Herding a Deluge of Good Samaritans: How GitHub Projects Respond to Increased Attention

Danaja Maldeniya
dmal@umich.edu
University of Michigan

Ceren Budak
cbudak@umich.edu
University of Michigan

Lionel P. Robert Jr.
lprobert@umich.edu
University of Michigan

Daniel M. Romero
drom@umich.edu
University of Michigan

ABSTRACT

Collaborative crowdsourcing is a well-established model of work, especially in the case of open source software development. The structure and operation of these virtual and loosely-knit teams differ from traditional organizations. As such, little is known about how their behavior may change in response to an increase in external attention. To understand these dynamics, we analyze millions of actions of thousands of contributors in over 1100 open source software projects that topped the GitHub Trending Projects page and thus experienced a large increase in attention, in comparison to a control group of projects identified through propensity score matching. In carrying out our research, we use the lens of *organizational change*, which considers the challenges teams face during rapid growth and how they adapt their work routines, organizational structure, and management style. We show that trending results in an explosive growth in the effective team size. However, most newcomers make only shallow and transient contributions. In response, the original team transitions towards administrative roles, responding to requests and reviewing work done by newcomers. Projects evolve towards a more distributed coordination model with newcomers becoming more central, albeit in limited ways. Additionally, teams become more modular with subgroups specializing in different aspects of the project. We discuss broader implications for collaborative crowdsourcing teams that face attention shocks.

CCS CONCEPTS

• **Human-centered computing → Computer supported cooperative work**; **Open source software**.

KEYWORDS

crowdsourcing, attention shocks, coordination, GitHub, PSM

ACM Reference Format:
Danaja Maldeniya, Ceren Budak, Lionel P. Robert Jr., and Daniel M. Romero. 2020. Herding a Deluge of Good Samaritans: How GitHub Projects Respond to Increased Attention. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380272

1 INTRODUCTION

Collaborative crowd communities have recently become commonplace within the information economy. They represent a significant departure from traditional organizations in terms of worker motivation, organizational structure, and operating dynamics. Community members are driven by non-monetary motivations such as reputation and collective identity [6]. Further, these communities consist of loosely-knit and geographically dispersed teams with limited formal structure, roles, or routines. This is in contrast to organizations, which have clearly defined boundaries, authority structures, and formal routines [5]. Crowd communities such as open source software (OSS) projects exhibit growth, maturation, and decline driven by both endogenous and exogenous processes similar to more-traditional organizations. However, due to their differences from traditional organizations, our understanding of these crowds is limited. This is especially true for teams that are growing rapidly due to exogenous shocks that draw increased attention [21, 44, 46].

A common mechanism by which projects receive such shocks on GitHub is by appearing on the trending page. The default view of the trending page displays 25 projects that have demonstrated a recent growth in productivity and attention [8]. Projects on the trending page have the potential to attract significant attention, from users and future contributors. This represents both an opportunity and a challenge for their members. They may be able to expand their workforce by attracting committed and capable new contributors. Yet, their capacity to coordinate and remain responsive to their audience might be strained by a deluge of prospective users and contributors [21]. Furthermore, the vision of the original team may be at odds with the aspiring newcomers, leading to friction and negative interactions that may undermine further growth.

Motivated to characterize these challenges, opportunities, and their consequences, we study the behavior of new contributors and the adaptive response of existing members from the perspective of organizational change [10]. We first characterize the scale of growth and the interests and commitment of newcomers. Then we inspect how core members adjust their work routines in response. Finally, we employ a theoretical framework of distributed leadership to contextualize the evolution of the coordination structure.

Our results show that trending results in a veritable explosion in attention and engagement. Types of engagement include starring the project to indicate interest, using the project software and reporting issues with it, suggesting additional features and contributing code and other content to the project. However, the vast majority of this external attention is restricted to shallow short term interactions with the project—more so than external engagement under *normal* conditions. Despite this, the scale of the attention shock means that there is still substantial growth in team size and the work being done. Subsequently, core members shift toward coordinating the growing workforce and move away from direct production. Even with this new administrative orientation, members struggle to keep up with increased coordination requirements leading to backlogs, especially with regards to outsiders, such as taking longer to respond their engagements. Finally, we find that as members adopt a more comprehensive approach to managing the burgeoning workforce, they allow outsiders to unofficially fill in moderately important positions. In addition, members introduce technological solutions to improve work quality and efficiency—all in line with a more distributed coordination structure.

In sum, we (i.) characterize the scale and nature of shocks resulting from a project being thrust into the limelight and (ii.) shed a light on the resulting collaboration dynamics between the original team and interested outsiders from the community. This understanding will provide the necessary background for better equipping such teams to manage and take advantage of the deluge of good Samaritans drawn to projects during times of rapid growth.

2 RELATED WORK

Organizational Growth and Change. Organizational scholars have studied the impact of rapid growth in organization size. Early research focused on the benefits of such rapid growth such as an increase in organizational resources [3, 23, 43]. Scholars reasoned that these increases can allow organizations to seek out more opportunities for profits [16] and that they can reduce internal conflict as there is less need for internal competition for resources [3].

At the same time, organizational scholars have become aware of the dysfunctional consequences of growth. First, increases in size can lead to greater work complexity [7, 43] and, thus, difficulties in coordination [3, 27, 32]. Second, as organizations grow, relationships between employees become less personal—decreasing organizational cohesion [27]. Organizations often struggle with ensuring that these new employees learn their shared norms and work practices [43]. This is especially problematic when growth occurs rapidly [27]. In some cases reduction in cohesion is accompanied by increases in conflict [32]. Finally, increases in complexity and decreases in cohesion contribute to ineffectiveness and inefficiency [32]. This can lead to organizational processes being strained, stopping the organization from meeting its goals [43].

Recently, organizational scholars have shifted toward identifying approaches to mitigate the dysfunctions that result from growth. According to this stream of research, organizations must change their management style and structure to adapt to growth [27, 32]. To accomplish this, organizations must decentralize decision making and introduce more formal rules and procedures [3, 43]. Both decentralizing and introducing rules and procedures allow organizations to rapidly respond to changes in the environment ,while maintaining control to facilitate the coordination of work. Organizations should also restructure to ensure that individuals or sub-units do not get overwhelmed [32]. This often involves creating new sub-units and assigning them new work and/or redirecting work to them from existing sub-units. Finally, organizations must build and maintain relationships among existing and new members [32].

Drawing from these findings, we develop hypotheses for how GitHub teams respond to rapid growth due to going trending. However, we also consider the important differences between crowd collaborations and traditional organizations. For instance, traditional organizations have well-defined expectations in terms of employee productivity. In contrast, crowds are characterized by volunteers, with different levels of commitment, contributing due to non-monetary motivations [6]. As a result, participation in these communities is often characterized by a long-tailed distribution where most contributors do very small chunks of shallow work while a small core group performs the bulk of the work including overall coordination. [25, 33]. This workload inequality makes coordination of volunteers more manageable [2, 22, 35, 36].

