R&D Spillovers:

The 'Social Network' of Open Source Software

Revised December 30, 2008

Chaim Fershtman and Neil Gandal [#]

Abstract: Knowledge spillovers are a central part of knowledge accumulation. The open source model is a form of software development with source code that is typically made available to all interested parties. At the core of this process is a decentralized production process. Using data from Sourceforge.net, the largest repository of Open Source Software (OSS) projects and contributors on the Internet, we construct a two-mode network; a Project network and a Contributor network. Knowledge spillovers may be closely related to the structure of such networks, since contributors who work on the same project(s) may exchange information and knowledge. Defining the number of downloads as output we find that centrality measures of projects (in the projects network) and of developers (in the developers network) are positively associated with the number of downloads of projects. These results imply that knowledge spillovers play an important role in the development of the OSS projects.

Keywords: Knowledge spillovers, Social Network, Open Source, Microstructure of Collaboration

JEL Classification: L17

[#] Fershtman: Department of Economics, Tel Aviv University, Tel Aviv, 69978, ISRAEL and CEPR, fersht@post.tau.ac.il , Gandal: Department of Public Policy, Tel Aviv University, Tel Aviv, 69978, ISRAEL and CEPR, gandal@post.tau.ac.il. We are grateful to the Editors, Matthew Jackson and Xavier Vives and three anonymous referees whose comments and suggestions significantly improved the paper. We thank Rafi Aviav for very helpful research assistance. Research grants from Microsoft and the Net Institute (www.NETinst.org) are gratefully acknowledged. Any opinions expressed are those of the authors.

1. Introduction

Commercial and academic research is typically done by teams. While there are many stories about individuals who achieved major breakthroughs by themselves, the typical R&D project involves teams of researchers who work together on the same project. Working in teams involves exchanging ideas and sharing information. Whenever co-workers collaborate on a joint R&D project, they create knowledge spillovers. Participants of such R&D teams carry over their knowledge to other teams and other projects that they are involved with. Knowledge spillovers thus have two aspects. The most familiar type of spillover is when one innovation provides information and ideas for the development of another innovation.[1] There is however also a more personal and direct spillover among people who work together and exchange information and ideas.

The flow of knowledge and ideas among individuals depends on the details of their collaboration in R&D teams as well as on their social interaction with other researchers. For any structure of R&D teams, one can construct a two-mode weighted network that provides the details of the teams' structure. For the "project network" we say that two R&D projects are connected if there are developers who participate in both R&D projects. The weight of this link depends on the number of developers that participate in the two projects. In the related "developers network," two developers are connected if they work on the same R&D project and the weight of this link depends on the number of projects that the two work together.[2] The focus of this paper is to demonstrate that the structure of these two networks may affect the knowledge spillovers among the different projects and thus may also affect their success.

Detailed information about R&D projects is typically hard to obtain. In particular information regarding the identity of developers who participate in each project is not available. One exception is academic publications which are publicly observed. For example Goyal, van der Leij and Moraga-Gonzalez (2006) constructed the co-authorship network in Economics using data on all published papers that were included in EconLit from 1970-2000 and studied the properties of this network.[3] The recent "open source revolution" provides a unique opportunity to study a two-mode R&D network of projects and researchers and in

[1]For a model of R&D competition with spillovers see for example d'Aspermont and Jacquemin (1988). See also Goyal and Moraga-Gonzalez (2001) who examine the interaction between the architecture of the collaboration network in oligopolistic markets and the firm incentives to invest in R&D.

[2] Thus the researcher network has the same structure as academic co-authorship networks.

[3]Our emphasis in this paper however is not so much on the properties of the contributors' network like in Goyal, van der Leij and Moraga-Gonzalez (2006), but rather on the relationship between these properties of the projects' network and the success of different projects.

particular the relationship between the structure of these networks and the success of different OSS projects.

The open source model is a form of software development with source code that is typically made available to all interested parties; users generally have the right to modify and extend the program.[4] The open source model has become quite popular and often referred to as a movement with an ideology and enthusiastic supporters.[5] At the core of this process is a decentralized production process: open source software development is done by a network of unpaid software developers.[6] Since there are many such projects, these developers may be involved in more than one project and may work with different groups of co-developers in various open source projects. As the development of these projects is done in the public domain and the developers can be identified by their e-mail addresses we can use this information to construct the two-mode network of projects and developers.

The paper uses data from Sourceforge.net to construct a two-mode network of OSS projects and developers. Sourceforge.net is the largest repository of OSS code and applications available on the Internet, with 114,751 projects and 160,104 contributors (in June 2006). We primarily focus on the relationship between the network structure and the success of open source projects. Each SourceForge project page links to a "Developers page" that contains a list of registered team members.[7] The Sourceforge.net information structure is rooted in projects. Using these data, we can construct the project network and the contributor network. Interestingly, both the project network and the contributor network consist of one "giant" connected component and many smaller unconnected networks: in the case of the project network, the giant component contains 27,246 connected projects, while the second largest connected component consists of only 27 projects. In the case of the contributor network, there is a giant component of 55,087 connected contributors and many smaller components.[8] 77% of the contributors worked only on a single project while at the other end of the spectrum, there are a small number of "stars" who work on many projects (there are 344 contributors that worked on ten or more projects).

[4] Open source is different than "freeware" or "shareware." Such software products are often available free of charge, but the source code is not distributed with the program and the user has no right to modify the program.
[5] See for example Raymond (2000) and Stallman (1999).
[6] Having unpaid volunteers is puzzling for economists. For a discussion on the motivation of OSS contributors see Harhoff, Henkel and von Hippel (2003), Lakhani and Wolf (2005), Lerner and Tirole (2002) and Hertel, Niedner, and Herrmann (2002).
[7] Sourceforge.net facilitates collaboration of software developers, designers and other contributors by providing a free of charge centralized resource for managing projects, communications and code.
[8] The second largest component in the contributor network consists of only 196 contributors.

It is not easy to measure the success of open source software. Like other products based on intellectual property, the intellectual property in software (including open source software) is "licensed" for use. In the case of commercial software, however, there are license fees; thus it is possible to determine the number of licenses issued, as well as the revenues earned from these licenses. That is not the case with open source software, which does not have license fees and information on the number of licenses is not available. One way to measure project success is to examine the number of times a project has been downloaded. Although this is not always an ideal measure, downloads is good measure of success for open source software. Downloads are also often used in order to measure the impact of academic papers and articles on the web.[9] Hence, we assume that the number of downloads of open source projects is likely quite correlated with use and value.[10]

The objective of our paper is to examine the relationship between the project and contributor networks characteristics and the success of the OSS projects. Our first goal is to determine whether knowledge spillovers play any role at all in the development of OSS projects. If so, we wish to investigate which type of knowledge spillovers are associated with the success of projects. We examine whether, after accounting for effect of all control variables (the number of developers and other project characteristics), the *degree* of the project is correlated with a larger number of downloads. A significant association between *degree* and downloads will indicate that there is a positive direct spillover effect from having developers who work on other projects. The *degree* of a project is a measurement of the number of projects that are directly connected to the project and thus does not capture the possible indirect knowledge spillovers from projects that are not directly connected. Information may flow between individuals and projects even when they are not directly connected. We examine this issue by introducing into our empirical analysis the network variable *closeness centrality* which measures how far each project is from other projects in the network.[11] When *closeness centrality* is significantly associated with a larger number of downloads (beyond the direct effect of the project's *degree*) we have evidence of indirect knowledge spillovers among the projects.

---

[9] The Social Science Research Network, for example, provides information on the number of downloads for the papers on its website. While there are many economists that may believe that the number of downloads of papers from SSRN is not a good measure of success, our interaction with software engineers indicates that the number of downloads of open source software projects is as a good proxy for success of open source projects.

[10] We will also show that in the case of the Sorceforge.net data, the number of project downloads is especially large for projects selected "project of the month" at SourceForge. This reinforces the notion that downloads is a good measure of success.

[11] *Closeness centrality* defines for every project in the network the inverse of the sum of all distances between the project and all other projects (multiplied by the number of other projects).

There are different types of knowledge spillovers. When the knowledge spillover primarily involves information about the state of the world we would expect that direct and indirect spillovers will play an important role. Research often involves an additional spillover: new and innovative ideas. Ideas that come from different groups, different disciplines or different ways of thinking are often the most valuable ones. In this case simple measures of direct and indirect spillovers are not sufficient to capture the importance of these complementary ideas. A priori it is not possible to evaluate what type of knowledge spillover is more valuable. Clearly in some research projects the information spillovers are more useful, while in other types of projects only new and innovative ideas can facilitate breakthroughs.

