Participation Inequality and the 90-9-1 Principle in Open Source

Mattia Gasparini  
Politecnico di Milano  
Milano, Italy  
mattia.gasparini@polimi.it

Marco Brambilla  
Politecnico di Milano  
Milano, Italy  
marco.brambilla@polimi.it

Robert Clarisó  
Universitat Oberta de Catalunya  
Barcelona, Spain  
rclariso@uoc.edu

Jordi Cabot  
ICREA  
Barcelona, Spain  
jordi.cabot@icrea.cat

ABSTRACT
Participation inequality is a major challenge in any shared-resource system. This is known as the "volunteer’s dilemma": everybody wants to benefit from a resource without contributing, expecting others will do the work. This paper explores whether this problem also arises in open source development. In particular, we analyze the behavior of GitHub users to assess whether the 90-9-1 principle applies to open source. We study it both from a qualitative (ratio of activity types) and a quantitative (total number of activities) perspective and we show that the principle does not hold if we consider the GitHub platform as a whole. Surprisingly, results are reversed depending on the specific projects we look at. We believe these results are useful to project managers to better understand and optimize the behavior of the community around their projects and, as a side effect, they show the importance of diversity in sample selection.

CCS CONCEPTS
• Software and its engineering → Open source model; Software configuration management and version control systems; • Human-centered computing → Open source software; Empirical studies in collaborative and social computing; Social network analysis.

KEYWORDS
open source software, GitHub, social network analysis, user behavior

1 INTRODUCTION
GitHub is the most popular service to develop and maintain open source software. In this service, each user interacts with many others during the software development process: sending a pull request, opening an issue, committing code, … These activities can be studied to better understand the dynamics that drive an open source software project, to provide insights about the developers’ behavior and to define organizational structures that improve a project’s odds of success. To this end, GitHub can be studied as a social network, as it exhibits many different social behaviors.

In particular, in this work we study whether participation inequality holds in open-source software development platforms. Participation inequality is a well-known behavior [28] in many shared-resource systems. It is related to the "tragedy of the commons" [16, 18] scenario, where a limited resource is shared by several members of a community. In this setting, selfish members may acquire large quantities of the shared resource, depleting it and leading to a worse outcome for the community as a whole.

Nevertheless, an open source project does not exactly fit the "tragedy of the commons": the project is a shared resource, but using the project does not consume or deplete it as a resource [20]. In fact, open source projects benefit from having a large community of users in a variety of ways [7, 32]:

• Users can provide useful feedback that can improve the quality of the project, e.g. by detecting bugs or suggesting relevant features.
• Users are more likely to become contributors, as they are already familiar with the project and benefit directly from any improvement to the project.
• Popular projects are more likely to attract new contributions and resources (e.g., donations), improving the long term viability of the project.

In general, the sustainability of open source projects is an unsolved problem [14], as significant effort is required to manage, maintain and extend an existing project over a period of time. Having a large pool of users can also contribute negatively to the sustainability of an open source project.

On one hand, a large number of non-contributing users may increase software maintenance costs, e.g., the effort required from developers to manage the project’s backlog. For instance, in [4] a sample of 3426 bug reports from the Eclipse open source project included 36% that were marked as "duplicate", "invalid", "cannot be duplicated" or "will not be fixed".
On the other hand, open source is a particular case of the volunteer’s dilemma [13, 20]: participants can benefit from other’s contributions even if they do not contribute themselves, but if nobody cooperates the community loses as a whole. In this setting, a large pool of free-riding users may lead to the social psychological phenomenon known as diffusion of responsibility or bystander effect [36]: a lack of participation due to individuals assuming that “somebody else will do it”. In particular, the strength of this phenomenon has been shown to increase with the number of people involved.

A recent example of this situation in the open source community appeared in the OpenSSL project\(^1\). OpenSSL is a widely used library to encrypt traffic for many websites and services. In 2014, a severe vulnerability called Heartbleed was discovered in OpenSSL, endangering the security and privacy of all users of the library. The impact of the bug revealed the lack of resources available to OpenSSL\(^2\), despite being a piece of critical infrastructure used by many individuals, public and private organizations.