GitHub. Prior work on GitHub has focused on understanding the collaborative dynamics on the platform. Researchers have studied the social and collaborative interactions that take place on the site and how such interactions impact the software. Qualitative studies, mainly based on interviews, have found that users infer each other's goals, expertise, and interests based on their logged actions and profile information. This information is then used to make decisions about which projects to contribute to and how to organize collaborations [12, 28, 40]. Other work focused on measuring large-scale statistical properties such as the power-law-like distribution of the number of contributors and watchers per project, the low levels of reciprocity in the followers network, the geographical distribution users [26], and how coordination efforts by developers scale with the size of a project [36]. Researchers have also characterized the properties of GitHub repositories at large scale including the fraction of personal and inactive repositories and the fraction of repositories that use pull requests [20].

Another line of research focused on identifying the properties and routines of successful GitHub collaborations. Characteristics indicative of success include diversity in gender and tenure [41], the use of automated processes such as continuous integration [42], teams with a high fraction of members with a history of collaboration [9], and low levels of multi-tasking [39]. Rather than characterizing collaboration patterns on GitHub or identifying successful collaboration dynamics, our goal in this paper is to understand how such dynamics change when exposed to attention shocks.

Shocks on Crowd Collaboration. Past research has investigated the effects of external shocks on crowd collaborations in other crowdsourcing platforms such as Wikipedia [21, 44–46]. This work is unique in two important ways. First, OSS development involves substantially more complex and diverse work than Wikipedia. Projects on GitHub incorporate the hallmarks of crowdsourcing collaboration in openness and promoting external engagement, much like Wikipedia. However, they also demonstrate a clear insider vs. outsider dichotomy in terms of authority enabled by access privileges. Further, these projects face growth and competitive pressures from other similar projects that also rely on the same audience and volunteers for success. Second, Github provides tools for setting access privileges, categorizing, tracking and reviewing work as well as project management. That allows us to explore more nuanced questions regarding response of crowd collaborations to shocks that parallel what is known about traditional organizations.

3 HYPOTHESES DEVELOPMENT

In this section, we build a research framework informed by prior work in organizational change management as well as participation and coordination dynamics of crowd collaborations.

3.1 Community Engagement

The GitHub trending project page is the primary mechanism by which the platform introduces high quality repositories to the community [4]. Therefore, it serves as a source for developers and users to find repositories to contribute to and use [19]. Furthermore, the trending page provides a star button for each trending repository. Therefore, we posit that being featured in the GitHub trending list will lead to a substantial increase in community engagement with the repository. Thus, this is our first hypothesis.
Hypothesis 1: Trending will increase community engagement with a repository.

(a) Trending will increase observed community interest, measured through starring and forking behavior.
(b) Trending will lead to an increase in external contributions.

3.2 Growing Pains

Organizational theory suggests that traditional organizations with rapidly growing sales or market share are chronically short of resources and labor. This commonly leads to employee hires to compensate for growth. This, in turn, causes a number of growing pains as the existing organizational structure is strained by the increased number of new and untrained employees and new tasks associated with growth [13, 17]. We expect a similar trend for the repositories that are featured on the GitHub trending page. These repositories are likely experience a sudden increase in the number of external contributors (Hypothesis 1.b). We posit that this sudden increase will similarly strain the capacity of the core repository team. Therefore, we expect that increased workload for the members will translate into delays and backlogs in tasks.
Hypothesis 2: The shock will strain the capacity of the project and create backlogs of tasks.

3.3 Adaptation

We draw on organizational theory to build hypotheses as to how the members of a trending GitHub repository will respond to the increased attention and contributions from the community,

3.3.1 Work Routines. In firms where the labor force is rapidly growing, employees from the pre-growth period—being more experienced and knowledgeable about firm culture—are compelled to take on more management responsibility [10, 17]. Similarly, we posit that GitHub repository members would have to spend more time responding to queries and coordinating the work of the increased number of outside, and potentially inexperienced, contributors. Correspondingly, we expect them to also scale back their own development work.
Hypothesis 3: Members will take on a more administrative role

(a) Members will do more organizational work, such as responding to and directing external contributors.
(b) Members will do less development work.

3.3.2 Coordination. Growing organizations manage the increasing complexity of coordination by adopting a more decentralized and modular coordination style. They can achieve this through empowering lower-level employees, specialization of roles or the division of work among overlapping working groups [10, 17, 30, 43]. However, this flexible structure can lead to a decline in its core values and culture, which play an important role as a cohesive force. To compensate, management may choose to make its values explicit through formalized procedures and highly visible symbols and slogans [13, 17, 34, 43]. Here, we employ Gronn's Distributed Leadership framework as the lens for viewing these phenomena in GitHub [15, 29]. Gronn proposes that organizations that employ a distributed approach to coordination will demonstrate three broad properties; (1.) They encourage workers to dynamically form temporary teams for spontaneous collaborations, (2.) Employees establish relationships and responsibilities through collaborations and receive recognition for their contributions, and (3.) Working practices are institutionalized as part of the organizational governance. We operationalize these observations as follows.
Hypothesis 4: After the shock, repository coordination will become more open and decentralized.

(a) Members will increase their collaborative engagement with outside contributors.
(b) Outside contributors will take on more central roles within the work routines.
(c) Collaboration will take on a modular structure.
(d) Members will reinforce core values through automated enforcement.

4 BACKGROUND ON GITHUB AND DATA

GitHub is a popular social coding platform that provides tools for collaborative software development, social networking, and reputation, project, and access privilege management. The affordances of the platform have resulted in a large community of open source projects and developers. Due to the complexity of software development, GitHub provides several features for scaffolding of common types of work. The availability of these affordances on GitHub provides us with convenient measures of collaboration dynamics.

GitHub users can engage with a public repository in various ways. Users can make changes to a repository by submitting a *pull request*. Requests need to be approved by a member of the repository in order to be merged. Non-members can also submit and make comments on *issues*, which are used to report bugs, request changes, or to make general comments. Those interested in creating an independent copy of the repository for their use can do so by *forking*. Finally, users can signal approval of a repository by *starring*.

Some actions are restricted to only repository *members*. Repository members can do anything non-members can do in addition to adding new members, directly *pushing* changes to the code without a pull request, approving or disapproving pull requests, categorizing issues and contributions, and closing issues, among others. This allows the members to effectively steer the project but also introduces an explicit coordination bottleneck.

We rely on two different data sources: (i.) GitHub trending data, (ii.) GitHub project trace data. We use (i.) to identify the attention shocks and use (ii.) to examine how the GitHub community and the trending project team respond to the shock.

4.1 GitHub Trending Data

GitHub Trending page lists a set of repositories—updated 8 times a day—to characterize "what the GitHub community is most excited about today"[1]. One can filter the trends by language—all languages are shown by default. These languages include *C++, HTML, Java, Javascript, PhP, Python*, and *Ruby*, which are the top 7 languages that appear on the language selection drop-down list[2]. For each trending repository, website visitors see the owner/repository name, primary language name, repository description, a star button, and a list of the top five contributors for the project. According to GitHub, in order to choose the trending repositories they "look at a variety of data points including stars, forks, commits, follows, and pageviews, weighting them appropriately. It's not just about total numbers, but also how recently the events happened." [8].

In this study, we scrape the GitHub Trending page every three hours—to make sure each trending change is captured—for 7 months (6/27/18 - 1/31/19). We collect this data for all languages combined, as well as individual languages. We identify triplets (r, t, l) of repository r and time t such that r was trending at time t in language l and was not trending any time before during our scraping period [3]. To examine the projects with strong trending effects, we further limit our analysis to the projects that appear among the top 5 trending results irrespective of whether a project appeared directly or rose through the ranks. We identify 1,107 such triples [4].