In our empirical analysis we capture the importance of complementary ideas by investigating the relationship between downloads and the *betweenness centrality* of each project. The *betweenness centrality* of a node is defined as the proportion of all geodesics between pairs of other nodes that include this node, where a geodesic is the shortest path between two nodes. *Betweenness* captures the notion that a node is considered central if it serves as a critical juncture (bridge) between other nodes.

Our analysis shows that additional contributors are associated with higher output (downloads), both for projects in the giant component and projects outside of the giant component, but the increase in downloads associated with an increase in contributors is much larger for projects in the giant component. This robust result obtains even though the average number of contributors is higher for projects in the giant component. We then show that the association between the *degree* of the project and the number of downloads is positive and statistically significant both for projects inside the giant component and for projects outside the giant component. This suggests that direct knowledge spillovers may play an important role in the development process of these projects.

We then turn to examine the presence of indirect spillovers. We limit our analysis to the giant component and find that the *closeness centrality* of a project is positively and significantly associated with more downloads. This effect obtains even after controlling for the effect of direct spillovers captured by the *degree* of the project. (The coefficient associated with *degree* remains statistically significant.) This result suggests that knowledge spillovers are not restricted to direct spillovers; knowledge may flow between projects even when they do not share common developers.

We next find that *betweenness centrality* is highly associated with a higher number of downloads. Controlling for *betweenness*, we find that *degree* is not positively associated with the number of downloads. This result suggests that the "critical" knowledge spillovers being

transferred among OSS projects are likely to be innovative "complementary ideas" rather than large amounts of information.

We are careful not to attach a causal interpretation to our results because it is not possible to determine from the data whether increases in network measures (e.g. *degree*, *betweenness* or *closeness*) increase downloads or whether highly successful projects attract more productive contributors.

Our paper is related to the literature on 'the effect of network structure on behavior' (e.g., Ballester, Calvo-Armengol and Zenou (2006), Calvo-Armengol and Jackson (2004), Ioannides and Datcher-Loury (2005), Goeree, McConnell, Mitchell, Tromp and Yariv (2007), Jackson and Yariv (2007), and Mobius and Szeidl (2007)).[12]

Our focus is on knowledge spillovers between projects that are linked by a collaboration network. In this respect our paper is related to Goyal and Moraga-Gonzalez (2001) who examine the interaction between the architecture of the collaboration network in oligopolistic markets and the firm incentives to invest in R&D.[13] Their setting analyzes both the network formation in which firms establish links in order to enjoy R&D spillovers and the choice of R&D effort given this network among firms. We do not incorporate network formation and effort choice in our analysis. The main motivation for an OSS developer is peer recognition incentives and/or reputation, which may eventually be rewarded in the job market.[14] Hence the objective of a programmer is to have his name included in the list of contributors. Typically, there is a threshold level, and if the contribution is beyond the threshold level, the individual is rewarded by having his name appear in the list of contributors.[15] If the objective of a programmer is to have his name included in the list of contributors, we expect that he will contribute up to the threshold level, but not much beyond it. Similarly, there is not a project management team deciding whether or not to establish a link with another project in order to exploit spillovers. After all, we are talking about projects with no secrets; everyone can freely download and examine them with no restrictions.

Our paper is also closely related to Calvo-Armengol, Patacchini, and Zenou (2008) and Ahuja (2000) who also consider the relationship between network structure and performance.

---

[12] For more general surveys on the role of social networks in the functioning of the economy see Jackson (2006, 2008) and Goyal (2007) and for general methods and applications see Wasserman and Faust (1994).

[13] An important part of their setup is the fact that the firms compete in an oligopolistic market and R&D effort by one firm also benefits some of its competitors (with whom they are linked) and changes the market equilibrium.

[14] See Lerner and Tirole (2002) for the reputation argument and Hars and Ou (2001), Hertel, Niedner, and Herrmann (2002) for the peer recognition motivation.

[15] In the XFree86 project, for example, in order to become a developer, one must submit an accepted patch (see Halloran and Scherlis (2002)).

Calvo-Armengol et. al. use data on an adolescent friendship network and focus on how the existing network structure affects pupils' school performance. Ahuja (2000) examines the relationship between the network formation of technical collaboration among firms in the chemical industry (from 1981-1991) and innovation, as measured by U.S. patents.

2. The Two-Mode Network of Contributors and Projects

We obtained our data by "spidering" the website http://SourceFourge.net, which is the largest Open Source software (OSS) development web site.[16] The data was retrieved from SourceForge.net during June 2006 and includes 114,751 projects and 160,104 contributors who were listed in these projects. The contributors are identified by unique user names they chose when they registered as members in SourceForge. The site's information structure is rooted in projects. The interface of SourceForge.net allows almost all of the information about the projects to be viewed by anyone.[17] Each project has a "Project page" which is a standardized 'home page' that links to all the services and information made available by SourceForge.net for that project. The project page itself contains important descriptive information about the project, such as a statement of purpose, the intended audience, license, operating system etc.

Each project page links to a "Statistics page" that shows various activity measures, such as the number of downloads. Each project page also links to a "Developers page" that has a list of registered team members. This list is managed by the project administrators who are also listed as team members. The assumption in this paper is that the site members who are listed as project team members were added to the list because they made a contribution to the project that involved investment of time and effort. A project is thus seen as a collaborative effort by its team members, or *contributors*.

The data we obtained from SourceForge.net form a two-mode-network of projects and contributors. A two-mode-network is a network partitioned into two types of nodes, e.g. projects and contributors. We can use the two-mode network to construct two different one-mode networks: (i) the contributors' network and (ii) project network.[18]

---

[16] Spidering is term used to describe recursive algorithms used to traverse a website page-by-page and automatically extract desired information based on forms and content pattern.

[17] A very small number of projects block certain data from being accessed by anyone who isn't a project team member.

[18] We construct our project network by defining that two projects are linked if there are contributors that work in both of them. One can construct different types of networks that depend on application, language etc. i.e., two projects are connected if they are written for the same application. In our empirical analysis we control for these variables. While defining networks based on application and language does capture some aspects of knowledge

Contributor Network:

- The nodes of this network are the contributors, i.e., the distinct names (or emails) of the contributors.
- There is a link between two different contributor nodes if the two contributors participated in at least one OSS project together.
- Each link may have a value which reflects the number of projects in which the contributors jointly contributed.

Projects Network:

- The nodes of this network are the OSS projects.
- There is a link between two different project nodes if there are contributors who participate in both projects.
- Each link may have a value which reflects the number of contributors that participate in both projects.

The following table shows the distribution of contributors per project and projects per contributor for the two-mode-network at Sourceforge.net.

Project network		Contributor network	
Contributors per project	Number of projects	Projects per contributor	Number of contributors
1	77,571	1	123,562
2	17,576	2	22,690
3-4	11,362	3-4	10,347
5-9	6,136	5-9	3,161
10-19	1,638	10-19	317
20-49	412	20-49	26
≥50	56	≥50	1
Total Projects	**114,751**	Total Contributors	**160,104**

Table 1: The distribution of contributors per project and projects per contributor

spillovers, the thrust of our research is on knowledge spillovers created by individuals. We thus focus on the networks that are defined by having common contributors.

Table 1 shows that 68% of the projects hosted at Sourceforge.net have just a single contributor.[19] An additional 15% of the projects have two contributors. At the other end of the spectrum, there are 1,638 projects with 10-19 contributors and 468 projects with more twenty or more contributors. Similarly, Table 1 shows that 77% of the contributors worked on a single project, while an additional 14% contributed only to two projects. Thus more than 90% of the open source contributors worked on just one or two projects. At the other end of the spectrum, there are a small number of "stars" who work on many projects: 3,161 contributors worked on 5-9 projects, while 344 contributors worked on ten or more projects.

There are six levels of development that range from the planning stage to a mature status. There is an additional status reserved for projects that are inactive. Table 2 below provides the distribution of the development status for the single contributor and the multi-contributor projects. As is evident from this table the two distributions are similar. The possibility that the single contributor projects are in some way infant projects thus seems remote. In any case, we will control for the time for which the project has been in existence.

