Validity of network analyses in Open Source Projects

Conference Paper · May 2010

DOI: 10.1109/MSR.2010.5463342 · Source: DBLP

4 authors:

Roozbeh Nia
University of California, Davis
4 PUBLICATIONS **59** CITATIONS
SEE PROFILE

Christian Bird
Microsoft
80 PUBLICATIONS **3,491** CITATIONS
SEE PROFILE

Premkumar Devanbu
University of California, Davis
222 PUBLICATIONS **9,282** CITATIONS
SEE PROFILE

Vladimir Filkov
University of California, Davis
100 PUBLICATIONS **3,429** CITATIONS
SEE PROFILE

Validity of Network Analyses in Open Source Projects

Roozbeh Nia, Christian Bird, Premkumar Devanbu, Vladimir Filkov
Computer Science Department
University of California, Davis
{*nia, bird, devanbu, filkov*}*@cs.ucdavis.edu*

***Abstract*—Social network methods are frequently used to analyze networks derived from Open Source Project communication and collaboration data. Such studies typically discover patterns in the information flow between contributors or contributions in these projects. Social network metrics have also been used to predict defect occurrence. However, such studies often ignore or side-step the issue of whether (and in what way) the metrics and networks of study are influenced by inadequate or missing data.**

In previous studies email archives of OSS projects have provided a useful trace of the communication and co-ordination activities of the participants. These traces have been used to construct social networks that are then subject to various types of analysis. However, during the construction of these networks, some assumptions are made, that may not always hold; this leads to incomplete, and sometimes incorrect networks. The question then becomes, do these errors affect the validity of the ensuing analysis? In this paper we specifically examine the stability of network metrics in the presence of inadequate and missing data. The issues that we study are: 1) the effect of paths with broken information flow (i.e. consecutive edges which are out of temporal order) on measures of centrality of nodes in the network, and 2) the effect of missing links on such measures. We demonstrate on three different OSS projects that while these issues do change network topology, the metrics used in the analysis are stable with respect to such changes.

***Keywords*-Open Source, Social Networks, Information Flow**

I. Introduction

Some Open Source Software (OSS) projects have been runaway successes, sometimes besting commercial competitors. Their success, and the open availability of OSS project histories, including communication, development and maintenance activities, have made them valuable guinea pigs, (similar to *Caenorhabditis Elegans* and *Arabidopsis Thaliana*, in Biology) in the search for more effective ways of organizing distributed teams that collaborate using Internet modalities. Their email archives are a particularly interesting source of information concerning task-oriented communication behaviors of the OSS project collaborators. Social Network Analysis (SNA) on these networks, has proven valuable for generating a "bird's eye view" of networks of collaborating individuals, making it a natural setting for studying information flow and emerging organization in OSS projects by examining the social and communication networks of developers. Since SNA offers a quantitative, systemic type of analysis, it is appealing on multiple levels, especially to the empirical software engineering discipline which is growing in rigor and maturity.

There has recently been some criticism of this approach, focusing on data quality issues, the proper application of social network metrics, their adequacy for studying the problems of concern, and the interpretation of the results [1].

Of specific interest to this work are the SNA of developer communication networks constructed from correspondence mined from online developer mailing lists. These mailing lists are used for communication and coordination amongst the project workers, (e.g. to review possible changes to the source code [2]). One can derive social networks from the on-line mailing list archives. The nodes are the people sending messages on the list. If a person A replies to a message from another person B, then there is an edge connecting the node representing A to that representing B. Software engineering is a very knowledge-centric activity, and the mailing lists are the critical media for information exchange between developers. Information flow in the social network is naturally a critical area of study. The majority of what we know about the information flow in developers social networks is based on these mailing list interactions[1]

The email social networks have been analyzed in the past to determine the mediators of knowledge (hubs in those networks) as well as the emergence of community structure [4], [5].

Here we focus on two concerns about the validity of such studies due to the effects of inadequate data and metrics on information flow in email networks.

A. Incorrect Information Flow due to Temporal Aggregation

If developer A posts a message on the discussion groups (*e.g.* announcing a new feature that has been added), and developer B replies to developer A's message (warning that a duplication of function may have occurred), and C replies to B's message (concurring with B), one can reasonably

[1]There has been research on developer communication via IRC [3]; however, this research was interested in the length and attendance of meetings on IRC rather than actual *information flow* in a network.