4.2 GitHub Projects Trace Data

We use GitHub event stream data to examine how a repository team and the broader GitHub community respond to a repository being featured in the trending repositories list.

4.2.1 GHArchive Data. We use GHArchive archives (https://www.gharchive.org/), which provides events information about GitHub repositories, to retrieve event data between January 2018 and February 2019 (inclusive) for all repositories. This archive includes timestamped data on a large number of events including stars, pushes, pulled requests, issues, comments, member additions, among many others. There are approximately 350K repositories with at least one star or fork in the timeframe of our shocks (6/27/18 - 1/31/19).

4.2.2 REST API. While the GHArchive data is rich, it still does not include some important events that are central to our measures. As such, we complement that data by using GitHub REST API in three ways. First, we retrieve the language of each repository with at least one event within the study window. This important feature is used in our propensity score matching step (details below). Second, we collect data about 36 additional event types such as *subscribe*, *renamed*, and *labeled*, related to actions on issues and pull requests to study group dynamics. Third, we collect data on commits including author ids and status updates to identify automated tasks. Unlike GHArchive, data collection through the REST API is rather slow. Therefore, we collect this data for our shocked repositories and the set of control repositories identified using propensity score matching. In total, there are roughly 7K such repositories.

[1]https://github.com/trending
[2]Top languages list is personalized for logged in GitHub users. We use the non-personalized ordering here.
[3]In order to account for left censoring, we only consider trending repositories 14 days after the beginning of the scraping period
[4]We originally identified 1,297 triplets. However, 190 were created at most 7 days prior to when they were trending or had a very small amount of activity during that 7 day period (i.e. no stars or forks). We are unable to compute various measures for these repositories and thus we removed them from our study, resulting in 1,107 triples.

4.3 Preprocessing

We perform two preprocessing tasks that are important for ensuring the reliability of the our outcome measurements: (1) member identification and (2) removal of bot activity.

4.3.1 Member Identification. The evaluation of many of our hypotheses (2(a), 3(a), 3(b), 4(a), and 4(d)) rely on our ability to distinguish repository members from external contributors. To identify members, we use all events from GHArchive between 01/01/2018-01/31/2019. A user is labeled as a member if they perform any event that is restricted to members only. One limitation of this approach is that there could potentially be users who have been members of repositories since before 01/01/2018 but did not use any membership privileges during the period. However, these users are not exercising any member-level authority and are no different from outsiders in their observed behavior. Therefore we expect such users to have little or no effect on our outcome measurements.

4.3.2 Removal of Bot Activity. GitHub allows teams to use task automation or *bots* for tasks such as spam prevention and code quality assessment and deployment. Bots generate tremendous amounts of activity/events that are not useful to understand people's collaborative dynamics. We use the following two fold method to identify and remove bot accounts from our study. First, we filter out GitHub accounts with usernames that end with either "bot" or "[bot]". Next, to identify other bots that do not follow this standard, we perform a simple classification. We find an activity level, in terms of the average daily number of pushes when an account was active, above which an account is considered a bot. To evaluate different thresholds, we bucket accounts by number of pushes between ∞, 1000, 500, 200, 100, 50, 25, 13, 7, 4, 2, and 0. We manually label 10 randomly selected accounts from each bucket as bots or human by observing their activity. Next, we find the threshold that maximizes the macro F1 score weighted by the distribution of accounts in the complete data set. We find that the best threshold is 13 average daily pushes ($F1 = 0.78$). Based on this classification, we remove 38 bot accounts in 54 treated repositories and 45 bot accounts in 67 control repositories.

5 MEASUREMENT

In this section, we describe the construction of measures to test our hypotheses. Table 1 provides a summary of all the measures.

5.1 External Engagement

For hypothesis 1, we need to measure the engagement of external users (non-members) with the repository. We measure general interest and contributions separately. To measure external interest (H1a), we track the number of stars and forks received during a period. We use both the raw number as well as a log normalized version. For a measure M_t at time period t, the normalized version of the measure is $N_t = \log\left(\frac{M_t - M_{t-1}}{M_{t-1}}\right)$. We use this normalization for any metric that is a count or a duration. We also measure external contributions (H1b) using three measures that capture contribution at different levels of commitment; (1) Number of external contributors — users who perform any action in the repository other than starring or forking, (2) number of issues opened by external users, and (3) number of pull requests submitted by external users.

5.2 Backlogs and Productivity

For hypothesis 2, we need to measure the productivity of members to assess whether the flow of external contributors expected after the shock creates backlogs. We measure productivity in two ways.

First, we define *response delay* as the average amount of time it takes a member to respond to an issue or pull request. This measures the overall responsiveness of the members. Second, we measure *closure efficiency* as the effectiveness of the members in closing open tasks. To do so, we first define the concept of a *stale* issue or pull request. We consider a task that has remained open for 60 days as stale. We assume that stale items are probably outdated and not will not lead to further action. We then consider the amount of time available to close a task during a period as the minimum of the time remaining during the period and the time remaining before it becomes stale. Finally, we measure the repository issue (pull request) closure efficiency, as 1 minus the average fraction of time taken by members to close active issues (pull requests) and the time available to them.

5.3 Member Administrative Orientation

For hypothesis 3, we need to evaluate whether members in shocked repositories are becoming more administration oriented. We separately measure volume of administrative work (H3a) and development work (H3b). For development work, we measure (1) the total number of pushes and pull requests by members and (2) the total number of lines of code or text members added using pushes and pull requests. For administrative work, we measure the total number of issues (pull requests) closed.

5.4 Coordination Processes

For hypotheses 4, we need to measure various aspects of a project's coordination patterns and openness.

Member Engagement with Outsiders. Members engage outsiders primarily through pull requests and issues. Therefore, we measure member engagement with outsiders (H4a) using (1) the average number of issues (pull requests) edited by a member and the (2) the total number of outsider pull requests accepted during a period.

Decentralization. We quantify decentralization within projects by examining the distribution of collaborative effort between members and outsiders. We use two types of metrics to evaluate the changing role of outsiders[5]. First, in order to assess the extent of the impact that outsiders have on project's output, we measure the percentage of outsiders among the top x% users in terms of files edited.

Table 1: Measurement of GitHub project behavior used to test our hypotheses described in section 5

	Group	Measure
External Engagement (H1)	External Interest	# Stars # Forks
	External Contribution	# External Contributors # Issues Opened # PRs opened Avg. # Files Edited by Outsiders
Backlogs (H2)	Productivity	Issue Response Delay PR Response Delay Issue Closure Efficiency PR Closure Efficiency
Member Orientation (H3)	Development Work	# Pushes and PRs # Lines of code
	Admin. Work	# Issues Closed # PRs Closed
Coordination Processes and Decentralization (H4)	Member Engagement with Outsiders	Avg. # Issues Edited by Members Avg. # PRs Edited by Members Total # PRs Approved
	Decentralization	Outsiders in Top File Editors Outsider Centrality-File Net. Outsider Centrality-Issue Net. Outsider Centrality- PR Net.
	Specialization	Modularity of File Network Modularity of Issue Network Modularity of PR Network
	Automation	Binary use of Automation # Automated Tasks

Second, we measure the role of the outsiders in the collaborative process by assessing their degree centrality in three types of collaboration networks: collaboration in (1) files, (2) issues, and (3) pull requests. We construct each of the collaboration networks as follows. First, we build a bipartite network that indicates which users edited which items. In the file-user network, we add an edge between a user and a file if the user has edited the file. In issue-user and pull request-user networks, we add an edge between a user and an issue (pull request) if the user has taken any action in relation to it (e.g. open, close, comment, label, review, pin). Next, we construct the collaboration network by getting the projection of the bipartite network on the users such that the weight contributed by an item acted on by two users is inversely proportional to the total number of users that have acted on it [31]. Rather than measuring the change in the mean centrality of outsiders, which would be problematic since the size of the networks can change, we measure the fraction of outsiders that are among the top $K\%$ most central nodes. We test several values of K between 10% and 50%.

