Nathan Oostendorp; Mailbox #200
SI708 Networks Theory and Application
Final Project Report

Using Networks to Visualize and Understand Participation on SourceForge.net

SourceForge.net is an online repository for open source development that provides tools, file distribution, and web space for open source software developers. While it has become an establishment for distribution of open source software, it has also been used as a data resource for those interested in studying the nature of open source development. In the existing work there has been little discussion of the differences in the modes of participation between members and non-members of open source development projects. Our interest in performing a network analysis of SourceForge.net was to better understand the participation of users between projects, and how the activity of project members, that is those involved with the production of a software project contrasted with the contributions of non-members, typically those who are consumers of the software.

Much of the previous work on the SourceForge.net data set has been by Greg Madey from Notre Dame, who hosts the SourceForge.net research data for academic use. He and many collaborators have done numerous analyses of participation and community structure in the SourceForge data set. The topics explored include more in-depth looks at activity as a whole [1] developing models to simulate the developer network [3], and analysis of the membership networks [2]. David and Rullani [4] proposed that a probability matrix could describe for possible changes in user behavior on SourceForge over time. Separately, many people have used the SourceForge data set to analyze behavior, many of which are listed at MIT's open source research community site (http://opensource.mit.edu/).

In this project, using a copy of the SourceForge.net database from September 2006 a data set was rendered giving a representation of user participation in different projects. From this conditioned data set, large affiliation networks were rendered and analyzed. These were then decomposed into related clusters and these clusters were extracted from the

network and displayed in Guess, which allowed for observation of some of the properties of user participation. Additionally, the time dimension of the data was used to create an animation of network growth in these clusters, which brought out other observable traits of participation in a project cluster's lifetime.

Methods

From the SourceForge.net database, we identified the major data types and extracted similar fields from each the different data types including bug reports, forum posts, project news items, project documentation, screenshots, file releases, help wanted postings, and task assignments. This technique is similar to work previously described by Christley and Madey [1] on the SourceForge.net data set. We also included in each record the membership status of the user in the project they were participating in. The result is a file containing roughly 6.2 million contributions to SourceForge projects, ranging from November 1999 to September 2006. Table 1 illustrates an example record.

Project id	User id	Data type	Timestamp	Project Member
1435	9344	Forum post	Mar 12, 2003 11:35:23	True

Table 1: a typical record in the SF.net participation data set

Using this data, we created two undirected bipartite affiliation networks: one for members and one for non-member participations. In this case the two sets of nodes in each network were projects and users and the weight of the edge between them indicated the number of database records that each user had contributed. Not included were anonymous posts, and participations in the "SourceForge" project which acts as a support ticket tracker for the site's own web application.

The networks were analyzed using the Pajek software application, in an attempt to see if on the aggregate level these graphs exhibited substantially different traits. Each network was analyzed both as a bipartite network and projected onto undirected weighted one-mode networks where edge weights represented the number of participants each group had in common. Some of the raw data is illustrated in table 2.

	Member Participation Network (bipartite)	Non-member Participation Network (bipartite)
Number of Nodes	169,067	541,098
Number of Edges	113,172	560,621
Average Degree	1.34	2.08
Connected Components (>2)	57,250	18,540
Size of Giant Component	13,985	380,792
Average Edge Weight	25.2	3.42
Total Contributions	2,849,763	1,917,317
	Member Participation Network (one-mode)	Non-member Participation Network (one-mode)
Number of Nodes	83,610	142,312
Number of Edges	48,094	1,873,958
Average Degree	1.15	26.3
Connected Components (>2)	11,607	1084
Size of Giant Component	5,457	28,384
Average Edge Weight	1.03	1.09
Average Clustering Coef	0.213	0.130
Stddev Clustering Coef	0.401	0.310

Table 2: Raw data for the Member and Non-member Networks

Immediately it's apparent the two networks are substantially different. While the member network is smaller in terms of vertices and edges, it represents more participation in the site overall. The size of the giant component is much smaller in the member network, and the number of unconnected isolates is higher.