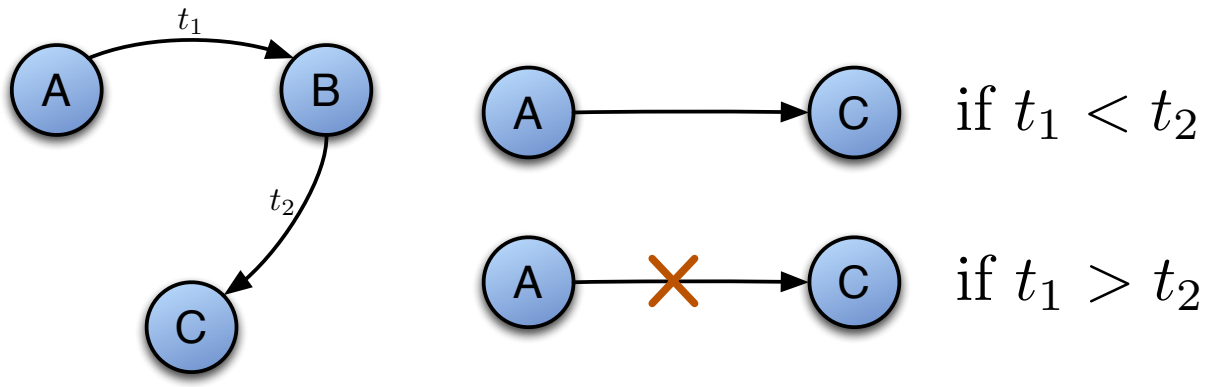


Figure 1: The same topology, left, may apply to two different cases based on the order in which the messages were posted. If $t_1 < t_2$, then information can flow from A to C. But if $t_1 > t_2$ no information can flow from A to C

conclude that there is information flowing from A to C. Alternatively, consider the situation in which B posts on some topic and C replies to B. Later A posts on an unrelated topic, and B replies to A. Unfortunately, many current network mining techniques consider all messages transmitted during an epoch, and construct a network of links observed during this epoch. After the network is constructed, the specific timing of each message is typically not recorded, and just the edges are retained. But, if one ignores the temporal ordering of these emails and focuses purely on the topology of the network, one might infer a transitive information flow relationship between A and C. In this scenario the order of events is the problem, *i.e.* it matters which of these events took place first. If we were limited to topological information with no time stamps on the edges, we might infer information flow through paths in the graph that in reality did not exist. We call such an erroneously inferred flow a *transitive fault*. Figures 1 illustrates this problem.

Many metrics used in SNA such as degree centrality, betweenness centrality, assortativity, etc. [6], disregard all but the topological information in the networks. If used without careful consideration, these metrics could yield misleading results. At the core, this is a temporal aggregation issue, i.e. transitive faults would not occur if the time interval under consideration is small enough. The effects of temporal aggregation on SNA results and the perils of ignoring them have been adequately addressed elsewhere [7]. However, certain amount of aggregation can never be eliminated, both because of the discreetness of the data but also because aggregated analysis is often the very intent of the studies. So then, we ask:

RQ1: *How much temporal data aggregation can be tolerated before SNA results become unreliable?*

B. Information Flow in the presence of Inadequate or Missing Data

Typically, social networks are derived from mailing list archives, using the "reply-to" field in messages. This practice is based on the observation that the "reply-to" field in a message is an indication of information flow. Thus, we say that there is information flowing from A to B if B replies to a message that A has posted on a thread: presumably B replies after s/he has digested the information content of A's message. One problem that arises from this formulation is that we can only observe information flow if a participant actually posts a message. If B read's a message posted by A, but does not reply, then there is information flowing from A to B, but there is no way for us to know that. Put simply, the existence of a reply from A to B indicates information flow from B to A, but the *lack of a reply does not imply that no information flow occurred.* This situation is due to the inadequacy or *missing data* in the email "reply-to" network. Depending on the topology of the observable network, the value of computed SNA metrics could potentially be affected seriously by the unobserved edges. When one uses SNA metrics node centrality to determine important people, or the clustering coefficient to determine local community structure on such networks, the results may not reflect reality. We therefore ask:

RQ2: *To what extent does missing data influence SNA metrics?*

There are surely other critiques of the extraction of social networks, and the use of SNA metrics; for now, we focus on the above two issues.

C. Our Contribution

In this work we seek to quantitatively ascertain the effect of the above two presented concerns on typical SNA anal-

yses on email networks of three OSS projects: Apache [8], Perl [9], and MySQL [10]. We begin with **RQ1**, *viz.,* the frequency with which we might not have information flow between developers due to transitive fault in the email network topology. Specifically, we address two questions. First, we ask: How frequent are transitive faults? If transitive faults are relatively rare (say a fraction of a percent of the time) then we could probably just ignore them. The second question arises if these faults are not rare: to what extent do transitive faults effect SNA analysis? We find that while transitive faults can be as frequent as 50%, their frequency is highly dependent on the time interval of aggregation, and that even when very frequent, they do not change results from SNA analysis critically.

We approach **RQ2** by specifically focusing on the question: How much do missing links affect downstream SNA analysis, when missing links are modeled under three different attachment scenarios? We find that the calculated betweenness centrality and clustering coefficient are stable in the presence of a large number of missing links.

Our contributions are thus:

1) We find that all social metrics that we have examined are robust to the *transitive faults* that occur when social network data is aggregated at intervals of one hour to one year.
2) We observe that all social metrics studied are robust to *missing links* as modeled by standard social network growth models.
3) We present a set of techniques which can by used by other researchers that utilize social network analysis metrics so that they can either show that their own metrics are robust or select different metrics which are.