Thus, while having a large number of users in an open source project is important, it is necessary to keep track of how they relate to active developers. When analysing the community around an open source project, we must take into account that online structures often exhibit power law distributions. For example, considering the number of inbound links to a website, a few popular websites tend to concentrate the majority of the links [19]. In particular, regarding online content creation, studies have observed that 90% of users are lurkers who never contribute to generate content, 9% of users provide only minor contributions, and 1% of users, referred to as superusers, account for almost all the action [28]. For this reason, this empirical statement is also known as the 90-9-1 principle.

We want to discover if this observation also applies to open source, and in particular to the GitHub platform, which is our object of study. That is, we consider whether most developers are actually not producing anything while few users generate most of the content, or if instead different behaviors arise. As a proxy for distinguishing between passive and active developers, we measure the quality of their contributions even if they do not contribute themselves, but if nobody cooperates the community loses as a whole. In this setting, a large pool of free-riding users may lead to the social psychological phenomenon known as diffusion of responsibility or bystander effect [36]: a lack of participation due to individuals assuming that “somebody else will do it”. In particular, the strength of this phenomenon has been shown to increase with the number of people involved.

A recent example of this situation in the open source community appeared in the OpenSSL project\(^1\). OpenSSL is a widely used library to encrypt traffic for many websites and services. In 2014, a severe vulnerability called Heartbleed was discovered in OpenSSL, endangering the security and privacy of all users of the library. The impact of the bug revealed the lack of resources available to OpenSSL\(^2\), despite being a piece of critical infrastructure used by many individuals, public and private organizations.

Thus, while having a large number of users in an open source project is important, it is necessary to keep track of how they relate to active developers. When analysing the community around an open source project, we must take into account that online structures often exhibit power law distributions. For example, considering the number of inbound links to a website, a few popular websites tend to concentrate the majority of the links [19]. In particular, regarding online content creation, studies have observed that 90% of users are lurkers who never contribute to generate content, 9% of users provide only minor contributions, and 1% of users, referred to as superusers, account for almost all the action [28]. For this reason, this empirical statement is also known as the 90-9-1 principle.

We want to discover if this observation also applies to open source, and in particular to the GitHub platform, which is our object of study. That is, we consider whether most developers are actually not producing anything while few users generate most of the content, or if instead different behaviors arise. As a proxy for distinguishing between passive and active developers, we measure the volume (number) and the value (quality) of their contributions in terms of productivity and actual relevance to the project. For the evaluation, we will consider Zipf’s law as our proxy to understand the presence, or absence, of participation inequality [5]. This law was first formulated for corpora in natural language, showing that the frequency of any word is inversely proportional to its rank in the frequency table [15], and then it was extended to several studies in physical and social sciences [6, 8, 27]. Also, variations to this law were studied, such as Lotka’s law [29] and Price’s law.

The paper is structured as follows: Section 2 presents the related work. Section 3 presents how the analysis has been designed. Section 4 describes the output of the analysis, while Section 5 discusses the results. Then, Section 6 considers the threats to validity, and Section 7 draws the conclusions and proposes future work.

\(^{1}\)https://www.openssl.org/

\(^{2}\)http://veridicalsystems.com/blog/of-money-responsibility-and-pride/

2 RELATED WORK

Participation inequality has been studied in several types of social networks, with some differences regarding the conclusions. For instance, some works focus on Digital Health social networks: in [5, 34], the authors verify the presence and consistency of the distribution and argue that superusers (1%) should be attracted to the platform, as they are the key for the long-term success of a platform. Also, classical social networks attracted this kind of analysis, as in [2], where they describe how the 90-9-1 distribution in Twitter exists but it is less pronounced. Wikipedia is another target platform for studying such behavior: among all, [31] includes temporal variables and several metrics to study the distributions, arguing that the same aggregated inequality values follow several different dynamics if these values are calculated monthly.

As for OSS platforms, we can distinguish two types of studies. First, some works consider non-code contributions to a project [11], such as e-mail messages in the project mailing list, bug reports, requirement requests, … On the other hand, other papers focus on code contributions, such as commits to a software repository.

In the first group, [23] studies the Apache web server support system. [25] verifies the inequality principle in requirements definition using bioinformatics communities as target, and showing that core developers provide most of the requirements. [22] studies the discussion patterns in the KDE developer mailing list. Similarly, [12] monitors the discussion in the developer forums devoted to porting the Debian Linux distribution to 9 different processor architectures. Finally, [33] studies e-mail discussions in five open source projects (Python, Gaim, Slashcode, PCGEN and TCL).

