameter, which was calculated in a previous section. Also we investigated the evolution of the diameter of the network, which is decreasing with time.

This decreasing trend could have three reasons. First, preferential attachment will cause a significant increase in the degree of hubs in the network (the vertices with more links than average in the network), thus decreasing the diameter. Second, the increasing arbitrary size of a major cluster means the connected component may weight more in the overall system, thus increasing the overall connectivity of the network. Finally, the diameter decreases as internal links (for example, old developers joining projects that previously existed) increase the interconnectivity of the network, thus decreasing the diameter.

3.2.4 Clustering coefficient decays with time

Clustering is another important characteristic of the topology of real networks. So the clustering coefficient, a quantitative measure of clustering, CC , is also an attribute we investigated. We can also observe the decaying trend of the

clustering coefficient. The clustering coefficient for a developer network tells us how much a node's co-developers are willing to collaborate with each other, and it represents the probability that two of its developers are collaborating on a project together. Thus with the evolution of the developer network, more edges (collaboration relations) are formed. This will lead to a decrease in the probability that two co-developers will join a new project together due to the fact that they are participating on a number of projects approaching their limits. Furthermore, this leads to the decrease in the clustering coefficient.

3.3 Modeling

As discussed in previous sections, our collaboration network of SourceForge is different from many other types of collaboration network in regards to detachment (detachment means a developer quits a project). We made our model based on the bipartite graph (the collaboration network of SourceForge) so we would be able to describe the detachment more straightforwardly.

3.3.1 Model description

The model for simulation is like a procedure definition with related parameters. In our simulation, there were two kinds of entities – developer and project. Developers, which are the active roles in real SourceForge network, will also be the active entities (agents) in our model. Our simulation is a time-step simulation, which means that every agent in the simulation is triggered to act at every time step. We define the time step in our model as one day in the real world.

At every time step, there was a given number of new developers, $N(ND)$, added into the simulation. In our model, the actual number of new developers at every time step was a random number generated by a uniform distribution averaged at $N(ND)$. Then every existing developer (existing and new developers) in the simulation was triggered to act one by one. The procedure definitions for new developer and existing developer were different, which we will discuss separately.

- *New developer.* Every new developer is triggered to act right after it is generated. There are two possible actions for a new developer – creating or joining. Creating meant that the developer would create a new project and thus add a link from this developer to a new project. Joining meant that the developer would select an existing project according to some rules, and thus a link from this developer to the project was added. $P(CN)$ and $P(JN)$ are

the two parameters that controlled the probability of creating and joining. The one exemption was that the very first developer in the simulation had to create a new project because there were no existing projects in the system yet.

- *Existing developer.* Every existing developer was triggered to act at every time step. There were four possible actions for an existing developer – creating, joining, abandoning and idling. Creating and joining were similar to the definitions for the new developer. Abandoning meant that the developer would abandon a randomly selected project in which he had participated. Idling meant that the developer did nothing, which is normally the most frequent action a developer will take. $P(CO)$, $P(JO)$, $P(AO)$ and $P(IO)$ are the four parameters to control the probability of creating, joining, abandoning and idling. The probabilities that a developer would choose a certain action are different in different models for simulations based on different considered attributes.

This is just the skeleton of the procedure definitions in our models. Different simulation models may have different procedure definitions. These differences focused on the rules a developer used to choose an action and the rules a developer used to decide what project to join. Also the related parameters may be different for different simulation iterations. We will explain these special procedure definitions and related parameters when we discuss that specific model.

4 Simulation procedure and results

According to this framework, we ran the simulations in an iterative fashion. This means we started from a basic model, running a simulation based on that model, and then verified and validated it using the empirical data. After adjusting the model and related simulation parameters, we ran the simulations again. We repeated the iterations until we got the “fit” model for the empirical data. Now we will discuss the details of our simulation models one by one.

4.1 Model one: random network (adapted ER model)

We started from the ER model since it is the first well-known complex network model and was the base of the others. This is the first model that employed nonlinear dynamics and displays some extraordinary properties caused by phase transition. One of them is the existence of well-defined boundaries that separate order from disorder [13].