Development status	Relative frequency in "single contributor" projects	Relative frequency in "multi contributor" projects
1 – Planning	21%	21%
2 - Pre-Alpha	17%	16%
3 – Alpha	18%	17%
4 – Beta	22%	23%
5 – Production/Stable	18%	20%
6 – Mature	1%	2%
Inactive	2%	2%

Table 2: Development Status

2.1 The Network of Contributors:

For the contributor network, there is a link between contributors *i* and *j* if they have worked on at least one project in common. The set of contributors can be divided into components such that all of the contributors in a component are connected to one another and there is no sequence of links among contributors in different components. The distribution of the components is shown in Table 3a. There is a "giant" component, which consists of

[19] While these projects do not provide links between contributors, such contributors who work on multiple projects provide links among projects.

55,087 contributors, or approximately 45% of the contributor network. The table shows that there are many small components as well.

Component size (Contributors)	Components (sub networks)
55,087	1
196	1
65-128	2
33-64	27
17-32	152
9-16	657
5-8	2,092
3-4	4,810
2	8,287
1	47,787

Table 3a: Distribution of component size

Degree	Number of contributors
0	47,787
1	22,133
2	14,818
3-4	20,271
5-8	20,121
9-16	16,228
17-32	10,004
33-64	5,409
65-128	2,040
129-256	802
257-*505*	491

Table 3b: Distribution of Degree

For every contributor in the network, we can define the *degree* as the number of links between that contributor and other contributors in the network.[20] Table 3b shows the distribution of *degree* in the contributor network. There are 47,787 contributors who work only in single contributor projects. At the other end of the spectrum 491 "star" contributors worked on projects in common with more than 256 other contributors.

2.2 The Network of Projects:

In the project network, a node is a project and there is a link between two projects if and only if there are contributors who have contributed to both projects. Table 4a shows that the project network consists of one “giant” connected component with 27,246 projects and many smaller unconnected components. The giant component contains approximately 24% of the projects at the Sourceforge website. It is indeed striking that the second largest “network” consists of only 27 projects. The *degree* of a project is the number of other projects with which that project has a link. Table 4b shows the distribution of *degree* for the

[20] Hence, a contributor who worked on a single project with four other contributors has a *degree* of four. Similarly, a contributor who worked on two projects, each of which had two additional contributors (who only worked on one of the two projects), would also have a contributor *degree* equal to four.

project network. Two-thirds of the project have *degree* less than or equal to one. At the other end of the spectrum, 370 projects have *degree* greater than thirty-two.

Size	Connected components
27,246	1
17-*27*	36
9-16	234
5-8	1,013
3-4	3,419
2	8,020
1	51,093

Table 4a: Distribution of component size

Degree	Number of projects
0	51,093
1	22,926
2	12,709
3-8	22,004
9-32	5,649
33-64	290
≥65	80

Table 4b: Distribution of degree

2.3 Measuring Success/Output in the Project Network

Defining or measuring the success of an open source project is problematic. There are no prices and no 'sales'. The projects are in the public domain and there is no need to request permission or to provide payment for using the OSS. One way to measure project success is to examine the number of times a project has been downloaded. Although this is not always an ideal measure, downloads is good measure of success for open source software.[21] Unlike downloads of academic papers, users will not typically download a project (and its code) unless it will be useful to them for some task.[22] Software downloads has been used as a measure of success also by Grewal et al. (2006) who refer to downloads as a "market-based" measure of popularity and note that the number of downloads is the typical proxy for "sales" when software products are freely available.[23]

Every month, the Sourceforge.net staff chooses a "project of the month." Although we do not know the exact criteria that are employed in choosing the "project of the month," these projects are likely to be very "successful." We obtained data on the "project of the

[21] Downloads are also often used in order to measure the impact of academic papers and articles on the web. The Social Science Research Network, for example, provides information on the number of downloads for the papers on its website.

[22] While developing the paper, we had the opportunity to interact with software engineers. They believe that downloads are a good proxy for success of open source projects.

[23] In some cases, the number of downloads is small relative to the number of contributors. In such cases, the number of downloads may be affected by the fact that developers may need to download the code of the project when working on the project. When we restrict our analysis to projects with more than 200 downloads, and a download/contributor ratio of at least ten-to-one (so that the number of downloads is at least an order of magnitude larger than the number of developers), our results remain qualitatively unchanged: hence our results are robust to the possibility of 'developer' downloads.

month" for the forty-two month period ending in June 2006. The "project of the month" projects have an especially large number of downloads.[24] "Project of the month" projects are typically in advanced stages (stages 4,5, and 6); thirty-eight of the forty-two projects of the month projects are either in stage 4, stage 5, or stage 6. The thirty-eight "project of the month" projects in advance stages had on average 6,028,560 downloads, versus 30,206 downloads (on average) for the other 35,821 projects in advanced stages. The median number of downloads for "project of the month" projects in advance stages was 1,154,469 versus 483 for other projects in advance stages. This suggests that the number of project downloads is an attractive measure of use and value.

There are several different download measures that we could use: (i) the total number of downloads since the project was initiated at Sourceforge.net (ii) the maximum number of downloads in any month, and (iii) the number of recent downloads. The correlation among these download measures is, however, quite high. Since it contains the most information, we chose to use the total number of downloads in our analysis. Henceforth, when we refer to downloads, we mean the total number of downloads and denote *downloads* as the total number of downloads for the forty-two month period for which we have data. We further define *ldownloads* ≡ ln(1+*downloads*), where "ln" means the natural logarithm. Since it may take some time for projects to reach an "equilibrium" level of contributors, we will also perform robustness checks by conducting the analysis for projects that have been in existence for at least two years.

2.4 Control Variables (Project Characteristics)

In addition to downloads, there is a group of control variables that includes the amount of time that the project has been in existence, the stage of development, the number of operating systems for which the program was written, the number of languages in which the program is written, as well as several other control variables. The variables are as follows:

- The variable *years_since* is the number of years that have elapsed since the project first appeared at Sourceforge: *lyears_since*=ln(*years_since*).

- The variable *cpp* is the number of contributors that participated in the project: *lcpp*=ln(*cpp*)

[24] Given that there are only forty-two such "projects of the month," we cannot use this as our measure of success.

- The dummy variable *ds_j* refers to the stage where j ranges from one to six. There is an additional stage, denoted *inactive*, which means the project is no longer active. See Table 2. A few of the projects are considered to be in multiple stages. Hence, for a particular project, it is possible that both ds_3 and ds_4 could be equal to one.

- The variable *count_trans* is the number of languages in which the project appears including English. Virtually all of the projects (95%) are available in English. The other popular languages include German (5% of the projects), French (4%), and Spanish (3%). *lcount_trans*=ln(*count_trans*)

- The variable *count_op_sy* is the number of operating systems (i.e., formats) in which the project is compatible. Some of the projects are available for several operating systems. The main operating systems in which the projects were written include Windows (32% of the projects), Posix (26% of the Projects), and Linux (21% of the Projects. *lcount_op_sy*=ln(*count_op_sy*)

- The variable *count_topics* is the number of topics included in the project description. Popular topics include the Internet (16% of the projects), software development (14%), communications software (11%), and games & entertainment software (10%). *lcount_topics*=ln(*count_topics*)

- The variable *count_aud* is the number of main audiences for which the project was intended. The main audiences are developers (35% of the projects), end users (30% of the projects), and system administrators (13% of the projects). Some of the products are intended for multiple 'main audiences' while other projects are not intended for these main audiences, but rather just for niche audiences, i.e., just for a particular industry (i.e., telecommunications) or just for very sophisticated end users. *lcount_aud*=ln(1+*count_aud*)

Clearly, there are different ways to construct these variables. For example, we could have simply counted the key operating systems, or used dummy variables for these operating systems. Similarly, we could have defined dummy variables for 'main audiences' or we could have added up the number of main audiences together with the number of niche audiences. We chose the definitions that seemed most natural. Our main results regarding the number of contributors and the network variables are robust to alternative definitions of these control variables.[25]

3. Knowledge Spillovers

Our analysis will be carried out in several steps. We first wish to examine if knowledge spillovers play any role in the development of OSS projects. Our focus is on

[25] Contributor effort is not observable. As we discussed in the introduction, the main reward to OSS contributors is being included in the list of contributors. Thus the incentive they have is to provide the sufficient effort to accomplish this status. Hence, effort is not likely correlated with network measures or the control variables – and hence, the absence of data on effort does not bias our results.

spillovers that are created by interaction between different developers who collaborate on the same OSS project. We thus start by looking at the *degree* of each OSS project in the project network and examine if, accounting for the effect of all the control variables (the number of developers and other project characteristics), *degree* is associated with a larger number of downloads. A significant effect implies that there is a positive spillover effect from developers that work on other projects.