II. Related Work

The empirical software engineering community in recent years has been deeply concerned with human aspects of software engineering. Many papers in main line software engineering conferences and venues, such as ICSE ([11], [12], [13]), FSE ([14], [5], [15]), ASE ([16], [17]), and MSR ([18], [19], [20]) have addressed this issue. In fact, a number of workshops such as Cooperative and Human Aspects of Software Engineering (CHASE) and Socio-Technical Congruence (STC) have arisen that focus specifically on this issue and have had numerous papers discussing the implications of human communication and co-ordination activities on software quality and developer productivity. In this line of work, mining of social networks from email archives has attracted quite a bit of interest. Social network metrics have long been used in sociology to analyze on-line communities [21] and are still [22]. A recent special issue of *Information Systems Research* [23] was devoted to analysis

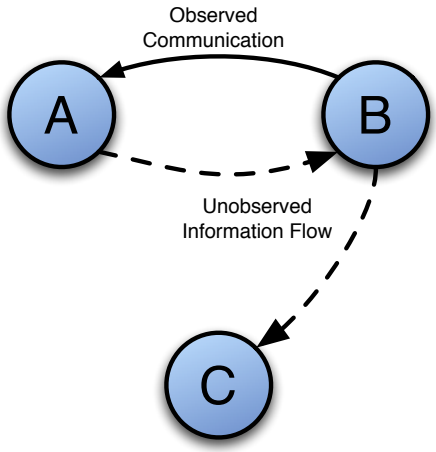


Figure 2: Observed communication (solid edges) is evidence of information flow from B to A. However, C may read B's message and B may have read A's response, which indicates unobserved information flow (dashed edges).

of on-line communities using such methods. In our own earlier work, we have shown that centrality of an individual as measured by social network metrics [4] in the email network is a powerful indicator of the technical activity and importance of an individual. Pohl and Diehl [24] more recently showed how networks could be used to determine roles of developers. We also found that (rather than being disorganized "bazaars") the email social networks of OSS projects show a very strong social structure [5] and are organized in ways that reflect the underlying collaborations between individuals [25]. Measures of importance such as betweenness [6] are used in many of these studies.

However, there are quite a few important assumptions, many previously unstated in published work, that are made in the construction of social networks from on-line email archives. Recently, Howison, Wiggins & Crowston [1] have presented a careful, thoughtful analysis of these assumptions, and argued that they should be considered as threats to validity of the results that use social network analysis over email archives. In addition to the temporal aggregation and information problems discussed earlier, they discuss other difficulties, such as the possibility of unrecorded back-channels of communication and modeling the intensity of communications. Temporal issues have also been addressed by Habiba [26], wherein the authors propose a new notion of betweenness that takes times of interactions into account. Braha and Bar-Yam [25] find that in dynamically changing networks, the "hub" roles of individuals actually change quite a bit from day to day, suggesting that efforts to disrupt social networks (*e.g.,* of criminals) by targeting key individuals, would require constantly shifting their sights.

Project	# of People	# of Messages	# of Edges	Timespan
Apache	1573	101250	11227	1995 – 2005
Perl	2411	112514	16026	1999 – 2007
MySQL	804	33678	1989	2000 – 2006

Table I: Original data details.

III. The Email Archives Data and Networks

A mailing list in an OSS project is a public forum. Anyone can post messages to the list. Posted messages are visible to all the mailing list subscribers. Posters to developer mailing lists include developers, bug-reporters, contributors (who submit patches, but don't have commit privileges) and ordinary users. Mailing lists can be quite active; for example, on the Apache developer mailing list, there were about 4996 messages in the year 2004 and 2340 in 2005. For Perl, these numbers were 10019 and 9606. For the time period studied, 2549 distinct individuals participated on the Apache developer list.

We have mined archival records of developer mailing lists to generate reply-to social networks for the three OSS projects: Apache, MySQL, and Perl. Because we are only interested in information flow, we only include messages that have received replies, since we have no evidence that a message without a reply was actually read by anyone (it clearly did not contain information worthy of a reply). One of the problems with email data is that one person may use multiple email addresses to send messages on a mailing list. We therefore use a semi-automatic alias resolution algorithm so that all messages are ascribed to the correct people. We refer the reader to previous work for details of the process [4]. Table I contains summary statistics for all of the projects.

For each of these projects, we construct an information flow network based on messages that are sent as replies to previous messages. When someone replies to a message on a mailing list, the id of the message being replied to is contained in the header of the message that is the reply. If message B is sent as a reply to message A, then there is information flow from the person that posted message A to the person that posted message B. There may also be information flow from the poster of B to the poster of A, but we have no way of knowing that the poster of A actually read the reply (unless of course the poster of A sends a second message in reply to B, in which case the methodology described here will create an edge of information flow). We use this methodology on all mined data to create a network of mailing list participants.