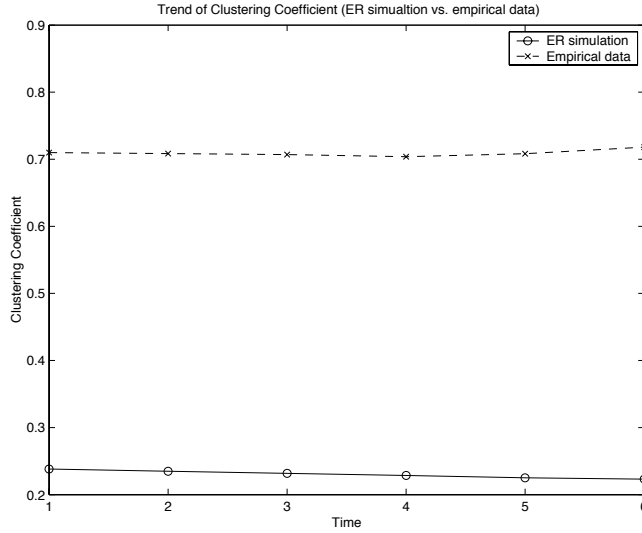


Figure 1: Clustering coefficient in ER: unit for X coordinate is month, and 1 represent July 2002

The network in this model is growing, which is different from the traditional ER model. So we call it an adapted ER model. But randomness is the dominant factor in the procedures of the adapted ER model. The rules a developer used to choose an action and the rules a developer used to decide what project to join are all based on randomness.

After running the simulation for the ER model, we validated it by comparing the simulation output with existing theoretical results and verified it by comparing the simulation output with the empirical data. The first property we investigated was the clustering coefficient. The clustering coefficient (CC) of the simulated ER model is shown in Figure 1. The clustering coefficient of the ER model is below 0.24 (presented as ‘o’ in the figure), which is much smaller than that of the empirical data of SourceForge developer network (where $CC \sim 0.75$, which are presented as ‘x’ in the figure). Also, it diminished as the overall system size increased. The relation between clustering coefficient CC and system size N (between 21,200 and 35,320) is approximately

$$CC \sim N^{-0.14} \quad (7)$$

This observation is identical to the mathematical result for the clustering coefficient of the ER model [3], which proved $CC \sim N^{-\beta}$, where β is a constant smaller than 1.

From the previous section, we know that the clustering coefficient of SourceForge is relatively high at around 0.7 and decreasing. So the CC result of ER simulation is different from the empirical data.

Average degree and diameter are two other attributes we investigated in the ER model. The average degree and the diameter of the simulated ER model are shown in Figure 2. The average degree of the simulated data is decreasing while the diameter of the simulated data is increasing ('x' in Figure 2). In theoretical ER models, the average degree should decrease while the size of the network increases and the mathematical expression for the diameter of an ER model [3] is like

$$D \sim \frac{\log(N)}{\log(\langle k \rangle)}, \quad (8)$$

so the average degree and diameter of simulated data qualitatively agree with the mathematical results.

From the previous section, we know that the average degree should increase and the diameter of the network should decrease as the network grows based on SourceForge developer collaboration network data ('+' in Figure 2). The arbitrary value of the average degree of SourceForge (which is bigger than 7) is bigger than the average degree of an ER model (which is less than 6) while the value of diameter of SourceForge (which is bigger 6) is also bigger than the diameter of the ER model (which is less than 3). From the experimental data on the diameter, it is clear that the ER model does not fit the SourceForge collaboration network well.

Then we studied the cluster distribution. The cluster distribution of the simulated ER model is shown in Figure 3. The upper figure represents the cluster distribution with linear coordinates. There is one big cluster present, which is proven to appear in ER models at some time during evolution. The process of generating this cluster is called percolation, which we discussed in previous section. The lower figure is the cluster distribution after log-log transformation. The data points do not fit the linear regressions in both the empirical data and the simulated data (R^2 for empirical data is 0.5556 and R^2 for simulated data is 0.4445) due to the major cluster. Without considering the major cluster, the linear regression of cluster distribution of ER model has a similar slope to that of the empirical data and the R^2 values are quite good (R^2 for empirical data is 0.9598 and R^2 for simulated data is 0.9906). But the actual data points are different. The following simulated models all have similar slopes of linear regressions to the empirical data in regards to cluster distribution and data points. We will not discuss this in the following models, however, the related simulation results are available upon request.