From this we can draw a few conclusions about member and non-member behavior: Members contribute more to their own projects individually than non-members do individually shown by the average edge weight being much higher. That members generally work on fewer projects as members is exhibited by the lower average degree in one-mode network. The substantially higher edge density in the one-mode network indicates that non-members are more likely to participate in more projects as non-members.

Unlike Xu et al [2], we found a giant component in the member network rather than several large components despite the fact that we used more conservative criteria for

membership (members must actually have participated in our data types, rather than simply be in the membership table). This may be due in part to our analysis of the network from a 2006 database rather than a 2003 databrase.

The edge weights and degree distributions of the member and non-member bipartite networks both followed power law distributions closely, but with different logarithmic slopes.

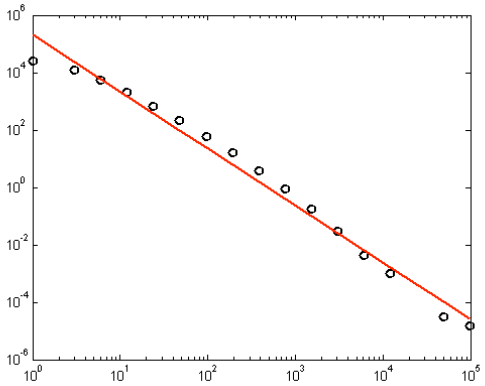

Fig 1. Member Bipartite Edge Weights
Log-binned $\alpha = 1.98$

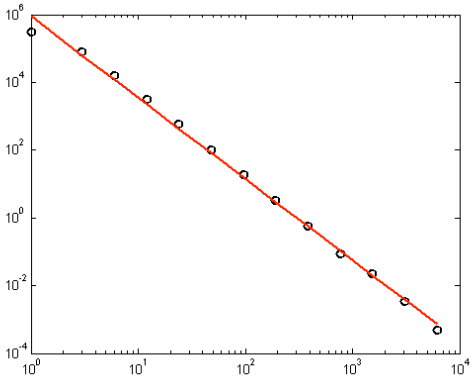

Fig 2. Nonmember Bipartite Edge Weights
Log-binned $\alpha = 2.40$

Figure 1 and 2 show the log-binned distribution of edge weight frequency in the two networks, representing different amounts of participation in each of the user-group pairs. In this case, the member network has more frequent “high weight” edges.

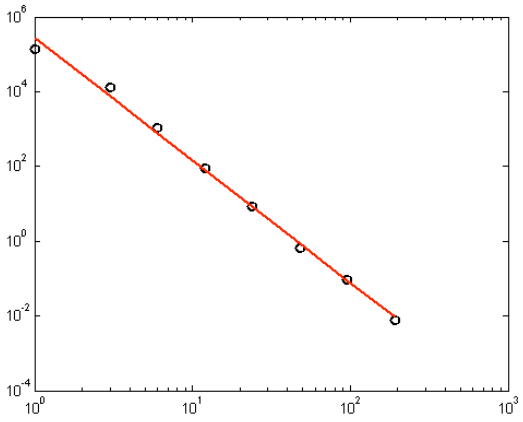

Fig 3. member Bipartite Degree distribution $\alpha = 3.27$

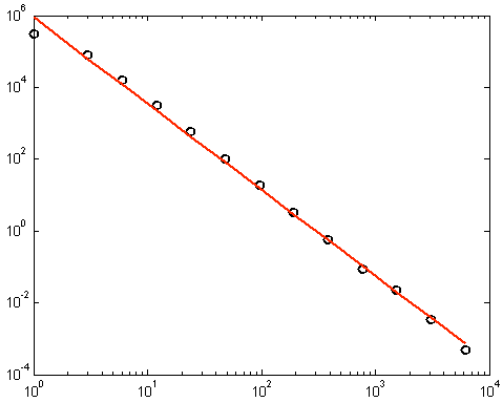

Fig 4. Non-member Bipartite Degree distribution $\alpha = 2.21$

Figure 3 and 4 show log-binned node degree frequency in the bipartite networks, and these also follow power law distributions closely. The degree in this case indicates both multiple participants in the same group, or multiple groups participated in as a member. The falloff in degree frequency is much sharper in the member network.

