Social dynamics of free and open source team communications

James Howison, Keisuke Inoue, and Kevin Crowston

School of Information Studies
Syracuse University
Syracuse, USA
{jhowison,kinoue,crowston}@syr.edu

Abstract[1] This paper furthers inquiry into the social structure of free and open source software (FLOSS) teams by undertaking social network analysis across time. Contrary to expectations, we confirmed earlier findings of a wide distribution of centralizations even when examining the networks over time. The paper also provides empirical evidence that while change at the center of FLOSS projects is relatively uncommon, participation across the project communities is highly skewed, with many participants appearing for only one period. Surprisingly, large project teams are not more likely to undergo change at their centers.
Keywords: Software Development, Human Factors, Dynamic social networks, FLOSS teams, bug fixing, communications, longitudinal social network analysis

1 Introduction and Literature Review

Free/Libre Open Source Software (FLOSS[2]) is a broad term used to embrace software developed and released under an "open source" license allowing inspection, modification and redistribution of the software's source without charge ("free as in beer"). Much though not all of this software is also "free software," meaning that derivative works must be made available under the same unrestrictive license terms ("free as in speech", thus "libre"). We study FLOSS teams because they are remarkable successful distributed work teams; we are interested in understanding how these teams organize for success.

In this paper, we investigate the informal social structure of FLOSS development teams by examining the pattern of communications between developers.

[1] Acknowledgement: This research was partially supported by NSF Grants 03–41475, 04–14468 and 05–27457. 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

[2] The free software movement and the open source movement are distinct and have different philosophies but mostly common practices. In recognition of these two communities, we use the acronym FLOSS, standing for Free/Libre and Open Source Software.

Please use the following format when citing this chapter:
Howison, J., Inoue, K., and Crowston, K., 2006, in IFIP International Federation for Information Processing, Volume 203, Open Source Systems, eds. Damiani, E., Fitzgerald, B., Scacchi, W., Scotto, M., Succi, G., (Boston: Springer), pp. 319-330

We are seeking social patterns reflected in artifacts of project activity, what de Souza *et al* call "an 'archeology' of software development processes" [5]. In this paper, we analyze communication network data over time, using snapshot data, to understand better how social structures in projects are changing over time. We first examine average centralization over time, then we examine change at the center and finally the stability of participation in project communications[3].

White et al [15] introduced the modeling of social structure over time using snapshot data. Our method is similar and their clear comment also applies, we "present no models of processes over time; there are neither predictions of other behavior nor explications of a stochastic process of tie formation and dissolution" (p 732). Rather the analysis below seeks merely to describe the structures as found at different points in time. Analysis of networks over time with attention to causes and predictions from structure and its change, such as preferential attachment, is an active area of research [11, 9] and one that may be fruitful on this data.

Analysis of networks over time is also new to analysis of software development communications. Recently de Souza et al [5] reported their examination of FLOSS project communications for a small number of projects at two points in time; they were able to see the movement of developers between the core and the periphery of the project. The work presented below extends such analysis to a large sample of data using automated analysis techniques.

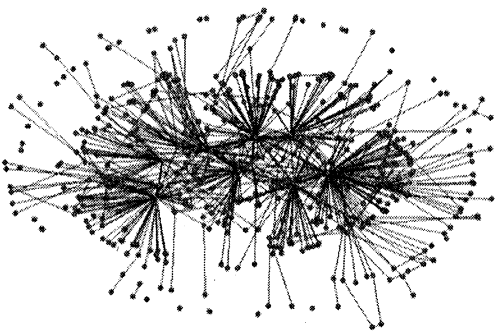

Fig. 1. `squirrelmail` from [4] Modular, or changes at the center over time?

Prior research has shown that FLOSS teams exhibit a wide range of centralizations, counter to both the common image of teams as totally decentralized and the academic expectation of centralization [3, 4]. This work has also shown that centralization scores are negatively correlated with number of participants in the bug report discussions, specifically, that small projects can be centralized

[3] A longer version of this paper, that presents full summary statistics and time series of network centralization over time, is available online at `http://floss.syr.edu/publications/`

or decentralized, but larger projects are decentralized. Figure 1 shows a large decentralized network.