IV. Methodology

A. Networks and Transitive faults

We generated reply-to networks from the discussion groups for each project as described in the data section. We construct these networks by aggregating all the messages over a given time interval, and constructing the social networks for this interval. In the following studies we experimented with different aggregation time intervals, δt, from 1 hour up to the total lifespan of the project. Once a start time and an interval δt were chosen, we divide the messages into partitions, based on which interval they fall into (*e.g.* messages sent in first month of activity, messages in second month, etc.). Finally, a network was generated for each time interval, comprising all reply-to relationships in the time interval $(t_i, t_i + \delta t)$. Each edge is directed and labeled with the time the message was sent. Using this timing information, we gauge the extent to which a given network actually may give rise to spurious information flow.

We now define a *transitive fault* as a directed path of length exactly two where the time label on the first edge is later than the time label on the second edge along the path, i.e. a directed 2-path with decreasing edge time stamps. The *node transitive fault rate* is the fraction of transitive faults over all 2-paths through that node. The *network transitive fault rate* is the sum of the node transitive fault rates over all nodes, divided by the number of nodes in the network. Clearly, these fault rates depend on topology, and we intend to investigate the fault rates in OSS email social networks.

B. Network Measures

In this paper we use the following SNA measures.

- *Number of 2-paths (2P)* — The number of 2-paths through a node is a measure of local social status as defined previously [27].
- *Betweenness Centrality (BW)* — The betweenness centrality of a node is a function of the how many communication paths a node lies on and is often used a measure of global social status [28].
- *Clustering Coefficient (CC)* – The clustering coefficient measures the local connectivity density, or local structure in the graphs [29].

We now formally define these measures. Let a graph g be defined as a set of vertices V and edges $E : V \times V$. Let g_{ij} be the number of shortest paths from $i \in V$ to $j \in V$[2], and let g_{ivj} be the number of shortest paths from i to j that pass through node v. For some node $v \in V$ in graph g, the betweenness centrality, BW is defined as follows:

[2]Note that there may be more than one shortest path between two nodes if multiple paths are of the same length.

$$BW(v) = \sum_{i,j,i\neq j,i\neq v,j\neq v} \frac{g_{ivj}}{g_{ij}} \quad (1)$$

This is a measure of the global importance, or centrality, of a node. The closer the betweenness of a person is to 1, the harder it is for others to communicate efficiently *without* information flowing through this person.

For a node v in a graph g, the clustering coefficient quantifies the density of the neighborhood of v with values ranging from 0, which indicates a star topology around v, to 1, indicating that the network formed by v and all immediate neighbors forms a clique.

$$C_v = \frac{|\{e_{jk}\}|}{k_v(k_v-1)} : v_j, v_k \in N_v, e_{jk} \in E. \quad (2)$$

where N_v is the neighborhood of v, defined as:

$$N_i = \{u : e_{vu} \in E \wedge e_{uv} \in E\} \quad (3)$$

e_{vu} is an edge, connecting v to u and k_v is the degree of a node defined as the number of vertices, $|N_v|$, in its neighborhood N_i. The clustering coefficient of a person in a social network is a measure of the connectedness of that person's neighbors among each other, and thus is indicative of the local clique strength that a person is in. (In undirected graphs, for any node this is just the number of triangles in which that node is a vertex, divided by the number of all possible pairs made from that vertex's neighbors).

The total number of 2-paths has been used to characterize networks in SNA, particularly when modeling node importance, or centrality, in random graph models [27], [30]. Indeed, the ranking of nodes based on the number of 2-paths going through a node correlates very strongly (using the non-parametric Spearman rank correlation [31] due to the power-law distribution of such metrics []) with the node ranking based on node degree (unpublished), and also with betweenness centrality [32]. In our studies, this is a natural measure of centrality since it has a direct connection to the number of transitive faults associated with a node.

C. Studies

We quantitatively evaluate **RQ1** with two studies. First, we gradually increase the aggregation epochs, stepping from hours through days, months, years through to project life times. We measure if, and by how much, transitive faults in the networks change as the epoch size increases. We also take a look more precisely at how the faults change at finer temporal resolutions for each of the three projects. Then, we assess what effect varying amounts of transitive faults have on the ranking of nodes according to the number of 2-paths, for each project. Because there could be multiple edges between the same people in the network, the same topology could be annotated with multiple time stamps on the edges, as the epoch size changes. Thus, some two-paths could be transitive faults for some epochs and not in others.