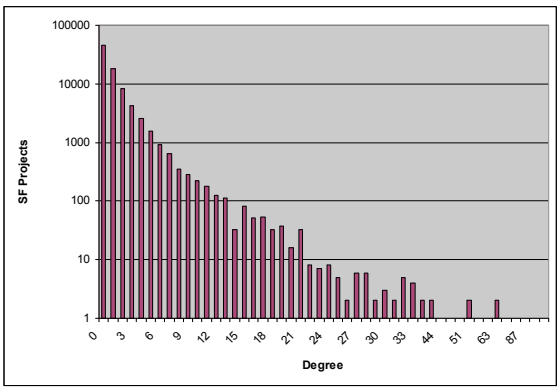

Fig 5. Degree of nodes in member 1-mode projection.

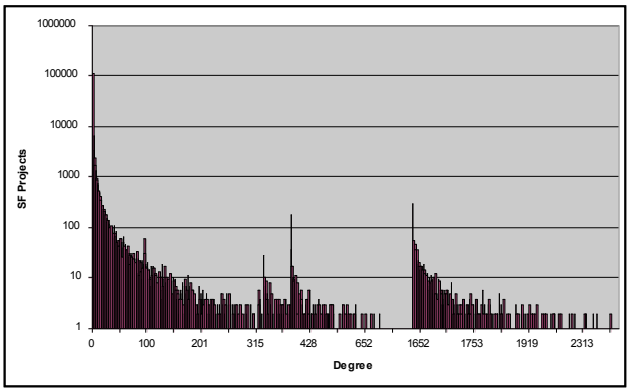

Fig 6. Degree of Nodes in non-member 1-mode projection.

When we investigated the project-wise one-mode projections of these networks, we found the degree distribution did not fit a power law curve. Figure 5 and 6 show degree distributions for these 1-mode networks, with the X axis on a linear scale and Y on a logarithmic scale. In the member network, more than 50% share no members with other

projects, and the degree falls quite sharply to 0 in the mid-double digits. In the non-member network 78% share no non-member participation with other projects, but there are quite a number of high degree nodes, with a substantial amount of noise. One curious finding is the gap followed by a large spike around 1600 links, which may indicate a near-fully connected core of older projects.

Finding Smaller Components

While the analysis of the networks as a whole provided some interesting features, we wanted to take a little closer look at some particular groups of projects to see if visualization could bring out the dynamics of participation. Since this was an affinity network, we were able to use m-slices to find components in the networks that had cross-user participation.

In the members network, the 2-slice completely fragmented the giant component forming 481 clusters, the largest of which were 31 and 28 nodes, with 92% of components less than 5 nodes. In the non-members network, m-slicing created some components, but did not completely fragment the giant component. This giant component was preserved well up to a 32-slice, although it then contained only 192 nodes. This was consistent with the much higher degree distribution in the non-member one-mode network. We also attempted to fragment the giant component of the member network using the "Fast Modularity" community structure algorithm. This found 96 clusters, with an average cluster size of 56 projects, but with several clusters containing several hundred projects.

Homophily in Clusters

In order to determine if the clusters found were reasonable representations of "communities" we tested the clusters on several homophily measures. The measures used for comparison included:

- the number of words in common between two project descriptions, excepting terms common in the English language.

- the longest common substring between two projects names
- the number of self-selected software map tags in common
- number of members in common between two projects
- difference in registration time between projects

Each cluster was tested as an average "all pairs" comparison – each project was tested against every other project in the cluster and we looked at the average of all those comparisons for each cluster. This did mean larger groups generally tested as less homophilious as opposed to small clusters or pairs of projects. Some of the different results between the two methods are shown in Table 3.