Two explanations have been offered for this finding: first, the fact that in a large project, it is simply not possible for a single individual to be involved in fixing every bug. As projects grow, they have to become more modular, with different people responsible for different modules. In other words, a large project is in fact an aggregate of smaller projects, resulting in what might be described as a "shallot-shaped" structure, with layers around multiple centers.

An alternative explanation is that the larger projects are more likely to have experienced changes in leadership. This seems particularly credible when one considers that participant counts are positively affected by project lifespan. During any given period, the network may be centralized around a current leader, but overlapping the networks from all periods gives a total network with multiple centers and thus an artificially decentralized network.

Accordingly after comparing average centralization over time with the overall centralizations reported in [4], we then examine changes at the center of the communications networks. Stability at the center of a project is likely important to the team's performance. Linus Torvald's position in the `Linux` project is legendary and there is constant concern that he is being over-stretched [10]. This concern is based, in part, on the knowledge that transition is difficult; central personnel likely hold much tacit knowledge and stability in structure ought to assist coordination through transactive memory.

Finally we examine the frequency of participation in project communications. The ability to attract and retain project participants is an important measure of FLOSS project success, demonstrating the project's viability as well as its ability to satisfy its participants. Repeated involvement, or what we might call tenure, should also serve as a knowledge and skill transmission device. This is particularly important amongst the core team but is also important amongst the periphery of active users, who learn to provide "usable" bug reports as well as how to run the latest development snapshots. Long-term active users may step in as 'newbie wranglers' able to answer the frequency asked questions and thus shielding the core developers, freeing up their time and attention. We examine the frequency of participant's involvement across time and relate it to the patterns of difference in centralizations

2 Data and Method

For this analysis we utilized data collected from the SourceForge bug tracker. The bug fixing process provides a "microcosm of coordination problems" [2] and is a collaborative task in which, as Eric Raymond [12] paraphrases Linus Torvalds: the people finding bugs are different from those that understand the bugs and those that fix the bugs.

We selected projects from SourceForge and downloaded project and bug database data using Web spiders (see [8]). The projects selected were projects

that had had more than 100 bugs (open or closed) in the tracker at the time of selection in April 2002 and which had more than seven developers active overall in the discussions. This yielded data on 120 relatively successful projects.

We extracted interaction data from the project bug reports to create interaction matrices. These were analyzed using social network analysis (SNA) [14]. The bug reports contain a thread of discussion (shown elsewhere in Figure 4 of [4]). The initial bug-reporter posts via a web interface, typically triggering a message to a group of developers, or the development mailing list, depending how the project is organized. Replies, often seeking more information or confirmation, are then posted to the bug, being copied to all previous recipients and posted in the public forum.

SNA requires the construction of sociomatrices, depictions of social networks organized around dyads (pairs of senders and receivers). The appropriate dyad in the case of an open forum is an interesting question in its own right. While the origin of the message can be determined from the Sourceforge ID, the message may well be received by all project participants (if the tracker is copied to a mailing list), by all previous posters to the tracker, or merely by the previous poster in the thread. This question is of great importance to studies relying on the information flow characteristics of social networks.

For this reason, we simply coded the interaction as occurring between the sender and the immediately previous poster and calculated outdegree centralization. This was reasonable because our reading of the bug-reports showed that most messages are a reaction to the immediately prior message and because we are primarily interested in contribution, and not information flows per se. Our dyad can be understood as 'was prompted to speak in public by,' an interpretation which is robust with our interpretations below. These 'in-public' dyads mean that it is conceptually difficult to utilize network measures, such as betweenness centrality, which assume that only the recipient has read the message, and that the recipient chooses whether to forward that information onwards.

Outdegree centralization measures inequality in the proportion of the total population spoken to by each node. A network in which a single individual has spoken with all other participants, but where those others have only spoken with that single individual would have very high outdegree centralization (1.0). Conversely a network in which each participant has spoken with every other participant would have very low outdegree centralization (0.0).

Each message has a time-stamp given when the message is received by the tracker system. We used this data to divide the networks into over-lapping snapshots. We sampled the network in 90-day windows, moving the window forward 30 days at a time. This means that a single dyad may be reflected in up to three consecutive snapshots. We chose to use overlapping windows to smooth changes in the network structure and 90 days was chosen so that the majority of the projects contain enough communications to analyze in each time period. The data and analysis scripts for this paper are available through FLOSSmole [7].