Because we often observe multiple messages being sent in both directions between the same two people, it is difficult to tell if a particular pair of edges that form a 2-path represent a transitive fault. For instance, if we see an edge, $B \rightarrow C$, prior to an edge $A \rightarrow B$, we may decide that the 2-path $A \rightarrow B \rightarrow C$ is a transitive fault because the order of edges is incorrect temporally. However, if within the same time interval we observe a later occurrence of $B \rightarrow C$, then there are two possibilities. If the second $B \rightarrow C$ corresponds to a message which contains information that was originally sent in the message that created the edge $A \rightarrow B$, then the 2-path *does* represent a valid flow of information temporally. Since we don't know if actual information is being exchanged in each pair of edges that represents a 2-path, we model an optimistic and pessimistic model. Our optimistic model represents a lower bound on the transitive fault rate. In this case, whenever we see $B \rightarrow C$ following $A \rightarrow B$, we indicate *no* transitive faults for the 2-path $A \rightarrow B \rightarrow C$, regardless of if there is an edge $B \rightarrow C$ prior to the edge $A \rightarrow B$ (which in isolation would represent a transitive fault). Our pessimistic model represents an upper bound on the fault rate. Here, whenever we see an edge $A \rightarrow B$ *after* an edge $B \rightarrow C$, we label the 2-path $A \rightarrow B \rightarrow C$ as a transitive fault regardless of what other edges between A, B, and C exist in the same time interval. The true transitive fault rate lies somewhere in the middle of these two values. We calculate the upper and lower bounds for all three projects using intervals ranging from 1 hour to 10,000 hours (just over one year).

Turning to **RQ2**, we note that the concern here is the potential impact of the missing edges on the stability of the network measures. In other words, are the measures affected by missing edges? We evaluate this by *adding* edges to the measures, and gaugeing the effect on the measures. We use three different models for adding edges to the observed graph. These models are based on prior, validated theories of real-world social network dynamics, and represent how dynamic social networks grow over time. When we artificially insert additional edges into the networks, as predicted by these models, we hope to realistically simulate the "missing links" of information flow that occur in developer networks; the links are "missing" in the sense that they are not observable in message traffic. In the first model of "missing links", which we call the Time Window (TW) model, we assume that people read all postings within a time interval (*e.g.* the last 30 days) before posting a reply. This model assumes that all message posters are reading all the messages in the given interval before their

Time Interval	Apache	MySQL	Perl
Hourly	0.48 – 0.55	0.38 – 0.43	0.45 – 0.52
Daily	0.43 – 0.55	0.41 – 0.53	0.44 – 0.55
Monthly	0.21 – 0.50	0.38 – 0.51	0.27 – 0.51
Yearly	0.11 – 0.49	0.37 – 0.50	0.17 – 0.50
Lifespan	0.15 – 0.50	0.41 – 0.51	0.17 – 0.51

Table II: Upper and lower bounds on network transitive fault rates for varying time intervals

post, and thus information in these message has flowed into them. In the second and third models, we assume that links are missing randomly based on two different well-known random graph models, the Erdős-Rényi (ER) [33] and Preferential Attachment (PA) [34] models, respectively. The results from the 3 models are simulated using a Monte-Carlo approach, and compared. We used a biased coin approach so that total number of edges added in the latter 2 models were close to the number of links added in the TW model. Once we augment the networks with these links, we calculate the betweenness centrality and clustering coefficient for each node and correlate the metrics for the nodes in the new graph with the corresponding metrics in the original graph.

If the measures in the simulated graphs produce substantially different centrality results, there is a strong indication the measures are not stable in the presence of missing links, and therefore should be viewed with some suspicion.

V. Results and Discussion

A. Fault Rates

The range of fault rates for different aggregation time intervals (epochs) are given in Table II, in terms of a lower-bound upper-bound range. While the worst case scenario is always around 50%, it is apparent that as the aggregation intervals get longer the best case scenario shows fewer faults.

In Fig. 3 we show the more precise relationship between the fault rate change (both upper- and lower-bounds) and the aggregation time, for Apache, MySQL, and Perl (respectively). The difference between the lower bound curves for Apache and Perl on one hand and MySQL on the other probably tell us that the "back-and-forth" debates on issues are significantly shorter on the MySQL discussion boards vs the other two projects. It is also notable that the reply-to MySQL email networks, at the same temporal granularity, were much sparser than those of Apache and Perl, Cf. I.