Averages for all pairs comparison of:	2-Slices of member 1-mode projection	MEJN Fast Modularity on member 1-mode giant component
Number of Projects per Cluster	2.60	56.8
Common terms in project description	2.37	0.55
Common software map categorization tags	0.55	0.10
Longest common substring in project name	1.70	0.56
Difference in registration time	402.0	674.3
Number of developers in common	3.53	0.36

Table 3: Averages of all-pairs homophily measures for 2-slices and Fast Modularity clusters

Generally 2-slices resulted in smaller, more homophilous clusters than the fast modularity algorithm, which we infer is partially due to the large discrepancy in cluster size. Looking at the individual clusters, the few small and homophilous clusters found by the fast modularity algorithm were generally extremely sparse. It is possible that if fast modularity used an additional consideration of edge weights, it might derive similar clustering homophily to the 2-slices.

Visualization of Clusters

With the clusters derived from the 2-slices and fast modularity algorithms, we graphically rendered the clusters using the Guess software tool. For this, we used a combined bipartite network of member and nonmember participations. An example of a 2-slice cluster of several projects with high homophily is in Figure 7:

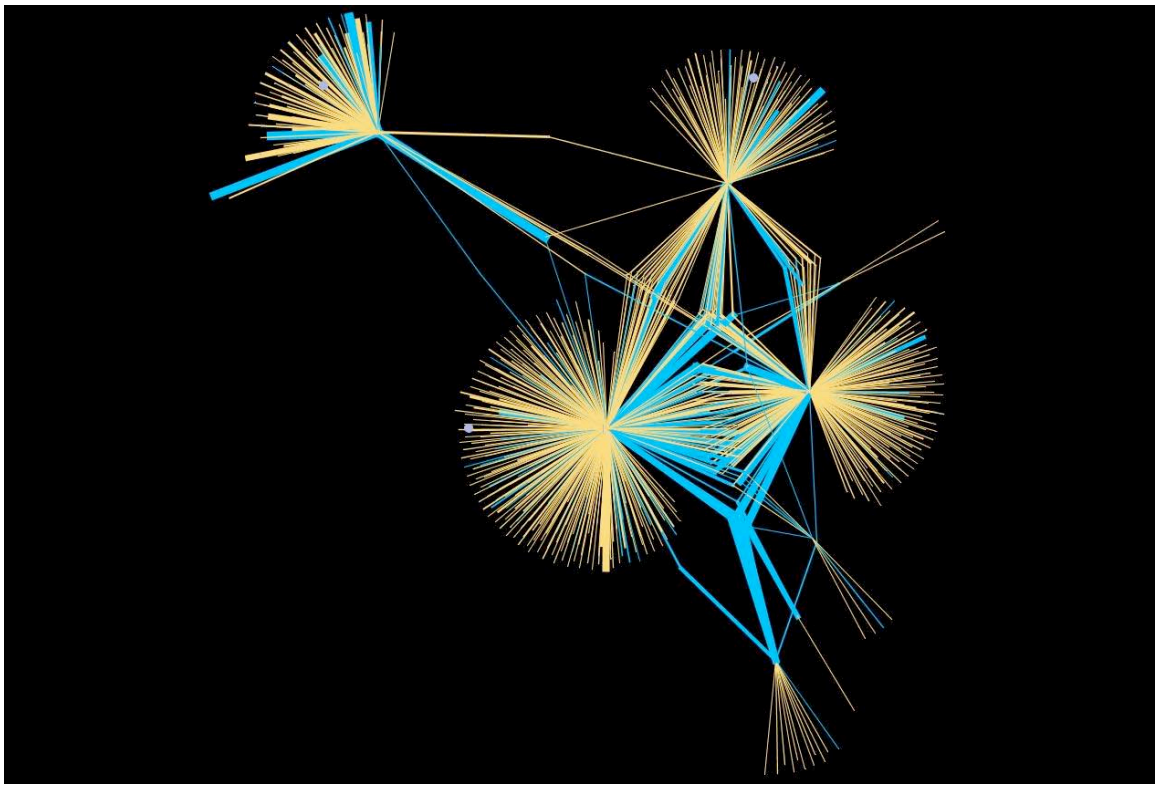

Figure 7: Visualization of 8 Zope/Plone-related projects in a member 2-slice cluster

In this visualization we used the GEM layout algorithm, with blue edges representing member contributions, and yellow edges representing non-member contributions. The hub nodes are groups, and edge nodes are users who only participated in one project in this cluster. We also made the edge thickness proportional to the edge weight in our original graph: that is the more participations a user has, the thicker their line is.