3 Findings

3.1 Centralization

Our snapshot data provided an outdegree centralization figure for each project in each frame. Thus we have a time series for project centralization. We hope to explore such patterns in detail using time-series techniques to measure stability and trends across the data set, but at present we describe the series only through their means and variance. The left-hand figure in Figure 2 shows the distribution for the average outdegree centralization over time. Centralization is distributed, with a mean of 0.59, and Median of 0.58 and a standard deviation of 0.15. The right-hand figure in Figure 2 attempts to measure the stability of the centralization scores by examining the standard deviations of the series. Given that centralization is normalized between 0 and 1, it is reasonable to compare the standard deviations. The distribution shows that the majority of centralization scores vary $\pm$ 0.2 through their lifetime.

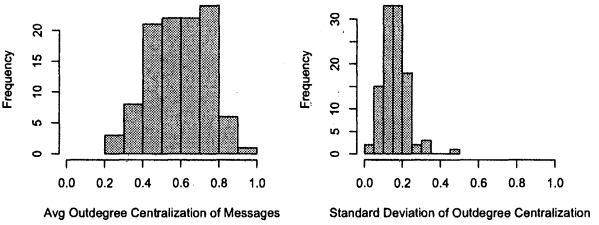

Fig. 2. Average Centralization over time is widely distributed, with moderate internal variance

If the hypothesis expressed in [4] was correct, and changes at the center had artificially reduced the centralization score by collapsing time, the distribution of average centralization ought to be higher overall than the distribution of overall centralization. This was not the case. There was no statistical difference between the distribution of average centralization presented in this paper and overall centralization presented in [4].

Figure 3 shows the differences between the average of our centralization scores computed from the snapshots, and the centralization score obtained by collapsing the network over time. The diagonal line shows equality, and the perpendicular distance from that line shows the difference, either positive (the collapsing of the network has produced an 'artificially' decentralized network) or, somewhat unexpectedly, negative (where the collapsing of the network has produced an 'artificially' centralized network). We can see that the projects with positive and high differences appear to include some of the projects, such

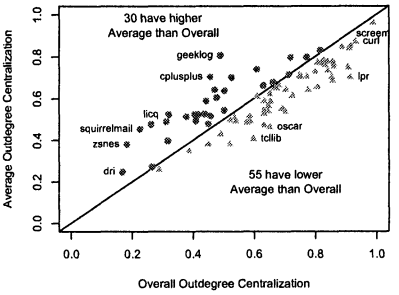

Fig. 3. The effects of collapsing networks over time

as `squirrelmail`, that we anticipated might have undergone change at the center, but the significant number of projects with low negative differences renders the two distributions statistically similar.

To clarify, we considered two ideal cases of networks over time that would produce such differences in overall and average centralizations. The first, shown in Figure 4, depicts the network where change at the center in an otherwise centralized network has lead to lower overall centralization. The second, shown in Figure 5, introduces a new case, in which an otherwise decentralized network is rendered centralized by collapsing over time due to a single participant appearing in each frame, but with entirely different 'partners'. Even in a decentralized network the developer with high 'tenure' appears to form a core, in regular discussion with a transient periphery.

In concrete terms these structures might indicate projects at different stages of their lifecycle (as described in [13]). The first, centralized structure might indicate projects on a growth trajectory driven by the creative vision of their leaders in communication with a group of active alpha testers. The second, decentralized structure might indicate a project in a maintenance mode, being tended to by a few long-timers and a transient group of infrequent bug reporters.

3.2 Changes in central members

We can assess the occurrence of change at the center graphically by examining individual centralities over time. In our data, individual outdegree centrality is a measurement of the number of individuals that a participant has replied to, standardized by the total number of participants (the potential audience). For the projects with the highest positive difference between average and overall centralization, we selected the five nodes with the highest average centralization as candidates for being at the center. We then computed their ranks in each