Specialization. Modularity of a network measures the degree to which it is organized into groups of nodes that interact more often with each other than with outsiders. In the file, issue, and pull request collaboration networks of a repository, we employ it is an indicator of the extent to which contributors specialize on different aspects of the project, such as contributing code for a particular feature or addressing certain types of issues.

Automation. GitHub project teams can use automation to perform a variety of routine tasks, such as running tests against submitted pull requests, edit-locking stale issues and pull requests for spam prevention, and build and deploy code. In general, these tasks

[5]Another potential way to examine decentralization is to quantify the rate with which outsiders transition to project members. However, recruitment of members is, in general, a slow process and we expect our study period may not be long enough to capture changes.

reflect the values and standards members wish to uphold in addition to being a mechanism for increasing productivity. We consider two measures in this context; (1) A binary value indicating if a project uses any task automation and (2) the unique number of such tasks.

6 METHODS

In the following sections, we detail our approach for constructing a comparison group of repositories using propensity score matching, operationalizing our measures, and testing our hypotheses using difference-in-difference.

6.1 Propensity Score Matching

We use propensity score matching (PSM) to select a comparable set of matched repositories [18, 37]. Broadly, this involves four steps; (1.) We estimate the likelihood (i.e. propensity) of a repository receiving the treatment (i.e. going trending on a given day) through a regression with an appropriate set of covariates described below. (2.) For each treated repository, we find the nearest neighbors w.r.t. a combination of the propensity and covariate similarity, (3.) We stratify the resulting set of shocks and matches by their propensity scores, and (4.) Within each strata, we estimate the overall similarity of the matches to the shocks w.r.t. covariates to ensure we have adequately accounted for selection effects. In the following sections, we describe this process in detail.

6.1.1 Covariates. While we are unaware of the exact design of the algorithm that GitHub uses to select trending repositories, we know that it is largely based on the size and the growth of external attention measured by stars and forks [8, 38]. In fact, according to the former head of GitHub's marketing team, "stars are the primary consideration for whether a repository is trending or not" [4]. Based on this we use the number of daily stars and forks for estimating the likelihood of each repository-date combination having experienced a shock. Our goal is to match each shocked repository to a control set of non-trending repository-date combinations that had a similar propensity to be shocked based on the scale and trend in attention growth in the preceding week. To do so, we condition on the following covariates. First, to account for volume, we use daily number of stars and forks a repository received over the week preceding the day of the shock. Second, to account for daily changes, we also use Log ratios of daily numbers of stars and forks of consecutive days, with the count of the more recent day as the numerator, over the week preceding the day of the shock. To be a candidate for the matched set, a repository-date combination must meet the following conditions: (i) the repository must not appear anywhere on the GitHub's trending page during our study period and (ii) the repository must receive at least one star or fork during the week prior to the date.

6.1.2 Propensity Scores. We model the likelihood of a repository trending as a logistic function of our covariates to estimate propensity scores for each repository-date in our dataset. Given the imbalanced nature of our data (1,107 positive and 27 million negative instances), we use weighted logistic regression and apply a weight to each class proportional to the class size. The resulting model clearly differentiates between shocks and non-shocks as seen by the propensity score distributions in Figure 1.

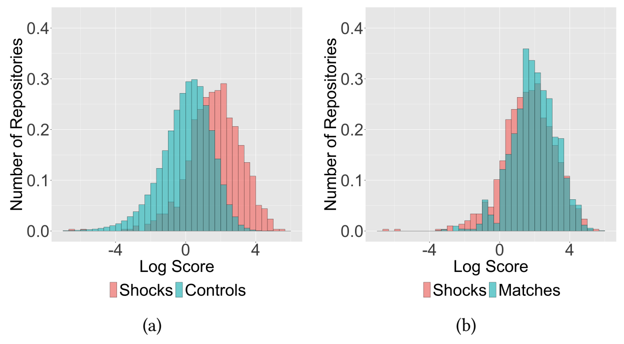


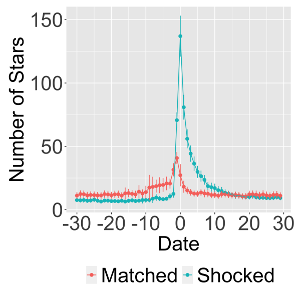
Figure 1: Propensity Score Distributions. (a). Shocks vs. All Non Shocked. (b). Shocks vs. Matches

6.1.3 Matching. We now use the estimated propensity scores to define our control set. Rather than simply using the repositories with the closest propensity score, we use a hybrid method that, for each shock, first defines a sufficiently similar neighborhood using the linear (i.e. log-odds) version of the propensity score (10% of the pooled standard deviation $\sqrt{\frac{1}{2}\left(SD^2_{shocks} + SD^2_{not\_shocks}\right)}$) and then selects the nearest neighbor repositories of the *same programming language* in terms of the Mahalanobis distance based on the covariates [37]. The version of the Mahalanobis distance we employ weighs its different components by their effect sizes from the regression. This way, we prioritize those covariates that are highly predictive of the propensity score [14]. If we do not find an adequate number of matches within propensity score neighborhood, we resort to selecting the remaining matches from the next nearest neighbors in terms of the propensity score itself [37]. Following this procedure, we select 5 matches for each of the 1,107 shocks that remain in our dataset.

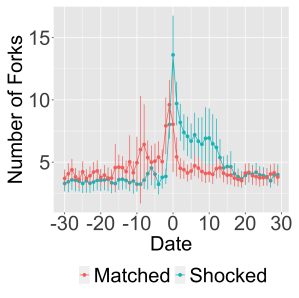
Figure 2 shows time series of the number of daily stars and forks for shocked and control repositories vs. the number of days from the day of the shock. We observe that both groups of repositories exhibit an increase in attention prior to the shock – which is expected since GitHub selects repositories with increasing visibility for their trending page, and control repositories should exhibit a similar pattern by the design of the matching procedure. However, after the shock, control repositories return to their baseline much faster than shocked ones. This is also to be expected since control repositories were not listed on the trending page and thus do not receive an additional boost in attention.

6.1.4 Stratification. Next, in order to account for any selection effects that may remain after matching, we stratify the combined shocks and matches dataset using the linear propensity score such that the behavior of shocked repositories is compared only against that of matched repositories in the same strata [18]. We apply the following procedure:

(1) Remove all matches that have a linear propensity score (PS) less than the minimum PS among the shocks (PS_{min}) (11 matches removed).