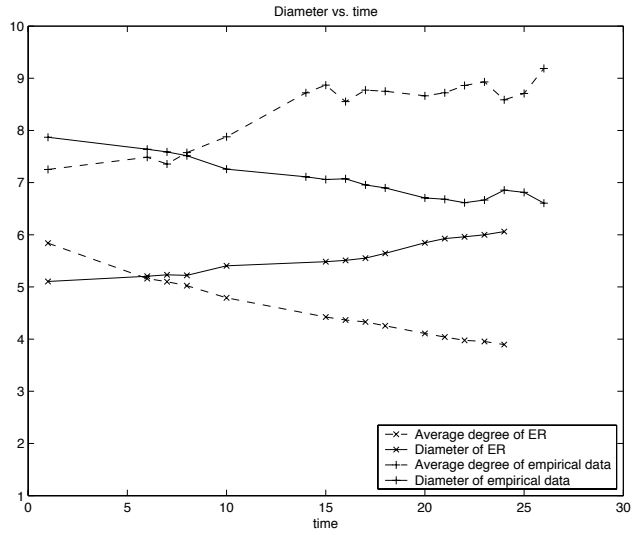


Figure 2: Average degree and diameter in ER simulation and the empirical data: unit of X is month, 0 represents December 2000 and simulated time 0

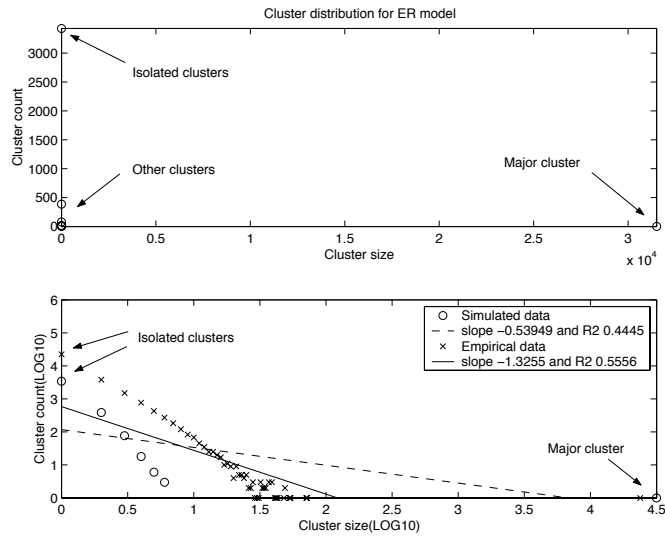


Figure 3: Cluster distribution in ER simulation and the empirical data

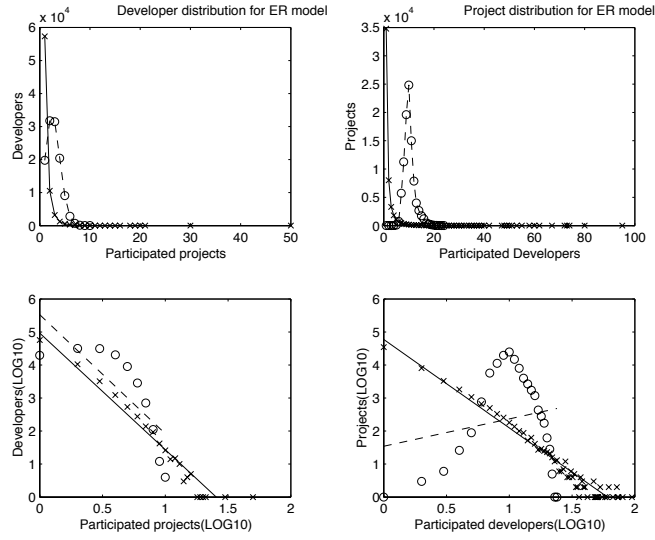


Figure 4: Developer and project distributions in ER simulation and the empirical data: ‘x’ represents empirical data and ‘o’ represents simulation data of ER model