This visualization technique offers some clues to the nature of member participation in this cluster. For instance, it's obvious that several very thick blue lines exist and only a

few thicker yellow lines are visible. There are also many more non-member edges than member edges. Two very large tightly coupled projects seem to share most of their heavily contributing members, and several, but not most non-member participants. Figures 8 and 9 show another 2-slice cluster with similar properties, and a sparse cluster with high homophily from the fast modularity breakdown.

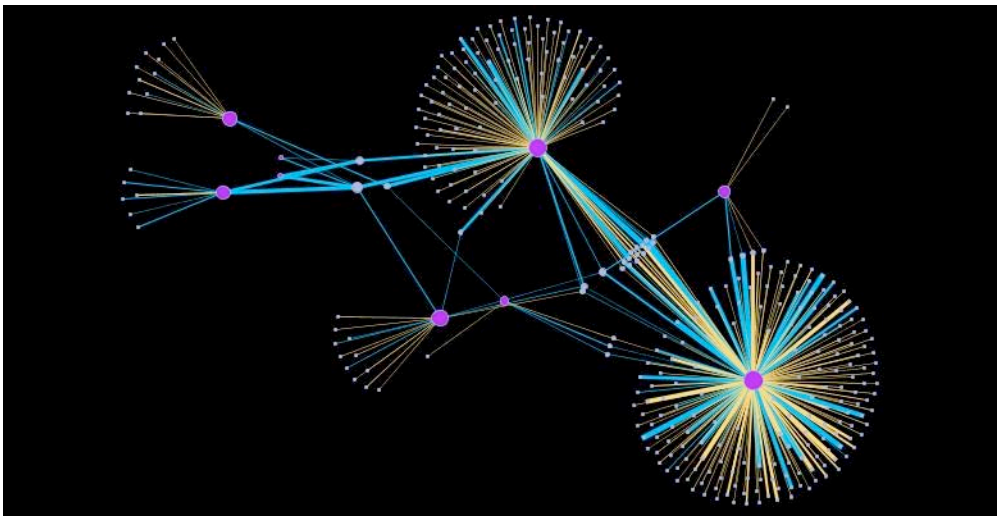

Figure 8: a 9 project 2-slice cluster of projects related to Gene Ontology.

In Figure 8, two large projects dominate, with several peripheral projects that share common member participants. While this cluster has fairly high homophily according to our measures, there is little non-member participation overlap inside the cluster aside from these two main projects.

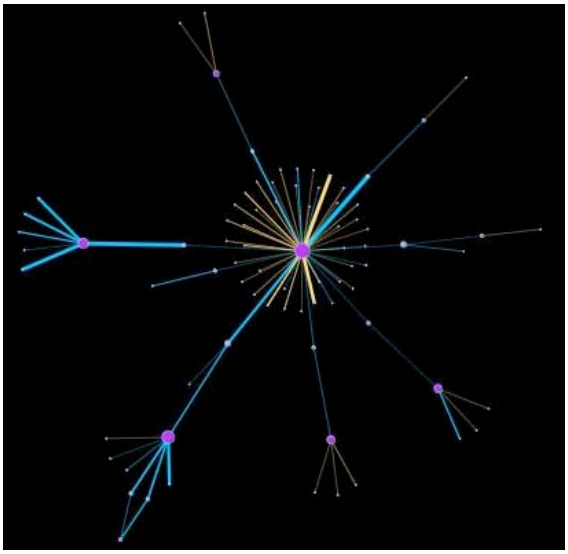

Figure 9: an 8 project cluster with of projects related to the Postnuke CMS.

In Figure 9, all projects are only related to the central project (postnuke) by one member and do not share any non-member participations. This type of loosely-connected cluster is typical of the smaller clusters detected by the fast modularity algorithm.