In the second group, [26] applies Lotka’s law to understand the developers’ contribution on SourceForge and Linux Software Map. Several studies have considered the applicability of the Pareto principle (also called 80/20 rule) to open source communities. This principle observes that a majority of contributions (80%) tend to be produced by a small subset of the developers (20%), known as the core team. For example, [17] studies 3 open source projects, and finds that the number commits, mails in the project mailing list and bug reports all follow the Pareto principle. Then, [35] evaluates 2,496 projects on GitHub and finds that between 40 and 87% of them do not follow the Pareto principle. In contrast, [1] analyzes 661 open source projects and concludes that 77% of them have “hero” developers: a subset of less than 20% of the users who contribute 80% or more of the commits.

This notion of “core team” has been further explored in many other references, e.g., [30]. A typical metric used to evaluate the core team is the bus factor [9, 35], the number of key developers that would hamper the progress of the project if they left (or were literally hit by a bus). This analysis can be very valuable from a risk-assessment perspective, but does not reveal additional insights about inequality in developer contributions.

In this paper, we study participation inequality in open source software considering source-code contributions in GitHub. Our analysis goes beyond the Pareto principle and considers whether the 90-9-1 principle applies to developer contributions. These contributions are not only quantified (number of commits) as in most previous works, but also classified depending on the type of action, i.e. a fork of a repository is not treated in the same way as a commit.
Furthermore, this analysis is performed in a large population of projects of different types ("popular" and "random") and also considers the behavior of randomly selected users, outside the scope of a specific project. For these reasons, we think this study provides a more detailed picture of participation inequality in open source software development.

### 3 METHODOLOGY

#### 3.1 Types of participation in a project

In software development, participation can be defined and measured in many different ways. On the GitHub platform, participation is mainly related to the activities performed by each developer on software repositories, either owned by her or by fellow developers. User activities are tracked by GitHub and grouped into several events, which are accessible from the official API. These events range from adding comments, starring a project, forking it, opening issues,... to committing code. As each type of event requires a different degree of commitment from the user, we manually inspected the description and the semantics of each event and we classified them into three categories:

- **Watch** includes passive events, actions that require little effort from the user and do not have an impact on the repository.
- **Interact** contains weakly active events, requiring minor actions and with a small impact on the repository.
- **Contribute** is associated with active events, requiring effort or expertise and with a major impact on the repository.

There is some degree of subjectivity in this taxonomy, as some events could be considered either passive or active depending on the interpretation. In particular, ForkEvent is the most complex to classify: on one hand, it can be considered as a "watch" event since it does not provide any improvement to the project directly, while on the other hand it replicates a complete repository, thus generating content on the platform. This could suggest considering a new intermediate level of interaction between "watch" and "interact". Nevertheless, in order to facilitate the analysis of the 90-9-1 principle we decided to preserve a taxonomy with three levels and classify the ForkEvent as a "watch" event. Table 1 summarizes the list of available GitHub event types, their semantics and their categorization.

Given this categorization, it is clear that a user can perform events belonging to more than one category. In order to classify users into a single category (Watchers, Interactive, Active) we set as basic criteria that a user belongs to the most active category for which it has an activity, e.g.: if it has at least one activity of the category *contribute*, then he/she is a strongly active user. For this preliminary analysis, we do not consider any threshold on the total volume of activities.

#### 3.2 Research questions

We can formalize the objectives of our study by defining the following research questions:

- **RQ1:** Do open source software developers exhibit participation inequality in their contributions?
- **RQ2:** Do open source software projects suffer from participation inequality?
- **RQ3:** Does the popularity of an open source software project affect participation inequality?

In a platform like GitHub, a user can participate in several projects. With this research questions we aim to discover whether the 90-9-1 principle holds for the GitHub platform as a whole (i.e., most users are watchers with few contributors) or if this only happens in specific projects. In particular, we are interested in the impact of project popularity: projects with many "stars" (GitHub’s notion of favorite projects) attract larger number of developers, and may therefore exhibit a more pronounced trend with respect to participation inequality.