Finally, we investigated the degree distribution of ER model. The simulation results of the ER model are shown in Figure 4. The upper two figures are developer distribution and project distribution in normal coordinates. There is no power law in the distribution. The distributions look more like the mathematically proven Poisson distribution. Also, we compared them with the distributions of empirical data in log-log coordinates, which are shown in the lower figures. For empirical data, the developer and project distributions fit a straight line well, characterizing the power law distribution. We found a linear regression with a slope of -3.5311 and a R^2 of 0.9533 for the developer degree distribution; we also found a linear regression with a slope of -2.6781 and a R^2 of 0.9633 for the project degree distribution. The ER model has distributions that are significantly different from that of our SourceForge collaboration network. We found the developer degree distribution has a R^2 as 0.619 and the project degree distribution has a R^2 of 0.0407. Also, we can see that in project distribution, the maximum number of developers in a single project is only 27, which is much less than the result of empirical data, which are 94. This is because the projects shared the same probability of being chosen, and no project had a preferential advantage to obtain significantly more developers. We validated the simulation of ER model with mathematical results, and we showed by simulation that the ER model is a poor model for the SourceForge developer collaboration network.

4.2 Model two: scale-free network (BA model)

The scale-free network model is the second model in our simulations. There are two major improvements made in the scale-free network: growth and preferential attachment. Since growth is already included in our simulation of adjusted ER model, so we just needed to add preferential attachment to our simulation.

In the model procedure definition, preference will influence both the rules that a developer used to choose an action, which is called developer preference, and the rules that a developer used to choose what project to join, which is called project preference. For project preference, the probability that a project would be selected was proportional to its degree. The probability of a project i being selected is defined as

$$P(i) = \frac{k_i}{\sum_{j \in AllProjects} k_j} \quad (9)$$

For developer preference, we suppose that the more projects in which the developer participated, the higher probability he would take an action (create, join or abandon) as opposed to being idle. So in our simulation, the degree the developer already had was also one of the factors that determined the developer's action probabilities. In implementation, a probability in $[0,1]$ was generated first, and the action selection decision was based on which range (every action has its own probability range) the probability was in. The four action ranges are changing according to the developer degree.

First, we investigated the diameter and the clustering coefficient of simulation data based on the BA model. The results are shown in Figure 5. The diameter is around 7 and decreasing. The clustering coefficient is around 0.7 and decreasing. They both adequately match the empirical data.

Then we investigated the degree distributions of this simulation iteration – BA model. The developer and project distributions of our simulation for the BA model are shown in Figure 6. In the figure, the upper two figures are the developer distribution and project distribution in normal coordinates and the lower two figures are their distributions in log-log coordinates. In the lower figures, the “x” points are the distribution of empirical data and the solid line is its linear polynomial fit; the “o” points are the distribution of simulated data and the dashed line is its linear polynomial fit. We observe that for the developer distribution, the simulation data fits the empirical data quite well. The function of the linear regression for simulated data is $y = -4.1803x + 5.1833$ and R^2 is 0.96. For the project distribution,