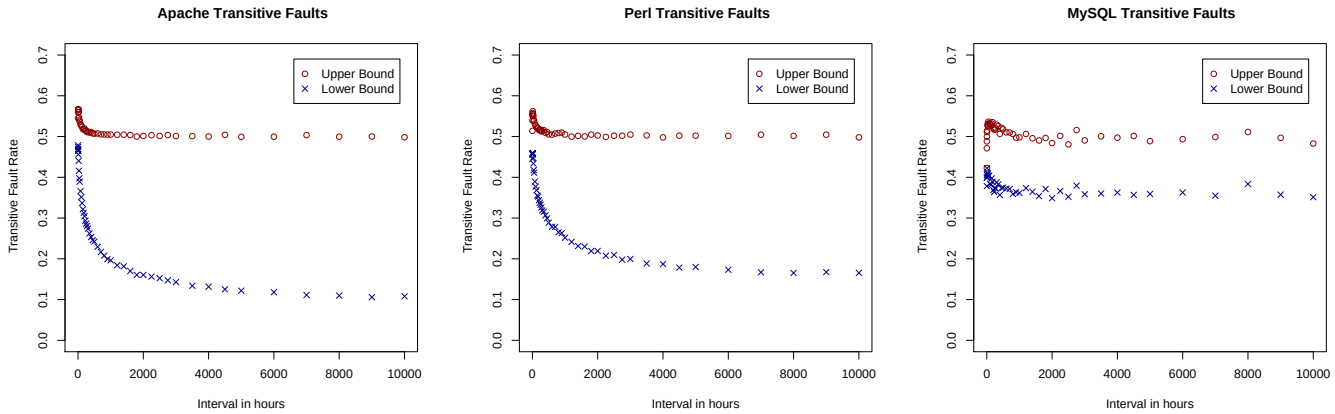
Finally, and most tellingly, Table III shows the rank correlations (using Spearman rank correlation) between ordering of nodes that includes the transitive faults vs ordering of nodes excluding the transitive faults going through the respective nodes. The ordering of nodes is based on the centrality measure of number of 2-paths through the node. The higher the value the better the relative ordering of nodes is preserved. The overall high values for the correlation indicate that the orderings are largely preserved, and except for the very lowest values (for MySQL, where the sparseness of the communication networks are the likely cause, e.g. there were very few 2-paths in any 1 hour interval) the probabilities of observing them by chance (i.e. the p-values) are very small. Still, we note that the time interval is a factor here, especially for smaller projects with low communication activity levels. We conclude that if the time interval is chosen to be not too small, the results will be largely unaffected by even large rates of transitive faults.

B. Missing Links

Table IV show the number of missing links added to the graphs using each of the three models. In most cases a very significant number of edges (several times more than the number of original edges) get added to the original data, so one might expect that the measures would yield substantially different results. For each of the projects we chose for comparison the top 10% of people who have the highest value for the betweenness or clustering coefficient in the original network, and the networks that we generated by simulations, and compare their rank correlations (again, using Spearman rank correlation). The highly-skewed Pareto nature of the distribution of activity in open-source projects [35], [4] is well known. This essentially means that most of the activity in these social networks arise from a few participants. Thus it's sufficient to look at at the 10% people. More than half of the people have betweenness centrality or clustering coefficient of 0. Tables V and VI show that the ranking of the betweenness and clustering coefficient of the original data set compared with the three new data sets that we created are highly correlated. The significance of these correlations is greater than or equal to 99% for all of the comparison. For each of the randomly generated graphs based on the original graph (i.e. ER and PA), we generated 100 different random graphs, and averaged the rank correlation over these 100 runs. Although we add a significant number of edges (several times more than the number of edges in the original graph) we see that the people who had the highest betweenness and clustering coefficient in the original data are still amongst the top people in the newly generated data. Moreover the overall

Time Interval	Apache (p-val)	MySQL (p-val)	Perl (p-val)
1 day	0.67 (0.01)	0.52 (0.22)	0.74 (0.01)
5 days	0.71 (0.01)	0.63 (0.01)	0.77 (0.0001)
1 year	0.82 (0.0001)	0.73 (0.0001)	0.86 (0.0001)

Table III: Stability of rankings based on number of 2-paths. The values are rank correlations (Spearman) of 2-path rankings with vs without transitive faults. In parenthesis are the p-values (1-significance) of the tests.



(a) (b) (c)

Figure 3: The upper and lower bounds on transitive fault rates for Apache (a), Perl (b), and MySQL (c) as the time interval increases from 1 hour to 10,000 hours (just over one year).

ranking stays stable. Thus, missing links behaving according to our three models do not have a considerable effect on the structure of our social networks in OSS projects. This shows that the methodology is reliable and robust.

VI. Threats to Validity

While not a real threat to the validity of this work, we feel obliged to accent strongly the following. What we show in this work is that one can expect the behavior of some important SNA measures to not be affected significantly in the presence of 1) missing data, due to inherent limitations of the data sources, and 2) incorrect information flow inferences, due to temporal network aggregation. What we do not demonstrate in this work is the utility of any such measures for any particular goal; that has been the topic of many prior works. In other words, these techniques may be inadequate for some particular task so the results will not mean anything, but they will still be stable in the presence of the above data challenges. The stability is not to be confused with utility.

SNA analysis comprises many techniques and measures. Here, we recognize that showing a few measures (i.e. centrality and clusteredness) are stable in the presence of missing and inadequate data does not give a clean bill of health nor a license for future practice to the whole approach of SNA analysis. However, we point out that the information flow issues as well as the measures we used are fairly general, important enough, and widely assumed to be pretty safe. Thus, had they not passed the test that we subjected them to, many previous studies would have been challenged. Surely, studying the stability of many other local and global network measures is necessary before SNA analyses methods and techniques deserve the confidence people often place in them. Further, there are other properties of SNA metrics that are important to investigate. For example performance analysis is also critical.