For this reason, answering RQ1 will require us to study a population of *GitHub users*, considering their activity across different projects; while RQ2 and RQ3 will require us to study a population of *GitHub projects*, selected randomly in the case of RQ2 or by popularity in the case of RQ3. We believe that analyzing participation

---

inequality at the platform and project levels is relevant, as this dimension is not typically considered in other social networks.

### 3.3 Data collection

In order to verify the distribution with the presented approach, we collected a log of all the events (pull requests, issues, comments, …) that occurred on GitHub in 2018 from GHArchive⁴, consisting in 478M activities contained in zipped JSON archives. The collection has been performed using open endpoints that allow to compress in a single file all the events from a target day, month or hour using a specific syntax for the endpoint itself. For example, wget http://data.gharchive.org/2018-01-01..31-0..23.json.gz is the command to download an archive that includes all the events from GHArchive that occurred during the month of January 2018.

A second step is needed in order to decompress the archive and extract the information: tuples (user_id, repository_id, event_type, event_timestamp) for a set of target projects are saved in CSV files, which are the input for the analysis.

### 4 RESULTS

Considering our research questions, three different datasets were sampled:

- **Random Users**: Activities related to any project in GitHub performed by 10K randomly selected users during 2018, for a total number of 1.1M events.
- **Random Projects**: Activities performed during 2018 related to 500 random projects, which involved 2803 users and 24795 events.
- **Top Projects**: Activities related to the 500 most starred projects of 2018, performed in the same year by 736650 users, covering 3M events.

To measure distribution of users’ activities with respect to their quality, we proceed as follows:

1. Calculate the normalized number of activities in each category for each user;
2. Sort users by contribution and interaction percentage in decreasing order, i.e. most active first.

Charts in Figure 1 show the distribution of GitHub developers for the three sample datasets. Each vertical line in the chart depicts the activity of a single user, showing the proportion of activities of each type where she was involved (dark red: contribute; red: interact; light red: watch). For instance, a vertical line in dark red means a user that only contributes. Vertical lines with different shades of red depict users that participated in different types of events. Users are ordered from left (most active) to right (least active), starting from users that only contribute, then those that contribute and interact, then those that do not contribute but interact, and finally those that only watch. Visually, darker areas in the chart correspond to more active groups of users.

In order to measure whether this user behavior follows the 90-9-1 principle, we need to classify users in three categories from least to most active. To this end, using the definition established in Section 3, developers are associated with the most active category for which they have an activity. The percentage of users for each assigned category in each of the datasets is presented in Table 2.

As a remark, we need to underline that users with few activities could be considered as active users, because they have higher ratios: for instance, if a user has 1 contribution and 1 interaction, then it has ratio 1/2. This situation happens often, and it is illustrated by the “dents” in the charts: these are developers with few activities and, by consequence, sharply defined ratios like 1/2, 1/3, 2/3, … This observation shows that we need to consider the volume of activities, too, if we want to have a better picture of the phenomenon.

We calculate the log-log distribution for the total number of activities to analyze the overall volumes: resulting distributions are presented in Figure 2. The blue line is the empirical counting: each dot represents the total number of user’s activities in logarithmic scale. The black dashed line is the theoretical power law curve that follows Zipf’s Law. In this case, the overall activities of the users are shown, without distinction on the categorization: the most active developers set the head of the distribution with thousands of activities for few users, while most passive ones reside in the tail, with thousands of users that perform one activity only.

### 5 DISCUSSION

The results presented in previous section are open to a wide range of discussion points. The first remarkable conclusion is that participation inequality varies when consider the GitHub platform as a whole and when considering specific projects. Regarding RQ1, developers do not exhibit participation inequality in their contributions: as we can see in Table 2, GitHub users tend to be active contributors (81%). Considering RQ2, random GitHub projects exhibit some degree of participation inequality (70% of watchers versus 23% of contributors), but do not reach a 90-9-1 distribution. Only when looking at popular projects (RQ3) we notice a 90-9-1 distribution, with 2% of contributors and 87% of watchers.

This data indicates most developers are productive in the context of their own projects, while they mostly become lurkers looking for inspiration more than providing real contributions when they move to others’ projects.