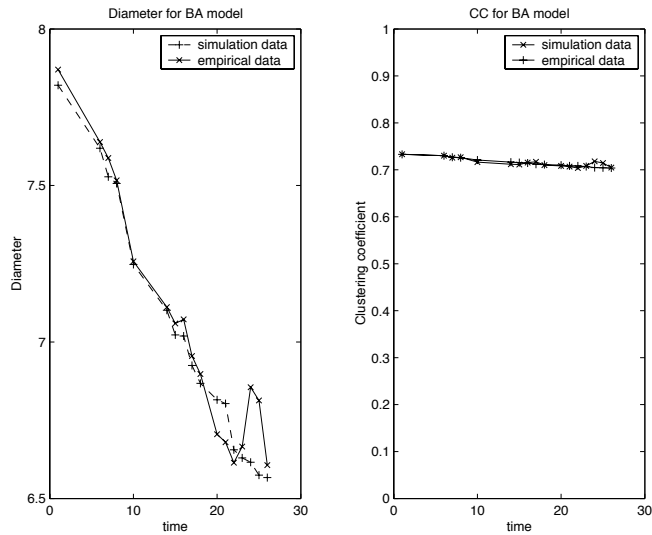


Figure 5: Diameter and clustering coefficient in BA simulation and the empirical data: unit of X coordinate is month and 0 represent December 2000

the data points and the linear regression, however, do not fit well. The function of linear regression for simulated data is $y = -1.2454x + 2.5823$ and R^2 is 0.442. The discrepancy between the linear regressions of the simulation data and empirical data occurs because of the tails, especially for the largest project. We can find that the largest project in simulated data includes almost 8000 developers while the largest project in empirical data just includes about 100 developers.

Also there exists the phenomenon that young nodes were possibly out-running old ones (for example, Google search engine overran the Yahoo! search engine in just a few years) in the real network. But the BA model is not capable of reproducing this phenomenon.

So this model can reproduce most of the topological statistics we observed in empirical data, including the power law in both developer distribution and project distribution. But the project degree distribution does not ‘fit’ the empirical data and it can not reproduce the “young upcomer” phenomenon.

4.3 Model three: scale-free network with constant fitness

The scale-free network with constant fitness is the third model in our iterations. This model introduced a new attribute – fitness to reproduce the “young upcomer” phenomenon. Fitness is an appended attribute proposed by Barabási for

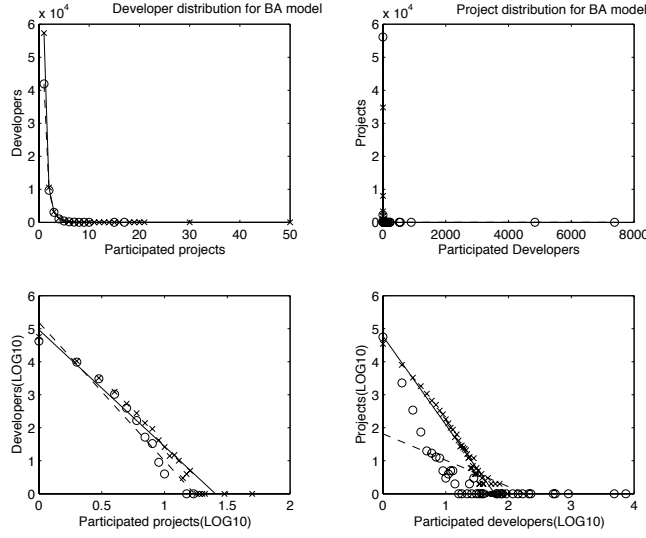


Figure 6: Developer and project distributions in BA simulation and the empirical data: ‘o’ represents simulation data of BA model and ‘x’ represent empirical data

the Scale-free network model to depict the “young upcomer” phenomenon. Considering the above deficiency, we studied the BA model with constant fitness as our third model. We defined the fitness as a constant parameter η_i for every project. Thus every time a new project (say, project j) is created, a fitness η_j was added to the system, where η_j is chosen from a distribution $\rho(\bullet)$. The probability of participating in a project i is proportional to the degree and the fitness of the project i ,

$$P_i = \frac{\eta_i k_i}{\sum_j \eta_j k_j} \quad (10)$$

The introduction of fitness can make the preference based on both fitness and degree. So this preferential model is capable of explain the “young upcomer” phenomenon.