The next step will be to examine if there are indirect knowledge spillovers from projects that are not directly connected. That is, does information flow between individuals even when they do not work together on the same OSS project? We examine this issue by introducing the network variable *closeness centrality*.[26] *Closeness centrality* measures how far each project is from other projects in the network. If -- after controlling for the direct effect of the project's *degree* -- *closeness centrality* is significantly associated with a larger number of downloads, we have an evidence of indirect knowledge spillovers among the projects.

The last step will be to examine the hypothesis that the spillover of ideas is not just due to the number of direct and indirect links. Consider the project network shown in figure 1 in the Appendix. This is the large component of the strongly connected network, where, a 'strong' link between two projects exists only when they have at least two developers in common. We can see that this network has an interesting structure. There are three clusters or groups of highly connected projects.[27] The three clusters remain connected as part of one component only because project 81 is connected to all these three groups. Project 81 has a relatively small *degree*, but its position in the network is unique and central. This position is relevant for an additional type of knowledge spillover. Assume for example that the three groups in Figure 1 describe a friendship network among people. Moreover assume that each cluster in this network is a group of friends that are similar in their backgrounds and preferences. Suppose that the knowledge transmitted in this network is about the quality of a restaurant or a movie. In this case the information received from members of the same group would be more valuable than information received from members of other groups. On the other hand, there are research settings where ideas come from groups of researchers who think and solve problems in different ways: In such a case, the more effective knowledge spillovers and/or more effective complementary ideas come from outside of the research group's inner core. In these cases, the position of project 81 (Figure 1), which is linked to

[26] Since *closeness centrality* can be calculated only for a connected network, we do this analysis for projects in the giant component only.

[27] Each has some periphery networks that are connected only to one particular group.

several different clusters of projects, may benefit from valuable knowledge spillovers from the different clusters of projects.

We capture this effect by introducing *betweenness centrality* into our empirical analysis. *Betweenness centrality* is defined as the proportion of all geodesics between pairs of other nodes that include this node. *Betweenness* captures the notion that a node is considered "central" if it serves as a valuable juncture between other nodes. Project 81 in Figure 1 indeed has relatively high *betweenness* (and relatively low *degree*).

4. Empirical Analysis: Characteristics Associated with Success of Projects

We estimate a simple log/log model of the form $ldownloads_i = \alpha + \beta N_i + \gamma C_i + \varepsilon_i$, where the subscript *i* refers to the project. N_i is the natural logarithm of the "network variables" and C_i is the natural logarithm of the control variables.[28] For binary ([0,1]) variables, we, of course do not employ logarithms; ε_i is a random error term.

We have data on 114,450 observations for all of the network variables as well as on *years_since*.[29] However, data on the stage of development and the count variables are incomplete; data on all of the control variables are available only for 66,511 projects. Since there is no selection issue,[30] we use only the data on the 66,511 projects for which we have complete information.[31] In section 4.1, we conduct an analysis using these projects and examine the association between *degree* (and the control variables) and success. In sections 4.2 and 4.3 we follow up this analysis by examining the giant component in detail (18,697 projects for which there is complete information), which enables us to include the variables *betweenness* and *closeness* in the analysis.[32] We then perform robustness checks by examining established projects only (section 4.4) and projects with more than one contributor (section 4.5).

---

[28] The relationship between the number of contributors *downloads* is likely non-linear: additional contributors are likely associated with a larger number of downloads, but the marginal effect of each additional contributor declines as the number of contributors increases. The same is likely true for the relationship between the network variables and downloads as well. This suggests that a "log/log" model is appropriate. When we estimate a linear model, the results are qualitatively similar to the "log/log" model, but the linear model fits less well, i.e., the adjusted R-squared is smaller. Hence, we employ the "log/log" model throughout the paper.

[29] There are 114,751 total projects, but we are missing data on downloads for a small number of them (301).

[30] See Griliches (1986) and Greene (1993).

[31] We do not discard the information that these projects provide concerning the network structure and the values of network variables that are included in the database. Further, it is comforting to know that our main results regarding the association between the number of contributors and success and the centrality variables and success are qualitatively unaffected by whether we use the full data set, or the observations for which we have data on all relevant variables. These results are available from the authors on request.

[32] The values of *degree, closeness centrality*, and *betweenness centrality* are calculated using the software program Pajek, which is a software program for large network analysis. See http://pajek.imfm.si/doku.php.

4.1 Direct spillover effect

In our setting, direct knowledge spillovers imply a spillover only from projects that share at least one contributor. We examine this effect by using the *degree* of the project which is the total number of projects, with which it has at least one contributor in common and define the variable *ldegree*=ln(1+*degree*).

The effect of the *degree* (as well as other effects) may depend on whether the project is in the giant component or not; we therefore introduce the variable *"giant_comp*" which is a dummy variable that takes on the value one if the project is in the giant component, and takes on the value zero otherwise.

In order to allow for the possibility that the association between *degree* and downloads and between the number of contributors and downloads depends of whether the project is inside or outside of the giant component, we also include the following interaction variables in the analysis:

- *lgiant_degree* = ldegree*giant_comp,
- *lgiant_cpp* = lcpp*giant_comp,

By including the interaction variables, we allow for the possibility that there will be different download "elasticities" for projects in and projects outside of the giant component.[33]

Descriptive statistics of the variables are shown in Table A1 in the appendix. Table A1 shows that projects in the giant component have on average more downloads than projects outside of the giant component (42,751 vs. 10,959). Further, projects in the giant component are on average (i) older than projects outside of the giant component (3.63 years vs. 2.70 years), (ii) have more contributors (3.84 vs. 1.61), and (iii) have a larger *degree* (6.26 vs. 1.18).[34] The results of a regression with all 66,511 observations are shown in the first column of Table 5.

The effect of the number of contributors: The estimated coefficients show that the association between downloads and the number of contributors is positive – projects with more contributors have a greater number of downloads. For projects outside of the giant component, the estimated "contributor" elasticity is 0.46. That is, a one percent increase in the number of contributors is associated with a 0.46 percent increase in the number of

[33] The addition of different slopes for the control variables based on whether the project was inside or outside of the giant component has no effect on the main results regarding the number of contributors and the *degree* of the project.

[34] Correlations among the independent variables in the regressions are shown in Table A2 in the appendix.

downloads. This effect is statistically significant. The estimated "contributor" elasticity is virtually twice as large for projects in the giant coefficient: 0.90 (0.46+0.44). The difference in the estimated "contributor" elasticity between projects in the giant component and projects outside of the giant component is statistically significant: additional contributors are associated with greater increases in output for projects in the connected (giant) component than in the non-connected component. This result obtains despite the fact that there are many more contributors (on average) for projects in the giant component (3.84 vs. 1.61).

One possible explanation for this result is that the contributors to projects in the giant component are more skilled than the contributors who work on projects outside of the giant component. Alternatively, it could mean that there are knowledge spillovers among projects with ties that enhance the productivity of those who work together on these projects.

The effect of project's degree: The *degree* elasticity, i.e., the association between the *degree* of the project and the number of downloads, is positive and statistically significant both for projects inside the giant component and for projects outside of the giant component. This suggests that projects with a higher *degree* are associated with higher output (or downloads). For projects outside of the giant component, the *degree* elasticity is 0.19, while the *degree* elasticity for projects in the giant component is 0.14. Both of these magnitudes are statistically significant from zero; the difference in the magnitudes is not significantly different from zero. This result suggests that there are direct knowledge spillovers among the projects.

The effect of the control variables: The estimated coefficient of *lyears_since* is positive (1.42) and statistically significant. This suggests that projects that have been active longer have more downloads, and the estimated coefficient suggests that a doubling of the time a project has been active is associated with 142% more downloads.[35] The estimated coefficients on the stage variables have the expected signs. By and large, projects that are in more advanced stages are associated with more downloads. Similarly, projects written for several operating systems, projects available in more languages, projects written for more main audiences, and projects that span more topics are associated with more downloads as well.

[35] In section 4.4, we show that our results regarding the association between the number of contributors and downloads and the association between the centrality measures *(betweenness and closeness*) and downloads is robust to excluding projects that are less than two years old.