On a technical note, while we found that some measures are fairly stable in the presence of challenging data, we did so by using measures that matched well and were easy to test in our specific application (e.g. using the number of 2-paths as a centrality measure). Ideally, one would need algorithms and evaluation techniques that can calculate existing measures with and without network paths that cannot carry information. Instead, most of the existing algorithmic work in this area aim to either develop new measures or faster ways to calculate old ones. Also, typically, existing algorithms assume a static graph and suffer from the effects

Project	Apache	MySQL	Perl
# of edges in ER	26555	4917	40350
# of edges in PA	28326	4752	40122
# of edges in TW	29032	5362	37910

Table IV: Number of missing links added back to the graphs for all three models; TW=Time Window, ER=Erdős-Rényi, PA=Preferential Attachment

Project	Apache	MySQL	Perl
ER	0.78	0.98	0.65
PA	0.80	0.63	0.76
TW	0.62	0.67	0.62

Table V: Rank correlation (Spearman) of the top 10% of nodes, with vs. without transitive faults. The nodes were ranked based on their Clustering Coefficient. TW=Time Window, ER=Erdős-Rényi, PA=Preferential Attachment.

Project	Apache	MySQL	Perl
ER	0.88	0.90	0.72
PA	0.72	0.77	0.80
TW	0.89	0.77	0.86

Table VI: Rank correlation (Spearman) of the top 10% of nodes, with vs. without transitive faults. The nodes were ranked based on their Betweenness Centrality. TW=Time Window, ER=Erdős-Rényi, PA=Preferential Attachment.

of transitive faults. This work and the work of Howison *et al.* [1] illustrate the problem, which clearly presents an avenue for future research.

VII. Conclusion

We have shown that a set of measures of social network analysis are robust to noise in network data. Specifically, we have shown that:

1) The clustering coefficient and the 2-path counts are both robust to data aggregation across large intervals (over one year) even though such aggregation may lead to transitive faults.
2) The clustering coefficient and betweenness social network analysis metrics on a network with missing links are highly correlated with network that contain augmented networks with links added, indicating that they are also robust to some information loss.

These findings are good news in that they lend support to prior research in light of the concerns raised by Howison *et al.* [1]. In addition, further research that rely on social networks that suffer from transitive faults or missing links may continue to use these measures with confidence.

Our study has examined a limited set of SNA metrics. However, software engineering research from noted researches such as Pinzger, Zimmermann, and Nagappan [15], [36], Williams and Meneeley [37], [38], Wolf *et al.* [39], [40] and others have used these and other measures. In our study we have presented techniques that other researchers can use to test for robustness of additional metrics to missing links and transitive faults. When the metrics used are found to be robust, this increases confidence in the findings of a study. When metrics are found to be susceptible to transitive faults or missing links, other metrics may be chosen which are more robust.

Acknowledgements We acknowledge support from an IBM Faculty Fellowship. We also acknowledge with gratitude support from the NSF Science of Design Program, grant No. SoD-TEAM 0613949. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] J. Howison, A. Wiggins, and K. Crowston, "Validity issues in the use of social network analysis for the study of online communities. (under revision at journal for the association of information systems (jais))." 2009, under Review.

[2] P. Rigby, D. German, and M.-A. Storey, "Open source software peer review practices: A case study of the apache server," in *Proc. of the International Conference on Software Engineering*, 2008.

[3] E. Shihab, Z. M. Jiang, and A. E. Hassan, "On the use of internet relay chat (irc) meetings by developers of the gnome gtk+ project," in *Proceedings of the 6th International Working Conference on Mining Software Repositories*, 2009, pp. 107–110.

[4] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan, "Mining email social networks," in *Proceedings of the 3rd International Workshop on Mining Software Repositories*, 2006.

[5] C. Bird, D. Pattison, R. D'Souza, V. Filkov, and P. Devanbu, "Latent social structure in open source projects," in *SIGSOFT '08/FSE-16: Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*. New York, NY, USA: ACM, 2008, pp. 24–35.

[6] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge University Press, 1994.

[7] D. Braha and Y. Bar-Yam, "From centrality to temporary fame: Dynamic centrality in complex networks," *Complexity*, vol. 12, pp. 59–63, 2006.

[8] "The Apache HTTP Server Project." [Online]. Available: http://httpd.apache.org

[9] "The Perl Programming Language." [Online]. Available: http://www.perl.org

[10] "The MySQL Relational Database." [Online]. Available: http://www.mysql.org

[11] N. Nagappan, B. Murphy, and V. Basili, "The influence of organizational structure on software quality: an empirical case study," in *ICSE '08: Proceedings of the 30th international conference on Software engineering*. New York, NY, USA: ACM, 2008, pp. 521–530.

[12] C. Treude and M.-A. D. Storey, "How tagging helps bridge the gap between social and technical aspects in software development," in *Proceedings of the 31st International Conference on Software Engineering*, 2009, pp. 12–22.