We also demonstrated that this model has power law distributions for developers and projects in simulation and the results are shown in Figure 7. The upper figures are the comparison of developer and project degree distributions between empirical data and simulated data in linear coordinates. Then we applied log-log transformation on the data points and the results are shown in the lower figures. The R^2 for developer degree distribution in simulated data is 0.9559 and the R^2 for project degree distribution in simulated data is 0.5261. So the simulated data fit the empirical data much better on the developer degree distribution than on the project degree distribution. One of the significant

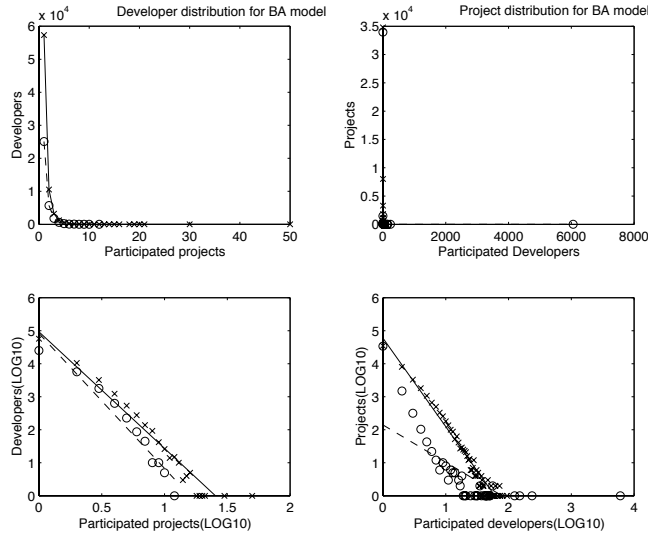


Figure 7: Degree distributions in BA simulation with constant fitness and the empirical data: ‘o’ represents simulation data of BA model with constant fitness and ‘x’ represent empirical data. For developer distribution, function of linear regression is $y = -4.1860x + 5.1876$ and R^2 is 0.9559; for project distribution, function of linear regression is $y = -0.8008x + 1.8169$ and R^2 is 0.5261

differences is still the size of the largest project, which is around 6000 in the simulated data while it is around 100 in the empirical data.

Then we investigate the diameter and clustering coefficient of the simulated data. We demonstrated that the actual values and evolutions of the diameter and the clustering coefficient of this model match those of the empirical data.

So the BA model with constant fitness not only matches most of the statistics of the empirical data but also is capable of explaining the “young upcomer” phenomenon. It still has some deficiency in reproducing the same result as the empirical data especially in project distribution. Further iteration was needed.

4.4 Model four: scale-free network with dynamic fitness

In the previous iterations, we discovered that scale-free networks with constant fitness were not able to match the project degree distribution to the empirical data.

We propose a dynamic fitness factor in the scale-free network, which allows the fitness for every project to decay with respect to time. In our model, we made the fitness decay a function of time. The rate of decay is different for all projects. Every project possessed a random fitness when it was created by a developer. The generation of fitness for

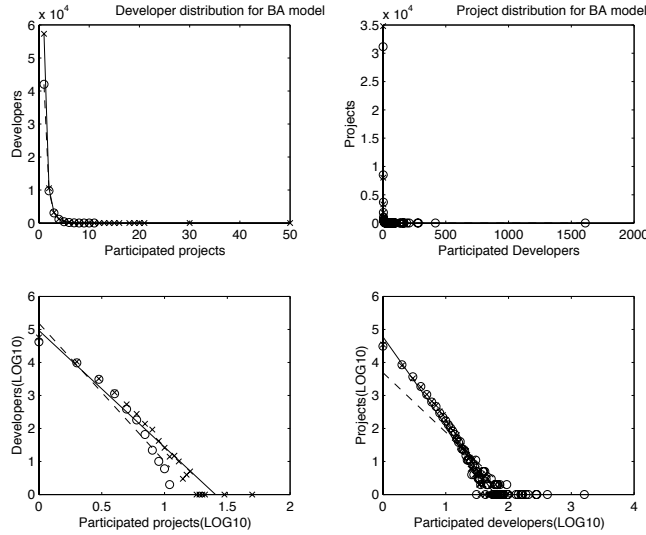


Figure 8: Degree distributions of BA simulation with dynamic fitness and the empirical data: 'o' represents the data points from the simulated data and 'x' represents the data points from the empirical data

new project followed an exponential distribution.