Dept Variable: Ldownloads	Regression 1 (All 66,511 Projects)		Regression 2 (Giant Component - 18.697 Projects)	
Independent Variables	Coeff.	T-stat	Coeff.	T-stat
Constant	0.72	17.76	5.71	10.76
lyears_since	1.42	60.66	1.68	31.14
lcount_topics	0.23	9.07	0.18	3.66
lcount_trans	0.35	11.73	0.43	7.85
lcount_aud	0.36	10.44	0.41	5.52
lcount_op_sy	0.11	5.95	0.18	4.92
ds_1	-1.96	-60.57	-2.02	-32.24
ds_2	-0.60	-17.58	-0.80	-11.89
ds_3	0.89	25.83	0.64	9.76
ds_4	1.86	57.21	1.78	29.08
ds_5	2.72	79.97	2.58	40.65
ds_6	2.12	27.07	2.01	15.31
Inactive	0.45	6.11	0.35	2.54
Lcpp	0.46	18.71	0.61	16.71
Ldegree	0.19	9.45	-0.13	-3.12
Giant_comp	-0.21	-3.86		
lgiant_cpp	0.44	12. 05		
lgiant_degree	-0.05	-1.26		
Betweenness			0.48	12.15
Closeness			0.38	1.76
# of Observations	66,511		18,697	
Adjusted R-squared	0.41		0.41	

Table 5: Regression Results: Dependent Variable: ldownloads

4.2 Indirect Knowledge Spillover Effects

An indirect knowledge spillover exists when there is spillover between two projects that do not have any developers in common. The indirect route can be by a learning mechanism such that a developer who participates in project X learns something when he participates in project Y and then "shares" this knowledge with the another project X developer who uses it when he works on project Z.

We examine this issue by using the network variable *closeness centrality* which defines for every project the inverse of the sum of all distances between the project and all other projects (multiplied by the number of other projects).[36] Formally, for any two nodes $i, j \in N$, the distance or degree of separation between them (denoted $d(i, j)$) is the length of the geodesic between them where a geodesic is the shortest path between two nodes.

[36] As discussed above, *closeness* measures how far each project is from the other projects in the network.

Closeness centrality of a node is defined as the inverse of the sum of all distances between the node and all other nodes, multiplied by the number of other nodes, so that it lies in the range [0,1].[37] *Closeness centrality* is calculated as follows:

$$(1) \qquad C_C(i) \equiv \frac{\#N - 1}{\sum_{j \in N} d(i,j)}$$

For the purpose of our empirical analysis we define *lcloseness* = ln(0.05+*closeness*).[38] We restrict our analysis only to the giant component as the distance between two projects that are not connected is not properly defined.

When we introduce the variable *lcloseness* into the regression (and restrict the analysis to the giant component of 18,697 observations), we find that the effect of *closeness centrality* is positive and significant (coefficient=0.70, t=3.21). The effect of *degree* remains positive and significant (0.13, t=2.77) although somewhat smaller. The result involving *closeness centrality* suggests that (in addition to the direct knowledge spillovers associated with *degree*) we have evidence for indirect knowledge spillovers among the projects.

4.3 Spillovers of complementary Ideas

Our next step is to introduce the variable *betweenness centrality* into our analysis. For a network of size "#N," the *betweenness centrality*, or *betweenness*, of a node is defined as the proportion of all geodesics between pairs of other nodes that include this node. [39]

Formally, the *betweenness* of a node i is given by

$$(1) \qquad C_B(i) \equiv \frac{\sum_{\substack{j<k \\ i \notin \{j,k\} \subseteq N}} \left[\gamma_{jk}(i) / \gamma_{jk} \right]}{(\#N - 1)(\#N - 2)/2}$$

where γ_{jk} is the number of distinct geodesics between the nodes j and k which are distinct from i, and $\gamma_{jk}(i)$ is the number of such geodesics which include i.[40]

Betweenness captures the notion that a node is considered "central" if it serves as a valuable juncture between other nodes. For the purpose of our empirical analysis we define *lbetween* = ln(.0001+*betweenness*).[41]

---

[37] See Freeman (1979), pp. 225-226 and Wasserman and Faust (1994), pp. 184-185.
[38] The reason we add such a small number is because the mean value of *closeness* is 0.14.
[39] See Freeman (1979), pp. 230-231 and Wasserman and Faust (1994), pp. 189-190.
[40] The denominator of (1) is the maximum possible value for the numerator, and thus standardizes the measure in the range [0, 1].

In the second regression in Table 5, we introduce both *betweenness* and *closeness* centrality measures to our analysis. Since *betweenness* and *closeness* are only comparable across linked networks, this regression is done for the giant component only. Table 5 shows that the estimated *betweenness* elasticity (0.48, t=12.15) is positive and statistically significant. The estimated *closeness* elasticity (0.38, t=1.76) is statistically significant as well (at the 0.92 level.)

These results suggest that it is not just the ties among projects that matter for downloads, but how the projects are tied together and their position in the network.[42] Projects that sit in critical information flows have greater downloads. The estimated *degree* elasticity is negative (-0.13) in this regression. This suggests that controlling for *betweenness* and *closeness* centrality, there is not a positive association between the number of downloads and the *degree* of the project: the two centrality measures (*betweenness* and *closeness*) are more important for the number of downloads than the *degree* of the project.[43]

The estimated coefficient of *lyears_since* is again positive (1.68) and statistically significant. The estimated coefficients on the stage and count variables again have the expected signs and are qualitatively similar to those in the first regression in Table 5.

4.4 Robustness of Results to Inclusion of Established Projects Only

Nascent projects may not have reached a steady-state number of contributors. Personnel additions are probably more likely for relatively new products. It is important to know whether the results are robust to using only established projects in the analysis. Hence, we re-did the regressions in Table 5 for projects that had been in existence for at least two years.[44] Our results are qualitatively unchanged.

In the case of the first regression, we are left with 44,638 observations (or 67% of the observations) when we restrict the analysis to projects that had been in existence for more than two years. For projects outside of the giant component, the estimated "contributor" elasticity is 0.51 (versus 0.46 in Regression 1 in Table 5), while the estimated "contributor" elasticity for projects in the giant coefficient is 0.91 (virtually the same as in the first regression in Table 5). The difference in the estimated "contributor" elasticity between

---

[41] The reason we add such a small number is because the mean value of *betweenness* is 0.00028.

[42] Indeed the estimated contributor elasticity is lower in regression 2 in Table (0.61 vs. 0.91in the first regression in the table) because of the positive association between the network variables and downloads.

[43] This does not mean that there are no direct spillovers. Because *degree* and *betweenness centrality* are correlated in our data set (see Table A2(b) in the appendix), *betweenness centrality* picks up the direct spillover as well.

[44] The median age of projects in our data set as of June 2006 was 2.66 years.

projects in the giant component and projects outside of the giant component is again statistically significant.

The estimated coefficients on *degree* for projects outside and inside the giant component are positive and statistically significant (0.21 and 0.14 respectively), nearly the same as in the first regression in Table 5. The estimated coefficients on the stage and count variables again have the expected signs.

When we run a regression analogous to the second regression in Table 5 for projects in the giant component that have been in existence for more than two years, we are left with 14,749 projects (or nearly 79% of the observations). The estimated contributor elasticity (0.63 in this new regression versus 0.61 in the second regression in Table 5) is again positive, statistically significant and virtually unchanged. The estimated *betweenness* elasticity (0.47 in this new regression versus 0.48 in the second regression in Table 5) is again positive, statistically significant and virtually unchanged. The estimated *closeness* elasticity (0.31 in this new regression versus 0.38 in the second regression in Table 5) is not statistically significant (t=1.24). The estimated *degree* elasticity is -0.11 (-0.13 in the second regression in Table 5) is virtually unchanged. (For ease of presentation, these two regressions appear in the Appendix in Table A3.)

4.5 Robustness of results to projects with more than one contributor

In this section, we repeat the analysis for projects with more than one contributor. We are left with 25,422 projects when we restrict the analysis to projects that have more than one contributor. For projects outside of the giant component, the estimated "contributor" elasticity is 0.46 (virtually the same as in Regression 1 in Table 5), while the estimated "contributor" elasticity for projects in the giant coefficient is 1.01 (versus 0.90 for the same as in the first regression in Table 5). The difference in the estimated "contributor" elasticity between projects in the giant component and projects outside of the giant component is again statistically significant.

The estimated coefficients on *degree* for projects outside and inside the giant component are positive and statistically significant (0.15 and 0.18 respectively), and quite similar to the first regression in Table 5. The estimated coefficients on the stage and count variables again have the expected signs.

When we run a regression analogous to the second regression in Table 5 for projects in the giant component with more than one contributor, we are left with 11,814 projects with more than one contributor. The estimated contributor elasticity (0.75 in this new regression versus 0.61 in the second regression in Table 5) is again positive and statistically significant.