(a)



(b)

Figure 2: Number of stars (a) and forks (b) vs. number of days from the shock for control and shocked repositories.

(2) Remove all shocks that have a PS greater than the maximum PS among the matches (PS_{max}).
(3) The initial block is defined by $[PS_{min}, PS_{max}]$
(4) Repeat the following steps until there are no blocks left to split
   (a) For each block, calculate the t-statistic (**t**) for the mean difference in PS between shocked and matched repositories
   (b) If $|\ t\ | \leq 1$, treatment is sufficiently uncorrelated with PS. There is no need to further split the block
   (c) If $|\ t\ | > 1$, Split into two blocks along the median PS of the block if each new block has at least $max(3, K+2)$ observations, where K is the number of covariates.

This approach splits our data into 6 consecutive blocks with different numbers of shocks and matches (Table 2). All except the smallest block retain approximately the 1 to 5 ratio between shocks and matches. In the smallest block with 44 shocks corresponding to the highest PS scores there are only 3.7 times as many matches.

Table 2: Propensity Score Block Statistics

Block	Lower Bound	Upper Bound	Shocks	Matches	t
1	-2.56	4.88	548	2768	-0.33
2	4.88	12.58	278	1379	-0.21
3	12.58	24.60	140	688	0.11
4	24.60	39.54	65	350	0.28
5	39.54	57.64	32	176	-0.02
6	57.64	375.5	44	163	0.96

6.1.5 Balance. We verify that matches are sufficiently comparable to the shocks by evaluating the balance of the covariates using the standardized mean difference (SMD). First, we estimate the maximum value that the SMD of a covariate can have if group membership (shocked or matched) accounts for less than 1% of its variance using the formula $r = \frac{d}{\sqrt{(d^2+1/(pq))}}$, where r is the square root of the variance, d is the SMD, p is the fraction of shocks and $q = 1 - p$ [11]. Considering the 1:5 ratio between shocks and matches overall, we arrive at an upper bound for the SMD of 0.269. For each covariate, we estimate an overall SMD across all the blocks by using the weighting scheme from [18] that takes into account the number of shocked and matched repositories as well as the distribution of the covariate values in each block. For all our covariates, SMD calculated in this manner satisfy the estimated upper bound constraint. The maximum covariate SMD is 0.22 and the mean, when weighted by their corresponding effect sizes in the regression, is 0.13. This indicates that match quality is substantially better for covariates strongly associated with the propensity.

6.2 Hypothesis Evaluation

6.2.1 Difference-in-Shocked Analysis. We evaluate the first order behavioral changes in the treated repositories before and after the shock. For a behavior, y, we measure the change in a given treated repository as $\tilde{y}_t - \tilde{y}_{t-1}$, where $\tilde{y}_{t-1}$ and $\tilde{y}_t$ are the median measurements of y before and after the treatment respectively. We evaluate the statistical significance of this difference using Kruskal-Wallis one-way variance test [24]. Given that the control repositories also exhibit increased outside engagement, the observed response may be driven by a combination of organic growth and the attention shock. Thus, in order to the disentangle these effects, we also employ a diff-in-diff model.

6.2.2 Difference-in-Difference Model. We use a difference-in-difference (DD) regression approach [1] to compare the changes in shocked and matched repository behaviors to test our hypotheses described in Section 3. Difference-in-Difference is a quasi experimental approach that estimates the effect of a treatment in observational data by comparing against the outcome in the treatment set to a counterfactual derived from a suitable control set. We use this measure to characterize the change in each of the behaviors of interest (described in Section 5) 30 days before and after the shock. In order to account for propensity score based stratification in our model, we use a fixed effect for the block to which an instance belongs. This mixed effects model is summarized by the following:

$$y_{it} = \beta_0 + \beta_1 t + \beta_2 I + \beta_3 tI + \beta_4 B + \beta_x C + \epsilon$$

where y_{it} is the outcome for repository i for behavior y, t is an indicator variable for time period (0 during the period before the reference date and 1 after it), I is an indicator variable for treatment (1 if i is in the treated set or 0 otherwise), B is the PS block to which i belongs, $\beta_x C$ is a placeholder for controls and ϵ is residual error. The coefficient β_3 represents the difference in changes in outcome y_{it} over time between the treatment and the control group.

7 RESULTS

In this section, we describe the results for our hypotheses based on the *difference-in-shocked* and *difference-in-difference* estimates.

7.1 H1: External Engagement and Contribution

7.1.1 Engagement. Figure 3 shows that the number of stars and forks received by a repository increases substantially after being shocked. The difference estimates show that a shocked repository receives approximately 107% more stars and 67% forks than before on average. Also, this increase is even larger when compared with the control (237% for stars and 131% forks). This suggests that attention on shocked repositories is growing rapidly, while the attention on the control repositories is rapidly returning to baseline.

7.1.2 *Contribution.*

We also observe ≈ 50% increase in the number of users who performed at least one action other than starring and forking (Figure 3). Further, this observation remains statistically significant for higher thresholds of 2 (33%), 3 (28%), and 5 (7.5%) actions. We observe similar but smaller effects for the number of pull requests (5%) and issues opened (25%). Similar to our observations with low-effort engagement, the increase in external contributions in shocked repositories is even larger when compared with the controls which seem to be slowing down. However, when considering the number of files edited by external contributors we find that the external contributions are not scaling at the same rate. In fact, we see that on average an outside contributor edits 37% fewer files and that this effect persists, to a lesser extent (10%), when compared with the control set. This provides a more nuanced perspective of increasing outsider contribution, where most contributions from the expanded outsider community are relatively shallow. Taken together, our results support hypotheses 1(a) and partially support 1(b) – there is a large increase in engagement, in terms of software interest, but also in willingness to contribute, albeit shallow ones.

7.2 H2: Backlogs

With external engagement increasing, we expect members of shocked repositories to be overwhelmed by the growing number of tasks that need their attention. While, some evidence of this emerges, our observations depict a more complex portrait of the situation (Figure 4). After the shock, the amount of time it takes a member to respond to a new issue or a pull request increases substantially, by 30% and 42% respectively. This suggests that members are indeed feeling the strain of an increased workload. This decrease in responsiveness is even larger compared against control repositories where external attention is declining when compared to the reference date.

However, we do not see conclusive evidence suggesting that members in shocked repositories are becoming less efficient in actually closing pull requests or issues. While the efficiency of dealing with pull requests seems to decrease slightly (≈ 4%), this slow down is no different from the one observed in control repositories, suggesting that this slowdown is a consequence of increased organic attention, but not of the shock itself. In addition, issue closure efficiency does not show any change in shocked repositories either with or without comparison to the controls.

Overall, members of shocked repositories are slower with regards to their initial response towards external engagement. However, they are still managing to keep up with the workload by following through on tasks. In summary, our results partially support hypothesis 2.

7.3 H3: Member Administrative Orientation

How do the members of shocked repositories compensate for the increasing demands on their time from outsiders? Figure 5 shows that, as we hypothesized, members of shocked repositories become more administration oriented.

7.3.1 *Admin Work.*

We observe that members are trying to keep up with the increased administrative overhead of engaging with outsiders. The number of issues and pull requests they close increase substantially after the shock, with the number of issue closures increasing by 33% and pull request closures increasing by 11% after controlling for the number active items (issues or pull requests) during each period.