[13] A. Sarma, L. Maccherone, P. Wagstrom, and J. D. Herbsleb, "Tesseract: Interactive visual exploration of socio-technical relationships in software development," in *Proceedings of the 31st International Conference on Software Engineering*, 2009, pp. 23–33.

[14] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," in *Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2009, pp. 111–120.

[15] M. Pinzger, N. Nagappan, and B. Murphy, "Can developer-module networks predict failures?" in *SIGSOFT '08/FSE-16: Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*. New York, NY, USA: ACM, 2008, pp. 2–12.

[16] R. Hegde and P. Dewan, "Connecting programming environments to support ad-hoc collaboration," in *23rd IEEE/ACM International Conference on Automated Software Engineering*, 2008, pp. 178–187.

[17] A. Sarma, G. Bortis, and A. van der Hoek, "Towards supporting awareness of indirect conflicts across software configuration management workspaces," in *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering*. New York, NY, USA: ACM, 2007, pp. 94–103.

[18] W. Maalej and H.-J. Happel, "From work to word: How do software developers describe their work?" in *Proceedings of the 6th International Working Conference on Mining Software Repositories*, 2009, pp. 121–130.

[19] O. Alonso, P. T. Devanbu, and M. Gertz, "Expertise identification and visualization from cvs," in *Proceedings of the 2008 International Working Conference on Mining Software Repositories*, 2008, pp. 125–128.

[20] G. Valetto, S. Chulani, and C. Williams, "Balancing the value and risk of socio-technical congruence," *Workshop on Sociotechnical Congruence*, 2008.

[21] B. Wellman, J. Salaff, D. Dimitrova, L. Garton, M. Gulia, and C. Haythornthwaite, "Computer networks as social networks: Collaborative work, telework, and virtual community," *Annual review of sociology*, vol. 22, no. 1, pp. 213–238, 1996.

[22] A. Gruzd and C. Haythornthwaite, "Automated discovery and analysis of social networks from threaded discussions," *International Network of Social Network Analysts*, 2008.

[23] R. Agarwal, A. Gupta, and R. Kraut, "Editorial Overview–The Interplay Between Digital and Social Networks," *Information Systems Research*, vol. 19, no. 3, p. 243, 2008.

[24] M. Pohl and S. Diehl, "What dynamic network metrics can tell us about developer roles," in *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*. ACM New York, NY, USA, 2008, pp. 81–84.

[25] D. Braha and Y. Bar-Yam, "Topology of large-scale engineering problem-solving networks," *Physical Review E*, vol. 69, no. 1, p. 16113, 2004.

[26] H. Habiba, C. Tantipathananandh, and T. Berger-Wolf, "Betweenness Centrality Measure in Dynamic Networks," Technical report, DIMACS, 2007, Tech. Rep., 2007.

[27] T. Snijders, P. Pattison, G. Robins, and M. Handcock, "New specifications for exponential random graph models," *Sociological Methodology*, pp. 99–153, 2006.

[28] L. C. Freeman, "A set of measures of centrality based upon betweenness," *Sociometry*, vol. 40, pp. 35–41, 1977.

[29] D. Watts and S. Strogatz, "Collective dynamics of ?small-world?networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[30] M. Loebl, J. Matoušek, and O. Pangrác, "Triangles in random graphs," *Discrete Mathematics*, vol. 289, no. 1-3, pp. 181–185, 2004.

[31] S. Dowdy, S. Wearden, and D. Chilko, *Statistics for research*, 3rd ed. John Wiley & Sons, 2004.

[32] C.-Y. Lee, "Correlations among centrality measures in complex networks," *arXiv:physics/0605220v1*, 2006.

[33] P. Erdős and A. Rényi, "On random graphs," *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.

[34] A. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, p. 509, 1999.

[35] A. Mockus, J. D. Herbsleb, and R. T. Fielding, "Two case studies of open source software development: Apache and mozilla," *ACM Transactions on Software Engineering and Methodology*, vol. 11, no. 3, pp. 309–346, July 2002.

[36] T. Zimmermann and N. Nagappan, "Predicting defects using social network analysis on dependency graphs," in *Proc. of the International Conference on Software Engineering*, 2008.

[37] A. Meneely, L. Williams, W. Snipes, and J. Osborne, "Predicting failures with developer networks and social network analysis," in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*. ACM, 2008.

[38] A. Meneely and L. A. Williams, "Secure open source collaboration: an empirical study of linus' law," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2009, pp. 453–462.

[39] T. Wolf, A. Schröter, D. Damian, and T. H. D. Nguyen, "Predicting build failures using social network analysis on developer communication," in *31st International Conference on Software Engineering*, 2009, pp. 1–11.

[40] T. Wolf, A. Schröter, D. Damian, L. D. Panjer, and T. H. D. Nguyen, "Mining task-based social networks to explore collaboration in software teams," *IEEE Software*, vol. 26, no. 1, pp. 58–66, 2009.