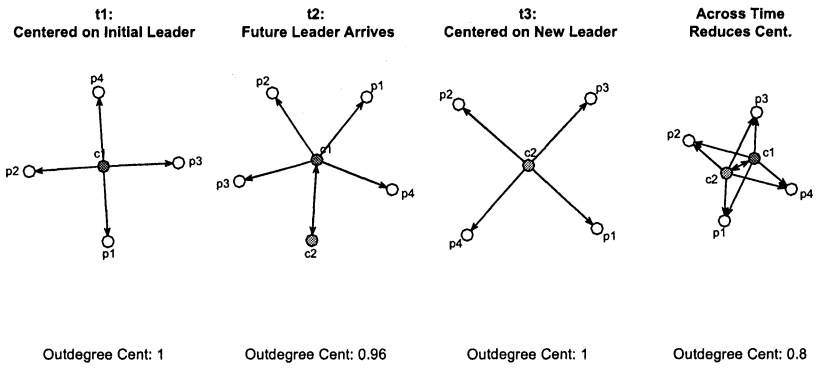

Fig. 4. Ideal Type: Change at the center

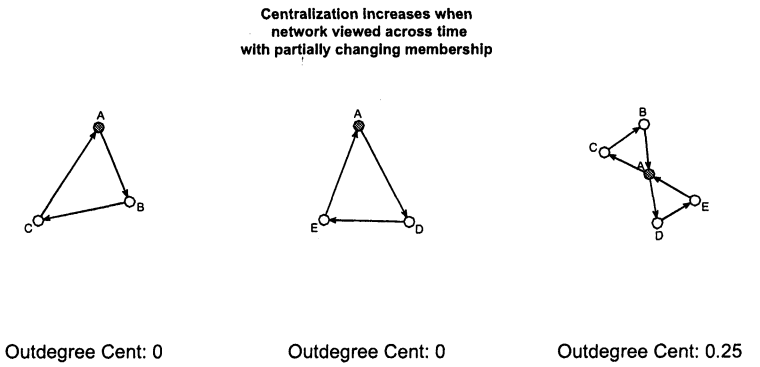

Fig. 5. Ideal Type depicting inequality in tenure

time period and graphed them in Figure 6. When the line ascends to the top (rank 1) it indicates that the node had the highest centralization, on its own, in that period. (Ties were separated by assigning the minimum value for the tied group, so if all lines head down to rank ≤ 5 that indicates that the 'central' position was shared during that period.)

`curl` is plotted first for comparison; it has not undergone change at the center. Its central node, the solid line, has maintained the top rank in individual centralities throughout the time period, shown by the horizontal line at rank 1. In contrast the four projects with highest differences show clear changes in the developer in the most central position. `cplusplus` is the clearest of all, we see that the developer represented by the solid line rapidly assumed the

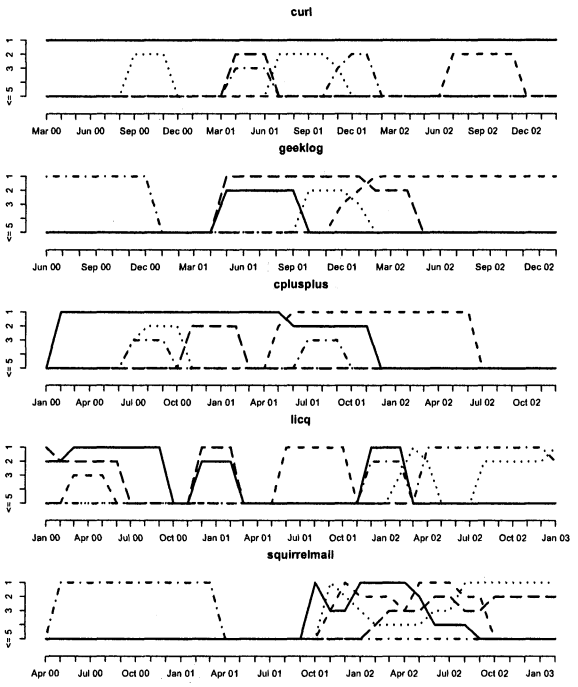

Fig. 6. Individual centrality ranks indicate change at the center

central position in early 2000 and maintained that until May 2001. At that time the developer represented by the single long dashed line emerged as a central participant, first taking the second spot and then assuming the top position until June 2002. Similar patterns are visible in other projects. `squirrelmail` had a dominant center (dot-dashed) through until April 2001 It was not until January 2002 that another relatively stable center, the solid line, emerged but he was soon replaced by the developer represented by the dashed line who was replaced in turn by the dotted line. The graphical analysis suggests that change at the center is a good explanation for the reduction in centralization that occurs when the networks are flattened across time.