The estimated *betweenness* elasticity (0.45 in this new regression versus 0.46 in the second regression in Table 5) is again positive, statistically significant and virtually unchanged. The estimated *closeness* elasticity is 0.44 in this new regression versus 0.38 in the second regression in Table 5. This coefficient (with a t-value of 1.53) is not statistically significant at the 0.90 level. The estimated *degree* elasticity, -0.14, is virtually unchanged from the second regression in Table 5. [45]

The robustness analysis in sections 4.4 and 4.5 reinforce our main results: (i) the association between the number of contributors and the number of downloads is higher for projects inside the giant component than it is for projects outside of the giant component and (ii) *Betweenness* centrality is the centrality measure most highly associated with the number of downloads. *Closeness* centrality appears also to be positively associated with downloads, but the effect is not statistically significant over all specifications. Controlling for the correlation between these two measures of centrality (*betweenness* and *closeness*), *degree* is not positively associated with the number of downloads. (For ease of presentation, these two regressions appear in the Appendix in Table A4.)

5. The Importance of Strong Ties

So far we defined two projects to be linked if there was at least one contributor in common between them. But the potential of spillovers between projects may depend also on the number of contributors that participated in the two projects. To capture this effect we change the definition of a link and focus only on "strong" links. Two projects are 'strongly' linked if and only if they have at least two contributors in common. That is, we define a new network in which the nodes are still projects, but the links are only 'strong' links.

Redefining the network has a dramatic effect on it structure. Previously in a network in which one contributor in common was sufficient for a link, there was a giant component of 27,246 projects. In the new network, the largest component of strongly connected projects consists of only 259 projects. There are four smaller strongly connected components with between 50-75 projects. Figure 1 in the Appendix shows the network structure of the largest component in the "strongly connected" network. A comparison of the median number of downloads between projects in the strongly connected component and other projects in the

[45] While programming language may be associated with success, programming language and centrality measures are likely to be uncorrelated. In such a case, the coefficients we estimated are unbiased, even though we do not control for "programming language." For discussion regarding the choice of programming language in Sourceforge projects, see Cottam and Lumsdaine (2008).

giant component suggest that a stronger connection is associated with more downloads. See Table 6.[46]

Group	# of projects	Mean # downloads	Median # downloads
Strongly Connected Component	259	82,238	2,035
Other Projects in Giant Comp.	26,897	30,230	98

Table 6: Strongly Connected Component vs. Other Projects in Giant Component.

We then run a regression employing three additional variables:

(i) A dummy variable for projects in the strongly connected component, denoted *strong*.

(ii) (ii) The variable *l*strong_*degree* = *ldegree* strong*.

(iii) (iii) The variable *l*strong_*cpp* = *lcpp* strong*.

We again find that additional contributors are associated with an increase in output, but that this increase is much higher for projects in the strongly connected component, than other projects in the giant component. The estimates of the contributor elasticity are 0.61 for projects in the giant component that are not part of the strongly connected component and 1.58 for projects that are in the strongly connected component (see Table 7). This suggests that strong ties make a large difference in the contributor elasticity. The other results are (not surprisingly) virtually unchanged from the second regression in Table 5.

[46] The same qualitative result obtains if we restrict the analysis to projects in stages 4-6. In this case, the projects in the strongly connected component have a median of 11,230, while other projects in the giant component in the same stages have a median of 1,431.

Dept Variable: ldownloads	Giant Component Projects with data on stage & count variables	
Independent Variables		
Constant	5.65	10.63
lyears_since	1.68	31.17
lcount_topics	0.18	3.63
lcount_trans	0.43	7.92
lcount_aud	0.41	5.53
lcount_op_sy	0.18	4.95
ds_1	-2.02	-32.25
ds_2	-0.80	-11.89
ds_3	0.64	9.74
ds_4	1.78	29.07
ds_5	2.58	40.61
ds_6	2.02	15.32
Inactive	0.36	2.54
Lcpp	0.61	16.50
Ldegree	-0.13	-3.09
strong_comp	-0.19	-0.23
lstrong_cpp	0.97	3.00
lstrong_degree	-0.56	-1.64
Betweenness	0.47	11.95
Closeness	0.38	1.74
# of Observations	18,697	
Adjusted R-squared	0.41	

Table 7: Regression Results Adding Variables for Largest Strongly Connected Component

6. Contributor Network Characteristics and Project Success

Until this point, we focused on project network characteristics and the way they were associated with the success of the projects. Our next step is to include the contributor network characteristics and to examine their relation to project success.

6.1 The effect of contributor characteristics.

We derive the network characteristics for each contributor. In order to examine the relationship between these characteristics and project success, we need to look at the group of contributors who participate in each project and define measures that capture the network characteristics of these contributors. For each project we form a list of contributors and construct the following variables:

(i) Average *degree* of the contributors in a project.

(ii) The average *closeness centrality* of the contributors to a project.

(iii) The average *betweenness centrality* of the contributors to a project.

Clearly this is only one way to aggregate the contributors' characteristics. In principal one can look at the whole distribution of these characteristics but for the sake of our empirical analysis we will use only the average of these variables.

The above variables differ respectively from the *degree* of a project, the *betweenness centrality* of a project and the *closeness centrality* a project. For example, consider project A with two contributors (denoted I and II), each of whom works on one other project. This means that project A has a (project) *degree* equal to two. Further suppose that contributor "I" also works on project B, and that there are three other distinct contributors on project B. Similarly, suppose that contributor II also works on project C, and that there are again three additional distinct contributors on project C. The "contributor" *degree* of contributor I equals four (since he/she participates with four other contributors in two different open source projects). Similarly, the contributor *degree* of "II" is four as well. Hence, the average contributor *degree* of project A is four.

While the *degree* of the project and the average *degree* of the contributors to a project are relatively highly correlated in our data set (0.44),[47] there is virtually no correlation between the *closeness centrality* of a project and the average *closeness centrality* of its contributors (0.03) and between the *between centrality* of a project and the average *betweenness centrality* of its contributors (0.02).

We first ran a regression similar to the second regression in Table 5 with the three contributor network variables instead of the three project network variables. We find that only the average *closeness centrality* of the contributors to a project is significant (coefficient=0.14, t=1.81). The average *betweenness centrality* of the contributors to a project and the average *degree* of the contributors on a project are insignificant. Further the adjusted R-squared of this regression is lower than the adjusted R-squared of the second regression in Table 5, which employed the project network variables. (This regression is shown in Table A5 in the appendix.)

We thus added the variable "average *closeness centrality* of the contributors" to the second regression in Table 5; we find that after controlling for the project controls and network characteristics, the average *closeness centrality* of the contributors who participate in the project is positively correlated with the success of the project, but that the effect is not

[47] This is the correlation between the natural logarithm of the variables, since we use those in the analysis.

statistically significant (coefficient =0.09, t=1.26.)[48] The interpretation of this result is similar to our previous argument regarding indirect knowledge spillovers. The "quality" of a contributor is affected by the knowledge spillovers that he enjoys from other contributors even when they do not participate in the same projects. But these knowledge spillovers are subject to the standard decay effect.

6.2 The "star" effect.

We define a "star" as a contributor who worked on five or more projects. An interesting question is if having a "star" in the team of developers has an effect on the success of a project. This variable is not derived from one of the two networks that we constructed but it is an important contributor characteristic.

To examine this, we add a dummy variable (denoted star) -- which takes on the value one if the project has at least one star and takes on the value zero otherwise -- to the second regression in Table 5.[49] We find that although the effect is not statistically significant (coefficient=0.10, t=1.41), the presence of a "star" contributor is positively correlated with the success of the project. This effect, which obtains even after controlling for of projects' *degree* and centrality measures, suggests that star contributors are associated with positive knowledge spillovers beyond what is accounted for by our network measures. This effect may mean that the 'star' positively affects the success of the projects in which he/she participates. Alternatively, this may be a result of having a group of developers who are eager to contribute to successful projects. The estimated coefficients on *degree*, *betweenness* and *closeness* are unaffected by the addition of "star."

7. Concluding Remarks

Knowledge spillovers are an important part of any learning or an R&D process. There are two possible mechanisms that facilitate such spillovers. One possibility is that an individual (or a firm) observes the outcome of an R&D effort of another individual, i.e., new technology or a patent, and learns about its own R&D process. A more direct mechanism is the interaction between different individuals who communicate with their colleagues, exchange emails, switch jobs and projects and collaborate in different research ventures. The first type of spillover is easier to model as a dynamic process in which any advance or

[48] The effects of 'project' *degree*, 'project' *closeness* and 'project' *betweenness* are unaffected by the addition of the average *closeness centrality* of the contributors.