The results of the degree distributions of the BA model with dynamic fitness are shown in Figure 8. The upper figures are the comparison of developer and project degree distributions in linear coordinates. The lower figures are the comparison of developer and project degree distributions in log-log coordinates. The R^2 of developer degree distribution from the simulated data in lower figure is 0.959 and the R^2 of project degree distribution from the simulated data in lower figure is 0.7657. Also the largest project size of the simulated data is just 1500. We can further lower this value by tuning the fitness parameter and function.

Also we proved that this model could match the other statistics (diameter and clustering coefficient) of the empirical data.

After analyzing the simulation output of the BA model with dynamic fitness, we found that our model reproduced the network similar to and even better than the BA model with constant fitness, in relation to the attributes that we investigated (degree distribution, diameter and clustering coefficient).

In this section, we used simulation methods to understand the topology and evolution of the SourceForge developer collaboration network. First we proposed a framework for simulation related study. Following this framework, we iterated all the well-known models (ER model, BA model and so on) in complex network with similar parameters to

generate the “virtual” SourceForge. By comparing the simulation outputs and the empirical data, we confirmed the results of BA model with dynamic fitness and also verified the correctness of our proposed model. The results in this section are summarized in Table 1.

Table 1: Summary of simulation results

Model	Parameter	expected pattern	observed pattern
ER	Developer distribution	Power law	Normal
	Project distribution	Power law	Normal
	Cluster distribution	Power law	Power law
	Average degree	Increasing	Decreasing
	Clustering coefficient	Decreasing (large value)	Decreasing (small value)
	Diameter	Decreasing	Increasing
BA	Developer distribution	Power law	Power law
	Project distribution	Power law	Power law (heavy tail)
	Cluster distribution	Power law	Power law
	Average degree	Increasing	Increasing
	Clustering coefficient	Decreasing (large value)	Decreasing (large value)
	Diameter	Decreasing	Decreasing
	“Young upcomer”	Existing	Not existing
BA with constant fitness	Developer distribution	Power law	Power law
	Project distribution	Power law	Power law (heavy tail)
	Cluster distribution	Power law	Power law
	Average degree	Increasing	Increasing
	Clustering coefficient	Decreasing (large value)	Decreasing (large value)
	Diameter	Decreasing	Decreasing
	“Young upcomer”	Existing	Existing
BA with dynamic fitness	Developer distribution	Power law	Power law
	Project distribution	Power law	Power law (small tail)
	Cluster distribution	Power law	Power law
	Average degree	Increasing	Increasing
	Clustering coefficient	Decreasing (large value)	Decreasing (large value)
	Diameter	Decreasing	Decreasing
	“Young upcomer”	Existing	Existing

5 Conclusion

5.1 Summary

We have discussed how Open Source Software (OSS) development is a classic example and prototype of a collaborative social network. Based on the empirical data we collected from SourceForge over the last two years, we were able to investigate the structure and the dynamic mechanisms that determined the topology and governed the evolution of such systems. SourceForge is one of the largest online collaborators for open source development projects, hosting over 50 thousand projects and over 80 thousand developers. Our empirical data on projects and developers was collected monthly beginning in January 2001. In this paper, we analyzed the empirical data we collected from SourceForge in order to obtain statistics and topological information of the OSS developer collaboration network. We extracted the parameters of the evolution by inspecting the longitudinal properties of the network. We generated a model that depicted the evolution of this collaboration network. Finally, we used simulation to verify and validate the models for the SourceForge community.

In studying the simulation model of the SourceForge collaboration network, we used the parameters we extracted from empirical data as the simulation parameters and iterated from the ER model, BA model and BA model with constant fitness until finally arriving at the proposed BA model with dynamic fitness. Through simulation, we verified all the properties of these models that could be calculated by mathematical methods (degree distribution) and also obtained the other properties that were difficult for mathematics to calculate (diameter, clustering coefficient and developing patterns). Then by comparing the simulation output with the empirical data, we concluded that our proposed model, BA model with dynamic fitness, is the fittest model for the SourceForge collaboration network because it could reproduce all the properties investigated in this research.