The snapshot data allows a numerical assessment of stability at the center two ways for each project in our sample. First we counted the number of developers ever at the top rank of individual centrality, and second we counted the number of times the top rank position changed (we counted a change if the top ranked developer at $t+1$ was different than the developer at t). If there are developers alternating in the center then the second figure will be larger than the first. We expected to find that most projects were more similar to `curl` than to `squirrelmail`, that the node at the center would be stable through the project, quite possibly the project founder.

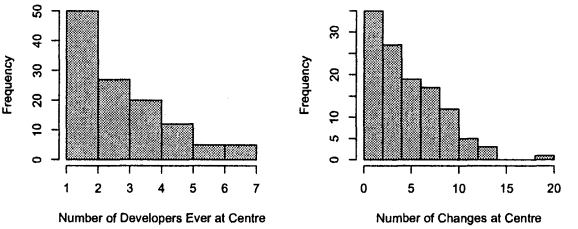

Fig. 7. Change at the center is uncommon

Figure 7 shows the distributions for our two measures of center stability. Among our sample the majority had only ever had one developer ever at the center and seven was the largest count. Leadership changes showed a similar distribution (the measures correlated at $r = 0.73$). This is an interesting finding because it suggests that change at the center of a project is uncommon.

We expected that larger projects, with many more candidates for the center and a greater 'load' on the central participants, would experience more change at the center. However our measures did not show correlation with the number of participants (0.18 and -0.02 respectively); larger projects do not seem more likely to undergo more changes at the center.

The measures of change at the center did show correlation (r=0.4) with the difference between average and overall centralization, lending quantitative support to the graphical exploration of change at the center in Figure 6 and to the hypothesis expressed in [4] at least for the cases with positive differences. We now turn to examine the potential of transient peripheries suggested by 5 above.

3.3 Transient Peripheries?

As an heuristic to understand stability in participation, we measured the number of time windows in which each participant posted a message and expressed that as a percentage of the total number of snapshots of the project's lifetime in our data. Figure 8 shows the distribution of this measure for projects where we had data on at least 10 periods. The data show a highly skewed distribution; the majority of participants are active for only between 10 and 20 percent of the periods in which we had data. This reflects the fact that the mode was activity for just a single period. On the other hand there are a number of projects, like `lyxbugs`, `ucsf-nomad` and `oscar`, that had their participants active in half of the periods examined, indicative of a fairly stable team.

While this finding is interesting on its own and would bear further investigation, it showed low correlation with the differences between overall and average

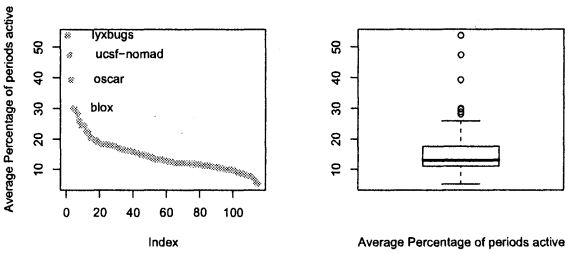

Fig. 8. Most participants are highly transitory

centralization suggesting that the second ideal-type model is not that common amongst our dataset.

4 Discussion

Our initial expectation that a dynamic snapshot analysis would revive our expectation of a pattern of high centralization in FLOSS project communications was not supported. There was no significant difference between the overall and average means and there were a large number of projects that had the opposite reaction, where collapsing the network over time in fact raised their centralization. We found reasonable evidence that changes in leadership played a role in suppressing the expected centralizations but did not find a full explanation for the negative cases.

Nonetheless, our analysis also provides possible insight into project leadership and change. Outdegree centrality in our study is essentially measuring contribution in the bug tracker. Contribution is crucial to leadership of FLOSS projects, partially a result of its self-organization and volunteer nature and partially as a result of its ideological commitment to meritocracy. It is tempting then to make a direct connection between high outdegree centrality and thus a central position, and project leadership.

Caution is called for, however, because this data is only measuring communications contribution, which is controversial as a measure of leadership compared to development contribution. In fact Raymond expects FLOSS leaders to 'speak softly' [12] and Alan Cox provides anecdotal reports of blow-hard 'town councilors' who speak a lot without writing code [1]. On the other hand our data comes from the bug tracker, a place of focused activity, rather than the project mailing lists where 'town councilors' are more likely to be found. Sustained contribution in the bug tracker, answering questions and seeking further information is likely to indicate a participant who is at least important to the project, if not the over-all leader.