7.3.2 *Development Work.*

Members of shocked repositories are doing substantially less development work in the aftermath of the shock. On average, they submit 38% fewer pushes and pull requests in the period after the shock. They fall behind members of control repositories even more (42%). We also inspect the lines of code (or text) changed by members as this metric can be more illuminating in this context. Again, we observe that members in shocked repositories reduce the number of lines they change—albeit by a smaller percentage of 18%. This decline is much more substantial (57%) when compared to the control set.

Overall, we find that members in shocked repositories forego development work to coordinate external engagement and contributions, providing support for hypotheses 3(a) and 3(b).

7.4 H4: Coordination Processes and Decentralization

Based on our hypotheses, we expect that GitHub projects that experience an attention shock will become more open and decentralized. In general, our results largely align with the expectation and provide partial support for hypothesis 4 (see figure 6). However, there are several important caveats and robustness tests discussed below.

7.4.1 *Member Engagement with Outsiders.*

The results show that members are trying to accommodate the greater volume of incoming contributions. The number of pull requests approved increases by a 12% for shocked repositories. However, this increase is no different from that of the control set. Therefore, this increasing openness appears to be a function of the organic growth of a repository rather than the effect of an attention shock. We also do not see a significant change in the number of issues edited by members among shocked repositories. However, in comparison to the control set, the story changes — there is indeed a large increase (30%) in the number of edited issues by members in shocked repositories relative to control. This suggests that shocked repositories are maintaining their engagement at higher levels than the control set. In terms of engagement on external pull requests, we observe that individual members in shocked projects edit moderately fewer pull requests (17%) than before, but, in fact, they are editing more, almost 44%, relative to the control set. We find support for hypothesis 4(a), but the effect is due to both natural growth and the shock; and the specific type of engagement (issues or pull requests).

7.4.2 *Decentralization.*

We find evidence for hypothesis 4(b). Members of shocked repositories do indeed allow outsiders to take on more central positions in work, thereby adopting a more decentralized brand of coordination. However, there is one important caveat. First, we observe that, among shocked repositories, the fraction of outsiders among the top 50% contributors with most files edited increases by approximately 8%. This increase remains roughly the same in magnitude when compared with control repositories (7.2%), indicating that, in the aftermath of the shock, outsiders are allowed to make more substantive contributions than they would have in

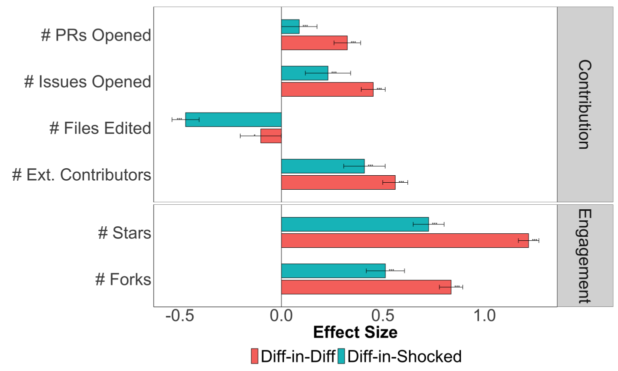


Figure 3: External Engagement (H1). The bars show the effect sizes for the Diff-in-Diff regression, which represents the effect of the shock on each measure relative to the control group as a percentage, as well as the difference in medians before and after the shock for shocked repositories only. The error bars show 95% confidence intervals. The stars show significance (* p-val < 0.001, ** p-val < 0.01, * p-val < 0.05). Other bar plots in this section follow the same format.**

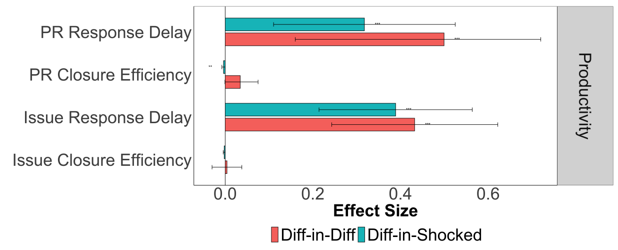


Figure 4: Backlogs (H2)

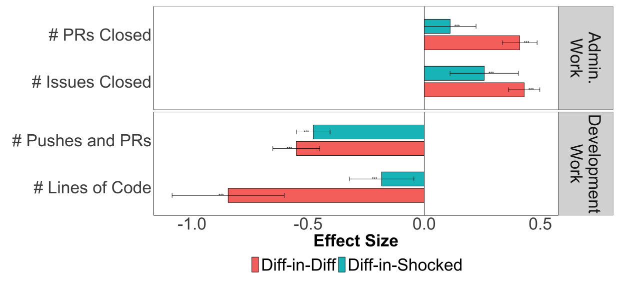


Figure 5: Member Admin. Orientation (H3)

other times of increased attention. Robustness tests show that this phenomenon persists at a higher threshold of 40% but not beyond. However, in contrast to this observation, we observe that the fraction of outsiders among the top 50% most central contributors with highest degree centrality in the file collaboration network does not change in the shocked repositories with or without comparing to the controls. Therefore, it appears that, while outsiders are being allowed to make greater contributions, these contributions may be relatively isolated from core parts of the project, where development is likely to be more collaborative.

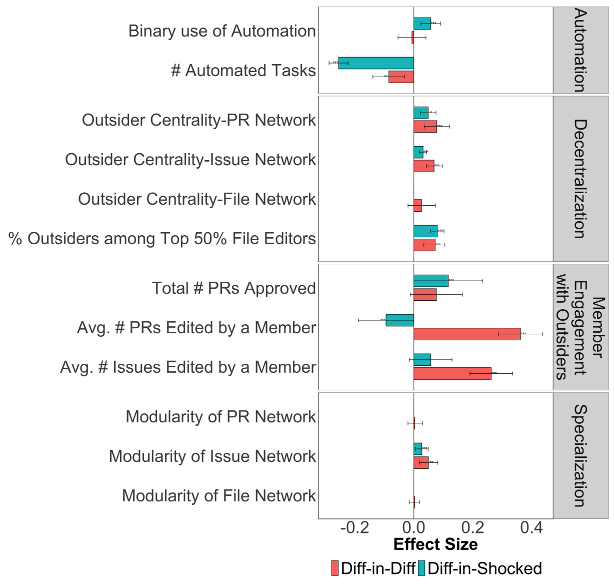


Figure 6: Coordination Processes and Decentralization (H4). Missing bars indicate that all values for the corresponding measurement are zero.

Less experience and effort is required to contribute to discussions surrounding issues and pull requests than to contribute code. Therefore, we expect that it would be easier for outsiders to take on more central roles in the collaboration related to issue and pull request discussions. Our results for the fraction of outsiders among the top 50% contributors with highest degree centrality for issue and pull request collaboration networks show that this is indeed the case. We observe that the number of outsiders among the most central contributors increases by 3% in the issue collaboration network and by 5% in the pull request network. In both cases, this effect is amplified when compared against the controls (7% and 8% among issue and pull requests networks, respectively), indicating the increased external attention drives decentralization. Additional robustness tests show that the increasing importance of outsiders in the issues collaboration networks is consistent even at higher thresholds of top 40%, 30%, 20%, and 10%, while it extends only up to the top 40% in the pull request network. Thus the centrality of outsiders is more persistent with issue discussions which require less effort and expertise than pull request discussions. This serves as further evidence that outsiders work within a hierarchy of commitment and experience as they become more central.