5.2 Limitations

There were several limitations in our work, which are listed as follows:

- Although SourceForge is one of the biggest hosting sites in the OSS community, this is only a single case. Studies of a single case could not reveal the common topology and evolution properties of the entire OSS community.

- Simulations of complex networks are too complicated for simulation toolkits to provide the user all the tunable parameters, so these toolkits always have some default setting for some simulation parameters that the user can not change. Thus using single simulation toolkits may also bring unexpected errors into the simulation output. We just used Swarm to run the simulation. This may cause some bias in the output.

5.3 Future work

According to the limitations of our work, there may be several future works as follows:

1. We need more complete empirical data for experiments by continuing to collect data from SourceForge.
2. We may analyze other famous OSS hosting site, like Savannah, using the same methods.
3. We could migrate our simulations to Repast to verify the inference of simulation toolkits to the simulation output.
4. We could use more advanced methods like operation research optimization in fine-tuning the parameters to find the best solutions.

References

- [1] W. Aiello, F. Chung and L. Lu. Random evolution in massive graphs. *IEEE Symposium of Foundations of Computer Science*, 2001.
- [2] R. Albert and A. Barabási. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
- [3] R. Albert and A. Barabási. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(47), 2002.
- [4] R. Axelrod. *The complexity of Cooperation: Agent based models of competition and collaboration*. Princeton University Press, Princeton, NJ, 1997.
- [5] F. Barros. Modeling and simulation of dynamic structure heterogeneous flow system. *Simulation: Transactions of the society for modeling and simulation international*, 78(1), 2002.
- [6] B. Bollobás. *Random graphs*. London:Academic, 1985.
- [7] R. Chamberlain and C. Baw. Evaluating the performance of photonic interconnection networks. *Proceedings of Annual Simulation Symposium*, pages 209–218, 2002.
- [8] G. Drummond. Open source software and documents: A literature and online resource review. [http : //www.omar.org/opensource/litreviewa](http://www.omar.org/opensource/litreviewa), 1999.
- [9] P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

- [10] Y. Gao, V. Freeh and G. Madey. Analysis and modeling of the open source software community. *NAACSOS, Pittsburgh*, 2003.
- [11] Y. Gao, V. Freeh and G. Madey. Conceptual framework for agent-based modeling and simulation. *NAACSOS, Pittsburgh*, 2003.
- [12] Y. Gao, V. Freeh and G. Madey. Topology and evolution of the open source software community. *SwarmFest, Notre Dame*, 2003.
- [13] S. Kauffman. *Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, Oxford, 1993.
- [14] G. Madey, V. Freeh and R. Tynan. Agent-based modeling of open source using swarm. *Eight Americas Conference of Information Systems*, 2002.
- [15] G. Madey, V. Freeh and R. Tynan. The open source software development phenomenon: an analysis based on social network theory. *Eight Americas Conference of Information Systems*, 2002.
- [16] G. Madey, V. Freeh, R. Tynan and Y. Gao. Agent-based modeling and simulation of collaborative social networks. *AMCIS, Tampa*, 2003.
- [17] M. Newman. The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci*, 98:404–409, 2001.
- [18] M. Newman and D. Watts. Random graph models of social networks. *Physics Review*, 64(026118), 2001.
- [19] M. Prietula, K. Carley and L. Gasser. *Simulating organizations: computational models of institutions and groups*. MIT Press, Cambridge, MA, 1998.
- [20] D. Rao and P. Wilsey. Modeling and simulation of active networks. *Proceedings of Annual Simulation Symposium*, pages 177–184, 2001.
- [21] D. Watts. *Small world*. Princeton university press, NJ, 1999.
- [22] D. Watts and S. Strogatz. Collective dynamics of small-world. *Nature*, 393(440), 1998.
- [23] D. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.
- [24] K. Zerfiridis and H. Karatza. Dissemination scenarios in peer-to-peer networks. *Proceedings of Annual Simulation Symposium*, pages 309–316, 2003.