An expectation that figures central to a project would be found in the bug tracker is in marked contrast to expectations in proprietary software development teams. Here bug-fixing is likely to be 'grunt work'; a leader in proprietary teams is more likely to be found in an architecting and over-sight role. Empirical work is needed to explore this difference further.

5 Conclusion

This analysis of FLOSS project communications over time has presented three substantive findings:

- We confirmed the finding reported in [4]. Projects vary widely in their social structures between projects even when the networks are analyzed over time. Initial examination of centralization over time within projects also shows substantial variance.
- We found that the majority of projects examined retain a single participant at the center for substantial periods of time, and found that larger projects do not change central participants more often than smaller projects. Perhaps 'Linus' does scale after all (contra McVoy et al [10]), or, more likely, lieutenants face a glass ceiling, collecting below and buffering a still active central actor.
- We provide evidence that a vast majority of project participants are involved for only a very small number of periods, and there is a characteristic power law distribution whereby a very small number are involved for long periods.

This paper, and the longer version available online, also makes a methodological contribution, describing a dynamic analysis of FLOSS project communication and suggesting that collapsing a network over time is not a reliable way to describe social structure as experienced by participants. Finally, the paper also introduces a possible quantitative method for assessing leadership change, a crucial event in virtual team dynamics. The individual centralization rank graphs in Figure 6 identify time periods where qualitative investigation of the project communications would be likely to reveal evidence of leadership change. Thus a dynamic SNA approach can function as a data reduction device. We hope to extend this work by examining the time series, combining it with an analysis of contribution in code repositories and exploring 'concentration' [6], a newly introduced SNA measure of centralization capable of placing a group, rather than an individual at the center of a project

References

1. Alan Cox. Cathedrals, Bazaars and the Town Council. *Slashdot*, 13 October 1998 1998.
2. K. Crowston. A coordination theory approach to organizational process design. *Organization Science*, 8(2):157–175, 1997.

3. Kevin Crowston and James Howison. Hierarchy and centralization in free and open source software team communications. *Knowledge, Technology and Policy*, 18(4), 2005.
4. Kevin Crowston and James Howison. The social structure of open source software development teams. *First Monday*, 10(2), 2005.
5. C. de Souza, J. Froehlich, and P Dourish. Seeking the source: Software source code as a social and technical artifact. In *Proceedings of GROUP '05*, pages 197–206, 2005.
6. Martin Everett and Stephn P Borgatti. Extending centrality. In Peter J Carrington, John Scott, and Stanly Wasserman, editors, *Models and Methods in Social Network Analysis*. Cambridge University Press, 2005.
7. James. Howison, Megan Conklin, and Kevin Crowston. Ossmole: A collaborative repository for floss research data and analyses. In *Proc. of 1st International Conference on Open Source Software*, Genova, Italy, 2005.
8. James Howison and Kevin Crowston. The perils and pitfalls of mining sourceforge. In *Proc. of Workshop on Mining Software Repositories at the International Conference on Software Engineering ICSE*, 2004.
9. M. Huisman and T. A. B. Snijders. Statistical analysis of longitudinal network data with changing composition. *Sociological Methods & Research*, 32(2):253–287, 2003.
10. L. McVoy, E. Raymond, and Others. A solution for growing pains (mailing list thread). email to linux kernal mailing list (30 sep 1998). Available from: http://www.ussg.iu.edu/hypermail/linux/kernel/9809.3/0957.html, 1998.
11. P. Monge and N. Contractor. Emergence of communication networks. In F. Jablin and L Putnam, editors, *New Handbook of Organizational Communication*, pages 440–502. Sage, Newbury Park, CA, 2001.
12. Eric S. Raymond. The Cathedral and the Bazaar. *First Monday*, 3(3), March 1998.
13. Anthony Senyard and Martin Michlmayr. How to have a successful free software project. In *Proceedings of the 11th Asia-Pacific Software Engineering Conference*, pages 84–91, Busan, Korea, 2004. IEEE Computer Society.
14. S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, Cambridge, 1994.
15. H. C. White, S. A. Boorman, and R. L. Brieger. Social structure from multiple networks I. Blockmodels of roles and positions. *American Journal of Sociology*, 81(4):730–780, 1976.