As a consequence, the bigger and more important a project is, the larger is the ratio of observers with respect to contributors. Looking at Figures 1 center and bottom, it is evident that contributors

### Table 2: Classification of GitHub users in each dataset.

<table>
<thead>
<tr>
<th>dataset</th>
<th>contributors ratio</th>
<th>interactors ratio</th>
<th>watchers ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>10K random users</td>
<td>0.8160</td>
<td>0.0726</td>
<td>0.1112</td>
</tr>
<tr>
<td>500 random projects</td>
<td>0.2284</td>
<td>0.0735</td>
<td>0.6980</td>
</tr>
<tr>
<td>500 top projects</td>
<td>0.0214</td>
<td>0.1048</td>
<td>0.8737</td>
</tr>
</tbody>
</table>

⁴http://www.gharchive.org
Figure 1: User activities on 3 datasets: (top) 10k random users, (center) 500 random projects, (bottom) 500 top-starred projects.
Figure 2: Distribution of users belonging to 500 random projects (left) and 500 top projects (right).

decrease by 10 times, which is the actual increase in the watchers. On the other hand, focusing on the volume of activities produced (Figure 2), we observe that top projects have many developers that generate many activities, while random ones have few core contributors. This difference may come from the fact that random projects are quite small with respect to top projects, and it shows the challenge of growing a project with contributors rather than observers.

These insights can be very important from a project management perspective, because they help managers to better understand and optimize the behavior of the community around their projects. They can tune their actions based on whether the goal is to attract (core) contributors or to increase project’s visibility or they can use the analysis dimensions we introduced to discover the presence of similar projects and how they compare with respect to them.

As a final observation, the presented results are extremely different in the 3 samples we studied. Diversity in software projects is a complex task, because it lacks of formal definitions, dimensions and metrics. In this study we provided several dimensions (user vs project, project popularity) to highlight a strong difference that would bias our findings: any study should carefully select the samples in order to provide valuable insights, trying to maximize the diversity of sample projects to have a better picture of the behaviors.

6 THREATS TO VALIDITY

Our work is subjected to a number of threats to validity which we classify into: (1) internal validity, which is related to the inferences made based on the application of our research methodology; and (2) external validity, which discusses the generalization of our findings.

Regarding the internal validity, our dataset construction process relied on the information accessible from the official GitHub API. The API provide access to events related to official logged-in users, while nothing is known about the external observers that access the website: that is, the watchers category is only a subset of the entire true watchers. In addition, the classification of events as watch, interact or contribute is based on our rationale about the participatory semantics we assigned to each event, but different groupings could be still valid and provide different results. Similarly, users are assigned to the most active category they generated an event, without considering any threshold or advanced technique to assign users to categories.

As for the external validity, note that our sample is based on a relatively small number of GitHub projects [21]. Even if we compare a random selection vs the top selection to minimize potential biases, our results should not be generalized to all GitHub projects or to projects in other software development platforms or hosted in private repositories.

7 CONCLUSION

We have shown how participation inequality is also present on the open source world both from a qualitative and a quantitative perspective, highlighting that its impact varies depending on the considered dimensions. On one side, the principle is reversed with respect to classical studies, with most of the users that are active and productive if we consider the GitHub platform as a whole, while on the other hand the inequality becomes evident if we consider the activities inside projects. In addition, we showed that top projects have the strongest separation, almost fitting the 90-9-1 distribution, while random projects have more relaxed ratios: the most popular projects are treated as source of inspiration from most users, who then focus on their own code and projects.

We believe this analysis will help project managers to better understand the participation level of the community around their projects and what actions they can do to steer it one way or the other (to a more passive or a more active role, depending on their goals). As a side effect, the large difference in the results of the two sets of projects highlights once again the importance of a suitable sampling strategy to select relevant projects [10] (where suitable and relevant need to be defined based on the goals of the study) and the need of better tools to build diverse samples [24].

As further work, we will deepen the analysis considering more levels of categorization for activities. In addition, the temporal dimension (when an event happened) could be included in the analysis, showing what are the dynamics that generate the distributions
we showed in this work. Finally, another possibility is to focus on lurkers and analyze if open source projects can benefit from them [3].

ACKNOWLEDGMENTS

This work is partially funded by the H2020 ECSEL Joint Undertaking Project “MegaMo@Rt2: MegaModelling at Runtime” (737494) and the Spanish Ministry of Economy and Competitiveness through the project “Open Data for All: an API-based infrastructure for exploiting online data sources” (TIN2016-75944-R).

REFERENCES