[49] 92% of the projects outside of the giant component do not have a star. 45% of the projects in the giant component have at least one star.

success involving one project positively affects the success of related projects. The second type of learning spillover crucially depends on the specific network of interaction among individuals who are involved in the learning process. It is much more difficult to extract information regarding who talks with whom and how knowledge is shared between individuals. The OSS project network provides a unique opportunity for tracing such interactions and for examining the effect of the properties of the "collaboration" network on the success of different projects. A similar study can be done with respect to academic research in which it is possible to construct the network of collaboration. While the collaboration network has been constructed for different academic fields, it is important to take the next step and relate the properties of these collaboration networks to outcomes ("successes"), which can be measured, for example, by citations of different papers.

References

Ahuja G., (2000),"Collaboration Networks, Structural Holes, and Innovation: A Longitudinal Study," *Administrative Science Quarterly*, 45: 425-455.

Ballester, A. Calvo-Armengol, A., and Y. Zenou (2006) "Who's Who on Networks: Wanted the Key Player" *Econometrica*, Vol.74 (5), 1403-1417.

Calvo-Armengol A., E. Patacchini and Y. Zenou (2008) "Peer effect and social networks in education" *Review of Economic Studies*, forthcoming.

Calvo-Armengol, A. and M. Jackson (2004), "The effect of Social Networks on Employment and Inequality", *American Economic Review*, Vol. 94(3) 426-454.

Cottam, J., and A. Lumsdaine, "Extended Assortitivity and the Structure in the Open Source Development Community," International Sunbelt Social Network Conference, January 2008. International Network for Social Network Analysis,"available at: http://www.osl.iu.edu/publications/prints/2008/Cottam2008ExtendedAssortitivity.pdf

D' Aspermont, C. and A. Jacquemin (1988), "Cooperative and Noncooperative R&D in Duopoly with Spillovers", *American Economic Review*, 78(5), 1133-1137.

Freeman, L. (1979), "Centrality in Social Networks: Conceptual Clarification." *Social Networks*, 1: 215-239.

Goeree, J.K., M.A. McConnell, T. Mitchell, T. Tromp and L. Yariv (2007), "Linking and Giving Among Teenage Girls", Working Paper, Cal Tech.

Goyal, S. (2007), *Connections: An Introduction to the Economics of Networks*, Princeton University Press.

Goyal, S., M. and J.L Moraga-Gonzalez, (2001), "R&D Networks," *Rand Journal of Economics* 32: 686-707

Goyal, S., M. J. van der Leij and J.L Moraga-Gonzalez, (2006), "Economics: Emerging Small World" *Journal of Political Economy*, 114, 403-412.

Greene, W., (1993), "Econometric Analysis, Second Edition. New York: MacMillan Publishing Company.

Grewal R, Lilien, G., and G. Mallapragada (2006). Location, location, location: How network embeddedness affects project success in Open Sources Systems? Management Science, 52(7): 1043-56.

Griliches, Z., (1986), "Economic Data Issues," in *Handbook of Econometrics*, Volume 3, Griliches, and M. Intriligator, editors. Amsterdam: North Holland Publishing Company.

Halloran, T., and W. Scherlis (2002), "High Quality and Open Source Software Practices, mimeo, available at http://opensource.ucc.ie/icse2002/HalloranScherlis.pdf

Harhoff, D., J. Henkel, and E. von Hippel (2003), "Profiting from voluntary spillovers: How users benefit by freely revealing their innovations, *Research Policy* 32: 1753-1769.

Hars, A., and S. Ou (2001), “Working for free? - Motivations for participating in open source projects,” *International Journal of Electronic Commerce*, 6: 25-39

Hertel, G., Niedner, S. and S. Herrmann (2003), “Motivation of software developers in open source projects: An internet-based survey of contributors to the Linux kernel,” *Research Policy*, 32, 1159-1177.

Ioannides, Y.M. and L. Datcher-Loury (2005), "Job Information Networks, Neighborhood Effect and Inequality" Journal of Economic Literature, Vol. 42(4), pp. 1056-1093.

Jackson, M.O., (2006), “The Economics of Social Networks,” In *Proceeding of the 9th World Congress of the Econometric Society* (ed. R. Blundell, W. Newey and T. Persson). Cambridge University Press.

Jackson, M.O. (2008), "Social Networks in Economics", forthcoming in the *Handbook of Social Economics* (edited by Benhabib, Bisin and Jackson), Elsevier.

Jackson, M. and L. Yariv (2007), "The Diffusion of Behavior and Equilibrium Structure on Social Networks" *American Economic Review (papers and Procedures)*.

Lakhani, K., and R. Wolf (2005), “Why Hackers Do What They Do: Understanding Motivation and Efforst in Free Open Source Projects, In: Feller/Open, J. Fitzgerland, S. Hissam, K. Lakhani (eds.), Perspectives on Free and Open Source Software, MIT Press, Cambridge.

Lerner, J., and J. Tirole (2002), "Some Simple Economics of Open Source" *Journal of Industrial Economics*, 52: 197-234.

Mobius, M. and A. Szeidl (2007), "Trust and the Social Collateral" Working Paper, Harvard University.

Raymond, E. (2000), “The Cathedral and the Bazaar", available at http://www.catb.org /~esr/writings/cathedral-bazaar/cathedral-bazaar/.

Stallman, R., (1999), “The GNU Operating system and the Free Software Movement,” in Dibona, C., Ockman, S., and M. Stone editors, *Open Sources: Voices from the Open Source Movement,* O’Reilly, Sepastopol, California.

Wasserman, S., and K. Faust, (1994), *Social Network Analysis: Methods and Applications*, Second Edition. New York and Cambridge, ENG: Cambridge University Press.

Appendix A: Tables

Table A1: Descriptive Statistics for 66,511 Projects with data all variables

VARIABLE	MEAN	STD. DEV.	MIN	MAX
Projects Not in the Giant Component (N= 47,814)				
downloads	10,959	938,658	0	2.00e+08
years_since	2.70	1.67	0	6.64
count_topics	1.51	0.81	1	7
count_aud	1.21	0.69	0	3
count_op_sy	2.08	1.58	1	21
count_trans	1.27	0.92	1	40
ds_1	0.25	0.43	0	1
ds_2	0.20	0.40	0	1
ds_3	0.20	0.40	0	1
ds_4	0.26	0.44	0	1
ds_5	0.21	0.41	0	1
ds_6	0.02	0.12	0	1
Inactive	0.02	0.14	0	1
Cpp	1.61	1.52	1	42
Degree	1.18	2.14	0	23
Star	0.08	0.28	0	1
Projects in the Giant Component (N= 18,697)				
Downloads	42,751	1,062,802	0	1.18e+08
years_since	3.63	1.70	0.08	6.65
count_topics	1.65	0.89	1	7
count_aud	1.34	0.70	0	3
count_op_sy	2.25	1.69	1	22
count_trans	1.38	1.66	1	45
ds_1	0.22	0.42	0	1
ds_2	0.17	0.38	0	1
ds_3	0.21	0.41	0	1
ds_4	0.30	0.46	0	1
ds_5	0.29	0.45	0	1
ds_6	0.03	0.17	0	1
Inactive	0.03	0.16	0	1
Cpp	3.84	6.72	1	338
Degree	6.26	8.53	1	299
Betweenness	0.00028	0.0015	0	0.12
Closeness	0.14	0.021	0.061	0.22
Star	0.45	0.49	0	1

Table A2(a): Correlation among all Variables: N=66,511

	ldown	lyears	lcpp	ldegree	ds1	ds2	ds3	ds4	ds5	ds6	inact	ltop	ltrans	los	laud
ldownloads	1.00														
lyears_since	0.29	1.00													
lcpp	0.23	0.18	1.00												
ldegree	0.24	0.22	0.44	1.00											
ds1	-0.38	0.03	0.00	-0.07	1.00										
ds2	-0.20	0.01	-0.01	-0.05	-.04	1.00									
ds3	0.04	0.03	-0.01	0.00	-0.2	-0.16	1.00								
ds4	0.25	0.04	0.03	0.05	-0.26	-0.24	-.19	1.00							
ds5	0.38	0.09	0.09	0.14	-0.23	-0.22	-.21	-.14	1.00						
ds6	0.11	0.06	0.04	0.07	-0.05	-0.05	-.05	-.05	0.01	1.00					
inactive	0.01	0.06	-0.01	0.02	-0.05	-0.04	-.05	-.06	-.05	-.01	1.00				
ltop	0.13	0.18	0.09	0.10	0.04	0.02	.04	0.06	0.08	.04	0.00	1.00			
ltrans	0.09	0.05	0.10	0.05	0.03	-0.01	-.03	0.05	0.08	.04	0.01	0.09	1.00		
los	0.12	0.29	0.09	0.05	0.04	0.02	0.01	0.02	0.05	.01	0.01	0.14	0.07	1.00	
laud	0.15	0.29	0.06	0.11	0.02	0.02	0.03	0.05	0.08	0.04	0.01	0.20	0.06	0.15	1.00