7.4.3 Modularity of Collaboration. Our results provide partial support for hypothesis 4(c), which predicts increase in specialization by members and outsiders to deal with coordination costs. First, in the file and pull request networks, which are associated with contributors with relatively high levels of commitment, we observe no effect on the modularity of collaboration in the shock set, with or without a comparison to the control set. However, in the issue collaboration network, which is associated with relatively low effort, we do see a small increase in modularity of 7.5%, which remains true when compared with the controls (5%), indicating that some level of specialization does occur. Why does file and pull request collaboration, which stands to benefit perhaps even more than issues, not show any effect? We argue that this may be due to the relatively high complexity of these contributions. Code contributions and discussions about pull requests require more effort and expertise, which take more time to build up compared to issue discussions, which often serve as landing site of general queries.

7.4.4 Automation. Regarding automation, our results provide only partial support for hypothesis 4(d). Shocked projects become slightly more likely to adopt automation if they haven't already (6%). However, this increased likelihood is no different in the control set. This suggests that the shocked projects that adopted automation, would have probably adopted automation anyway as part of their natural growth. Further, contrary our expectation, the number of automated tasks in shocked repositories decreases by 20% in the aftermath and this remains true to a lesser extent even when compared to the controls (8%). What explains this pattern? Organic growth that leads to a shock is sometimes associated with a concerted drive in development to produce a software release. The number of automated tasks right before a release (and the shock) could be substantially higher than at other times. This would inflate automation in the *before* period. Future work should determine if this potential mechanism is indeed at play.

8 LIMITATIONS

Our paper has several limitations. First, we only consider the short-term impact of the shock. Whether the shock has a long lasting impact remains as an open question. Second, our analysis is based on logged user actions such as committing and submitting pull requests, but ignores the text in comments or measures of code quality in pull requests. A more detailed analysis of this content could uncover new dynamics that we do not capture here. Third, we test our hypotheses independently of each other. While this approach is sufficient to characterize the effects of the attention shock on the collaboration dynamics, it is possible that there are dependencies between our collaboration measures. For example, the level of modularity in the issue collaboration network may impact the time to address issues. Future work can examine interactions between our measures through the use of structural equation models. Fourth, we do not consider the impact of repeated shocks on a repository. Additionally, since we do not have access to trending data prior to our 7 month period, we cannot be sure that the repositories that were trending during our period were not trending prior to it. Fifth, we ignore aspects such as the application and quality of repositories when selecting our controls. This could affect the quality of matching. However, because we selected controls based on GitHub's statements about trending repositories and covariate balance was within acceptable limits based on standard metrics, we deem our control dataset to be adequately similar to the trending projects. Finally, we characterize the response by repositories to attention shocks but do not address whether the observed responses are beneficial or harmful. Future work should uncover which types of responses lead to desirable outcomes such as higher quality and increase in adoption.

9 DISCUSSION

Our research expands the current understanding of open source software crowds by studying how they respond to attention shocks. Trending GitHub crowds actually respond or adapt similarly to successful organizations despite their differences. Based on the analysis using the distributed leadership framework, core members attempt to adapt their coordination style by distributing work responsibility. For, example, they allow outsiders to take more central yet unofficial positions. They also increasingly adopt automation as a means of enforcing community norms and work practices. The project also self-organizes into a modular structure while core members restructure their work. Despite this apparent positive response by core members there are some signals that they are being overwhelmed, at least in the short term. Due to the increase in contributions from outsiders, core members strain to coordinate work and begin to accumulate backlogs, which lead to delays in responding outsiders. This study provides implications for GitHub crowds responding to attention shocks. First, most of the engagement from arriving outside contributors are shallow. This has the effect of increasing the work of core members, making it potentially difficult to leverage the efforts of these arriving outside contributors. This begs the question: How can core members make the most of such increased resources? One approach is to make easily doable small tasks highly visible through public to-do lists that outsiders can work on. This can help outsiders easily identify urgent needs of the project and make it more likely that their contributions will be accepted.

Second, despite the best efforts of core members, a backlog of pull requests and issues pile up, and while they eventually close them, there are long delays in responding to them. This low responsiveness has the potential of dissuading the contributions of outsiders, including those who could have become productive contributors. How can crowds being impacted by attention shocks prevent low responsiveness from running off arriving outsiders? One possibility is to automate responses to pull requests and issues using boilerplate responses (e.g. Thanks! Will get back to you) [29]. An automated response might give outsider the sense that members are aware of their contributions and will eventually review them.

Finally, new work groups emerge and arriving outsiders become increasingly central. This may explain why members are able to keep up with closing issues and pull requests. However, this is more likely to be a spontaneous self-organized response rather than a planned strategy. An open question is how GitHub should manage the expectations of members of repositories that are listed on the trending page. Our findings suggest that this can create both opportunities for more attention and contributors, but also a potential disruption to their routines. Providing members with a warning of what is likely to come their way could help them be more prepared to herd the deluge of good Samaritans.

ACKNOWLEDGMENTS

This research was partly supported by the National Science Foundation under Grant No. IIS-1617820. We thank Jiaxin Ye for his assistance with data collection. We also thank Steve Oney and Christopher Quarles for helpful discussions and feedback.

REFERENCES

[1] Alberto Abadie. 2005. Semiparametric difference-in-differences estimators. *The Review of Economic Studies* 72, 1 (2005), 1–19.

[2] Ofer Arazy and Oded Nov. 2010. Determinants of Wikipedia Quality: The Roles of Global and Local Contribution Inequality. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work* (Savannah, Georgia, USA) *(CSCW '10)*. ACM, New York, NY, USA, 233–236.

[3] Douglas D Baker and John B Cullen. 1993. Administrative reorganization and configurational context: The contingent effects of age, size, and change in size. *Academy of Management Journal* 36, 6 (1993), 1251–1277.

[4] Andrew Begel, Jan Bosch, and Margaret-Anne Storey. 2013. Social networking meets software development: Perspectives from github, msdn, stack exchange, and topcoder. *IEEE Software* 30, 1 (2013), 52–66.

[5] Yochai Benkler. 2006. *The wealth of networks: How social production transforms markets and freedom.* Yale University Press.

[6] Yochai Benkler, Aaron Shaw, and Benjamin Mako Hill. 2015. Peer production: A form of collective intelligence. *Handbook of collective intelligence* 175 (2015).

[7] Peter M Blau. 1970. A formal theory of differentiation in organizations. *American sociological review* (1970), 201–218.

[8] The GitHub Blog. 2013. Explore what is Trending on GitHub. https://github.blog/2013-08-13-explore-what-is-trending-on-github/. Accessed: 2019-10-05.

[9] Casey Casalnuovo, Bogdan Vasilescu, Premkumar Devanbu, and Vladimir Filkov. 2015. Developer onboarding in GitHub: the role of prior social links and language experience. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering.* ACM, 817–828.

[10] John Child and Alfred Kieser. 1981. Development of organizations over time. *Handbook of organizational design* 1 (1981), 28–64.