A closer look at the visualizations can reveal some interesting details and unexpected circumstances. For instance, we know on average members participate more in their projects than non-member participants. Figure 10 includes two users who seem to violate this rule.

Figure 10: Two participants with thick non-member (yellow) and thin member (blue) edges are an anomaly.

In Figure 10, two of the users between two projects with substantial membership overlap have contributed more to the project they are not members in. This seems to contradict what we would expect, given that all of the other common contributors to these projects are members of both projects and since members tend to contribute more than non-members overall. Detection of this kind of anomalous situation is fairly easy given this visual representation.

Time Series Representation

While this bipartite representation gives a reasonable snapshot of the all-time historical participation for project members and non-members and the strength of bonds between projects, it doesn't give any indication of the time dimension in the data set. In order to gain a better understanding of inter-project relationships and participation over time, a perl script was written which could generate an animation of network growth in Guess.

The basic structure of the script takes a Pajek network file as input, and extracts all records from our data set relating to those projects. The records are ordered by the

timestamp field, and binned by a specified time interval. It then generates a GDF file containing all nodes and several large python scripts that make the necessary transformations to node visibility, link creation, and link width for every frame. An example of the output is available at: https://sshgate.sf.net/~oostendo/zope_projects.mov

This movie, rendered in Guess, gives the participation patterns of 665 users for 8 SourceForge projects in a 2-slice of the member network, which in this case are related to the Zope/Plone content management system. This animation spans 52 months of development, with each "state" corresponding to a month of activity. A number of features of this clusters activity are apparent in this animation. The projects in the cluster are started independently at different points in time and don't initially share project members. Much of the participation at first seems to be concentrated among members and later among non-members. Very few members join in the later months in these projects, while non-members continue to approach the projects. There also seems to be a great deal of variation in how many new participants join each month: in both the early and late phases it may be single-digit growth, while in the middle it seems to gain one or two dozen nodes in each stage.

Future Work

While the breakdown of the SourceForge.net member network into clusters of similar projects is useful for finding communities, it would be potentially helpful to see if the larger network structure had meaning by looking at connections between clusters. It also might be useful to analyze clusters with substantial non-member participation overlap, since the member network is relatively sparse. While the 2-slices and fast modularity algorithms were able to tease out some clusters, it may eventually be useful to develop a less generalized community-finding algorithm particular to this data set.

The visualization and animation techniques for these clusters could be improved to aid in recognition of strengthening ties, and potentially reduced so larger networks could be visualized effectively. In particular, member and non-member user edge nodes could

probably be combined which would greatly reduce the number of on-screen nodes and allow for visualization of bigger clusters. It may also be interesting to expire data, so that the time animation tracks a window of activity rather than building to a cluttered all-time view.

Acknowledgements

Jude Yew gave invaluable creative direction to the project, including assessment of objectives, proposing research questions, and editorial input on drafts of this paper. He also gave guidance for techniques for visualization and analysis.

Rob Thompson came up with the proposal for 2-slicing the member network, and his prior work as a research intern for OSTG created a precursor to the participation data set.

References

1. Scott Christley and Greg Madey. "Analysis of activity in the open source software development community". In Proceedings of the 40th Annual Hawaii International Conference on System Sciences

2. J. Xu, S. Christley, and G. Madey, "The Open Source Software Community Structure", NAACSOS2005, Notre Dame, IN, June 2005.

3. Yongqin Gao and Greg Madey, "Towards understanding: A study of the SourceForge.net community using modeling and simulation", Agent-Directed Simulation (ADS'07) - Part of the 2007 Spring Simulation Multiconference (SpringSim'07), The Society for Modeling and Simulation International

4. Paul A. David and Francesco Rullani, “Micro-dynamics of Free and Open Source Software Development. Lurking, laboring and launching new projects on SourceForge”, (Working Paper) http://opensource.mit.edu/papers/davidrullani.pdf

5. M. E. J. Newman, “Modularity and community structure in networks” Proceedings of the National Academy of Sciences vol. 103 no. 23