Note:

ltop = lcount_topics
ltrans= lcount_trans
los=lcount_op_sy
laud=lcount_aud

Table A2(b): Correlation among all centrality variables (Giant Component: N=18,697)

	lcpp	degree	lbetween	lcloseness	star
lcpp	1.00				
ldegree	0.49	1.00			
lbetween	0.71	0.64	1.00		
lcloseness	0.26	0.41	0.36	1.00	
star	0.17	0.74	0.26	0.27	1.00

Table A3: Regressions for projects at least two years old

Dept Variable: ldownloads	Regression '1' (All Projects)		Regression '2' (Giant Component)	
Independent Variables	Coeff.	T-stat	Coeff.	T-stat
Constant	-0.55	-5.98	4.60	7.22
lyears_since	2.21	36.06	2.27	19.97
lcount_topics	0.24	7.64	0.20	3.47
lcount_trans	0.38	10.15	0.39	6.34
lcount_aud	0.31	6.88	0.33	3.74
lcount_op_sy	0.15	6.90	0.21	5.32
ds_1	-2.12	-53.37	-2.08	-29.05
ds_2	-0.68	-16.07	-0.88	-11.26
ds_3	0.81	19.08	0.55	7.31
ds_4	1.84	45.84	1.68	24.31
ds_5	2.74	65.51	2.58	35.94
ds_6	2.14	23.12	2.00	13.76
inactive	0.45	5.31	0.41	2.68
lcpp	0.51	15.91	0.63	14.80
ldegree	0.21	8.05	-0.11	-2.24
giant_comp	-0.13	-2.06		
lgiant_cpp	0.40	8.88		
lgiant_degree	-0.067	-1.47		
betweenness			0.47	10.49
closeness			0.31	1.24
# of Observations	44,638		14,749	
Adjusted R-squared	0.40		0.38	

Table A4: Regressions for projects with more than one contributor

Dept Variable: Ldownloads	Regression '1' (All Projects)		Regression '2' (Giant Component)	
Independent Variables	Coeff.	T-stat	Coeff.	T-stat
Constant	0.51	5.53	4.04	7.99
lyears_since	1.60	36.71	1.74	23.60
lcount_topics	0.24	5.54	0.24	3.74
lcount_trans	0.41	9.12	0.44	6.80
lcount_aud	0.46	7.69	0.42	4.33
lcount_op_sy	0.13	4.28	0.16	3.47
ds_1	-2.13	-40.18	-2.17	-26.99
ds_2	-0.77	-13.65	-0.88	-10.12
ds_3	0.78	13.79	0.53	6.28
ds_4	1.89	35.76	1.75	22.56
ds_5	2.75	49.51	2.52	31.19
ds_6	2.05	16.46	1.83	11.17
inactive	0.28	2.12	0.36	1.80
Lcpp	0.46	7.94	0.75	13.90
ldegree	0.15	3.98	-0.14	-2.56
giant_comp	-0.60	-5.88		
lgiant_cpp	0.55	7.61		
Lgiant_degree	0.03	0.57		
betweenness			0.45	10.11
closeness			0.44	1.53
# of Observations	25,422		11,814	
Adjusted R-squared	0.43		0.40	

Table A5: Regression Using Contributor Characteristics

lcontributor *degree* = ln(Average *degree* of the contributors in a project)

lcontributor *closeness* = ln(Average *closeness centrality* of contributors to a project)

lcontributor *betweenness* = ln(Ave. *betweenness centrality* of contributors to a project)

Dept Variable: Ldownloads	(Giant Component)	
Independent Variables	Coeff.	T-stat
Constant	-1.05	-0.94
lyears_since	1.72	32.04
lcount_topics	0.19	3.67
lcount_trans	0.44	8.10
lcount_aud	0.44	5.88
lcount_op_sy	0.17	4.81
ds_1	-2.01	-32.01
ds_2	-0.78	-11.49
ds_3	0.66	10.03
ds_4	1.80	29.39
ds_5	2.62	41.23
ds_6	2.04	15.52
inactive	0.38	2.50
lcpp	0.93	34.29
lcontributor *degree*	-0.02	-0.92
lcontributor *closeness*	-0.19	-1.55
lcontributor *betweenness*	0.14	1.81
# of Observations	18,697	
Adjusted R-squared	0.40	

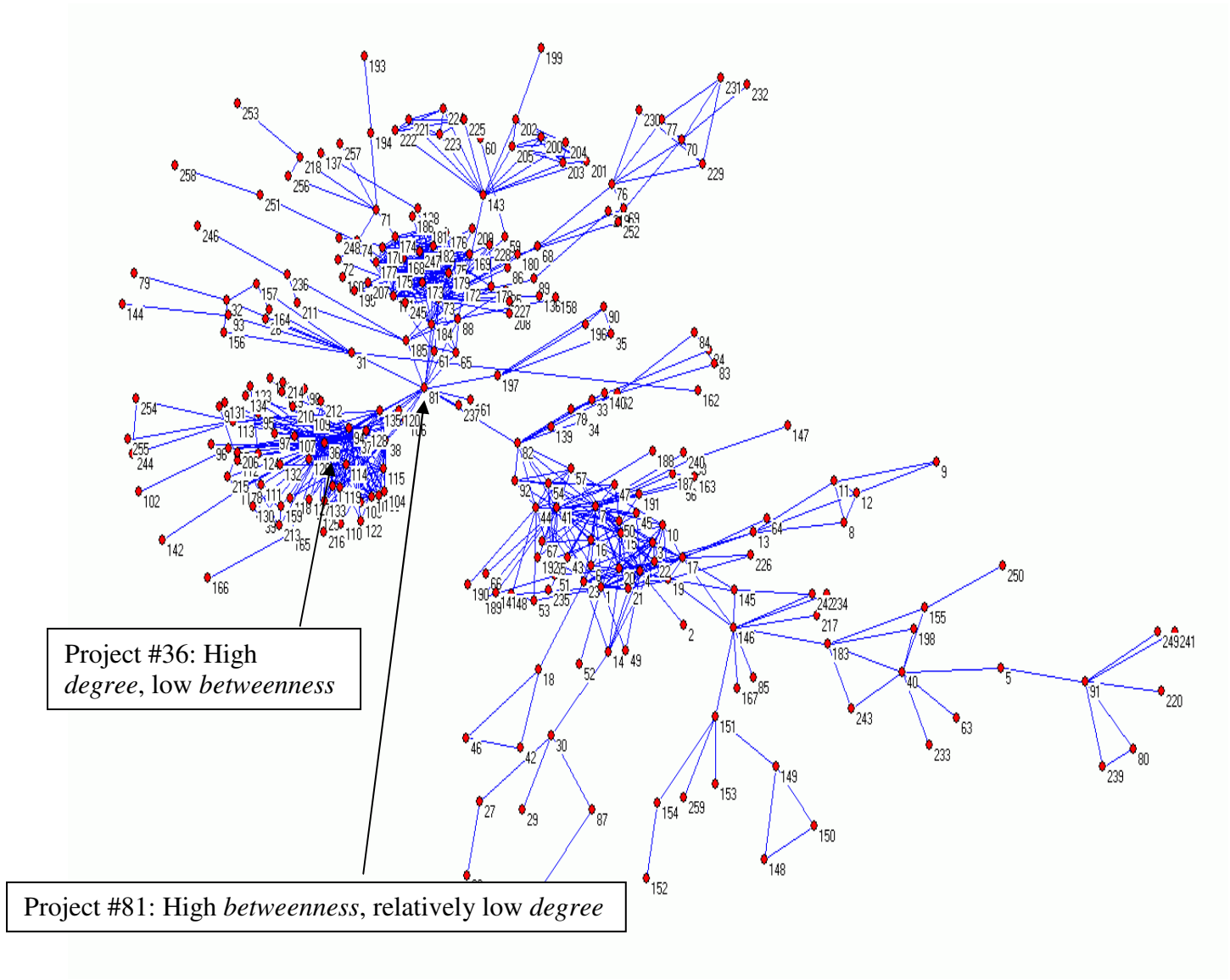

Figure 1: Projects in strongly connected component