[11] Jacob Cohen. 1988. *Statistical power analysis for the behavioral sciences* (2nd ed. ed.). L. Erlbaum Associates.

[12] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work.* ACM, 1277–1286.

[13] Charles J Fombrun and Stefan Wally. 1989. Structuring small firms for rapid growth. *Journal of Business Venturing* 4, 2 (1989), 107–122.

[14] Robert A Greevy Jr, Carlos G Grijalva, Christianne L Roumie, Cole Beck, Adriana M Hung, Harvey J Murff, Xulei Liu, and Marie R Griffin. 2012. Reweighted Mahalanobis distance matching for cluster-randomized trials with missing data. *Pharmacoepidemiology and drug safety* 21 (2012), 148–154.

[15] Peter Gronn. 2002. Distributed leadership as a unit of analysis. *The leadership quarterly* 13, 4 (2002), 423–451.

[16] Stefanie Habersang, Jill Küberling-Jost, Markus Reihlen, and Christoph Seckler. 2019. A Process Perspective on Organizational Failure: A Qualitative Meta-Analysis. *Journal of Management Studies* 56, 1 (2019), 19–56.

[17] Donald C Hambrick and Lynn M Crozier. 1985. Stumblers and stars in the management of rapid growth. *Journal of business venturing* 1, 1 (1985), 31–45.

[18] Guido W Imbens and Donald B Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences.* Cambridge University Press.

[19] Jing Jiang, David Lo, Jiahuan He, Xin Xia, Pavneet Singh Kochhar, and Li Zhang. 2017. Why and how developers fork what from whom in GitHub. *Empirical Software Engineering* 22, 1 (2017), 547–578.

[20] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M German, and Daniela Damian. 2014. The promises and perils of mining GitHub. In *Proceedings of the 11th working conference on mining software repositories.* ACM, 92–101.

[21] Brian Keegan, Darren Gergle, and Noshir Contractor. 2013. Hot off the wiki: Structures and dynamics of Wikipedia's coverage of breaking news events. *American Behavioral Scientist* 57, 5 (2013), 595–622.

[22] Aniket Kittur and Robert E. Kraut. 2008. Harnessing the Wisdom of Crowds in Wikipedia: Quality Through Coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work* (San Diego, CA, USA) *(CSCW '08)*. ACM, New York, NY, USA, 37–46.

[23] Niklas Kiviluoto. 2013. Growth as evidence of firm success: myth or reality? *Entrepreneurship & Regional Development* 25, 7-8 (2013), 569–586.

[24] William H Kruskal and W Allen Wallis. 1952. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association* 47, 260 (1952), 583–621.

[25] David Laniado and Riccardo Tasso. 2011. Co-authorship 2.0: Patterns of collaboration in Wikipedia. In *Proceedings of the 22nd ACM conference on Hypertext and hypermedia.* ACM, 201–210.

[26] Antonio Lima, Luca Rossi, and Mirco Musolesi. 2014. Coding together at scale: GitHub as a collaborative social network. In *Eighth International AAAI Conference on Weblogs and Social Media.*

[27] Stephan Liozu, Andreas Hinterhuber, and Toni Somers. 2014. Organizational design and pricing capabilities for superior firm performance. *Management Decision* 52, 1 (2014), 54–78.

[28] Jennifer Marlow, Laura Dabbish, and Jim Herbsleb. 2013. Impression formation in online peer production: activity traces and personal profiles in github. In *Proceedings of the 2013 conference on Computer supported cooperative work.* ACM, 117–128.

[29] Nora McDonald, Kelly Blincoe, EVA Petakovic, and Sean Goggins. 2014. Modeling distributed collaboration on Github. *Advances in Complex Systems* 17, 07n08 (2014), 1450024.

[30] Danny Miller. 1994. What happens after success: The perils of excellence. *Journal of Management Studies* 31, 3 (1994), 325–358.

[31] Mark EJ Newman. 2001. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical review E* 64, 1 (2001), 016132.

[32] Charlene L Nicholls-Nixon. 2005. Rapid growth and high performance: The entrepreneur's "impossible dream?". *Academy of Management Perspectives* 19, 1 (2005), 77–89.

[33] Felipe Ortega, Jesus M Gonzalez-Barahona, and Gregorio Robles. 2008. On the inequality of contributions to Wikipedia. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008).* IEEE, 304–304.

[34] E Platt and DM Romero. 2018. Network Structure, Efficiency, and Performance in WikiProjects. In *Proceedings of International AAAI Conference on Web and Social Media (ICWSM).*

[35] Eric Raymond. 1999. The cathedral and the bazaar. *Knowledge, Technology & Policy* 12, 3 (1999), 23–49.

[36] Daniel M Romero, Dan Huttenlocher, and Jon Kleinberg. 2015. Coordination and efficiency in decentralized collaboration. In *Ninth International AAAI Conference on Web and Social Media.*

[37] Paul R Rosenbaum and Donald B Rubin. 1985. Constructing a control group using multivariate matched sampling methods that incorporate the propensity score. *The American Statistician* 39, 1 (1985), 33–38.

[38] TechCrunch. 2013. GitHub Adds Trending Page To Filter By Project, Programming Languages And Developers. http://social.techcrunch.com/2013/08/13/github-adds-trending-page-to-filter-by-project-programming-languages-and-developers/ Accessed: 2019-10-05.

[39] Bogdan Vasilescu, Kelly Blincoe, Qi Xuan, Casey Casalnuovo, Daniela Damian, Premkumar Devanbu, and Vladimir Filkov. 2016. The sky is not the limit: multitasking across GitHub projects. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE).* IEEE, 994–1005.

[40] Bogdan Vasilescu, Vladimir Filkov, and Alexander Serebrenik. 2015. Perceptions of diversity on GitHub: A user survey. In *Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering.* IEEE Press, 50–56.

[41] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark GJ van den Brand, Alexander Serebrenik, Premkumar Devanbu, and Vladimir Filkov. 2015. Gender and tenure diversity in GitHub teams. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems.* ACM, 3789–3798.

[42] Bogdan Vasilescu, Yue Yu, Huaimin Wang, Premkumar Devanbu, and Vladimir Filkov. 2015. Quality and productivity outcomes relating to continuous integration in GitHub. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering.* ACM, 805–816.

[43] David A Whetten. 1987. Organizational growth and decline processes. *Annual review of sociology* 13, 1 (1987), 335–358.

[44] Ark Fangzhou Zhang, Danielle Livneh, Ceren Budak, Lionel Robert, and Daniel Romero. 2017. Shocking the crowd: The effect of censorship shocks on chinese wikipedia. In *Eleventh International AAAI Conference on Web and Social Media.*

[45] Ark Fangzhou Zhang, Danielle Livneh, Ceren Budak, Lionel P. Robert, Jr., and Daniel M. Romero. 2017. Crowd Development: The Interplay Between Crowd Evaluation and Collaborative Dynamics in Wikipedia. *Proc. ACM Hum.-Comput. Interact.* 1, CSCW, Article 119 (Dec. 2017), 21 pages.

[46] Ark Fangzhou Zhang, Ruihan Wang, Eric Blohm, Ceren Budak, Lionel P Robert Jr, and Daniel M Romero. 2019. Participation of New Editors after Times of Shock on Wikipedia. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 13. 560–571.