Newcomer OSS-Candidates: Characterizing Contributions of Novice Developers to GitHub

Ifraz Rehman[1] · Dong Wang[2] · Raula Gaikovina Kula[1] · Takashi Ishio[1] · Kenichi Matsumoto[1]

Abstract

The ability of an Open Source Software (OSS) project to attract, onboard, and retain any newcomer is vital to its livelihood. Although, evidence suggests an upsurge in novice developers joining social coding platforms (such as GitHub), the extent to which their activities result in a OSS contribution is unknown. Henceforth, we execute the protocols of a registered report to study activities of a "Newcomer OSS-Candidate", who is a novice developer that is new to that social coding platform, and has the intention to later onboard an OSS project. Using GitHub as a case platform, we analyze 171 identified Newcomer OSS-Candidates to characterize their contribution activities. Results show that Newcomer OSS-Candidates are likely to target software based repositories (i.e., 66%), and their first contributions are mainly associated with development (commits) and maintenance (PRs). Newcomer OSS-Candidates are less likely to practice social coding, but eventually end up onboarding (i.e., 30% quantitative, 70% follow-up survey) an OSS project. Furthermore, they cite *finding a way to start* as the most challenging barrier to contribute. Our work reveals insights on how newcomers to social coding platforms are potential sources of OSS contributions.

Keywords Newcomers · Open source software · GitHub

1 Introduction

The success of Open Source Software (OSS) has always been based on the continuous influx of newcomers and their active involvement (Park and Jensen 2009). Previous studies have shown evidence that many contemporary projects are at risk of failure, with one of the reasons being the inability to attract and retain newcomers (Fang and Neufeld 2009; Valiev et al. 2018). For example, Coelho and Valente (2017) proposed two strategies that include

newcomers which aim to transfer the project to new maintainers and to accept new core developers. In another study, Steinmacher et al. (2014) presented a model that analyzes the influential forces to newcomers being drawn or pushed away from a project. In contrast, the rise of social coding platforms has led to an explosion of potential developers. GitHub reported[1] around 10 million-plus new users in 2020 and allows over 40 million developers to showcase their skills to the world's largest community (44 million upstream repositories). With this upsurge in user activity, However, the extent to which these developers activities result in a contribution to OSS projects is unknown.

The term newcomer has usually been used in a loose way in literature (Steinmacher et al. 2014). Inspired by the incubation of OSS projects on GitHub, we coin the term "Newcomer OSS-Candidate", *who is not yet a newcomer, but has potential to become one*. Concretely, we define a Newcomer OSS-Candidate as a developer that satisfies these three criteria: 1) is a developer that does not have any prior experience contributing to an OSS project, 2) is a new user to a social coding platform, and 3) has the intention to onboard an OSS project hosted on a social coding platform. Although there is a complete body of work that has studied the barriers and struggles of newcomers (Steinmacher et al. 2014; Steinmacher et al. 2015), none has explored the contribution kinds of Newcomer OSS-Candidates. Most of the work revolves around newcomers that have already onboarded OSS projects.

This study is an execution of the protocol reported by Rehman et al. (2020), using GitHub as a case platform. We studied 171 Newcomer OSS-Candidates and their GitHub repositories, guided by four research questions:

– (*RQ*1) **What kinds of repositories does a Newcomer OSS-Candidate target?** Kalliamvakou et al. (2014) showed that most repositories hosted on GitHub are non-software. However, since Newcomer OSS-Candidates have the intention to later onboard a software project, we would like to test the assumption that *(H1) Newcomer OSS-Candidates are more likely to target software repositories*. Since GitHub users can either create their own upstream repositories or fork existing repositories, we compare these two kinds of repositories.
  We observe that 66% of Newcomer OSS-Candidates target software based repositories. The statistical test indicates that hypothesis *H1* is established. Furthermore, Experimental and Documentation are the most frequently targeted software repository kinds for fork and upstream repositories, i.e., 24% and 21%, respectively.

– (*RQ*2) **What are the kinds of first contributions that come from Newcomer OSS-Candidates?** Hattori and Lanza (2008) showed that OSS projects constantly add new content to software (i.e., development) more frequently than maintaining existing code. Hence, for this RQ, our motivation is to understand whether or not Newcomer OSS-Candidates are more likely to add new content or maintain the repository. Hence, by studying these two types of contributions, we test the hypothesis that *(H2) Contributions to GitHub repositories from Newcomer OSS-Candidates are more likely to do development activities*. We analyze two kinds of GitHub contributions, either a direct contribution through a commit, or a submitted Pull Request (PR).
  For the first commit contributions, we find that 74% of contributions from Newcomer OSS-Candidates are related to development activities. For the first PR contributions, our results show that 60% of contributions are associated with management activities. The statistical tests confirm that our hypothesis *H2* is established in first commit contributions, while is not established in first PR contributions.

---

[1]Statistics from https://octoverse.github.com accessed January 2020

– (*RQ*3) **To what extent do Newcomer OSS-Candidates practice social coding with their first contributions?** Since GitHub is a social coding platform, we would like to explore the extent to which a Newcomer OSS-Candidate is likely to make a social contribution as their first contribution. Specifically, we analyze whether or not a New-comer OSS-Candidate shares code, which is measured by single or multiple authorship on a file. Hence, similar to RQ3, we explore the commit and PR contributions to test the hypothesis *(H3) Newcomer OSS-Candidates are more likely to contribute to a file with multiple authorship.*

Our results show that after joining GitHub, a majority of Newcomer OSS-Candi-dates (i.e., 73% of first commits and 59% of PRs) do not share code with other authors. Moreover, the statistical tests validate that our hypothesis *H3* is not established for both first commit and first PR contributions.

– (*RQ*4) **What is the proportion of Newcomer OSS-Candidates that eventually onboard an OSS project?** In accordance with our definition, we explore the extent to which these Newcomer OSS-Candidates eventually onboard an OSS project. We would like to explore the proportion of Newcomer OSS-Candidates who eventually onboard an OSS project. Additionally, we validate what kinds of barriers that Newcomer OSS-Candidates face when onboarding OSS repositories.

Our quantitative analysis shows that 30% of Newcomer OSS-Candidates eventually onboarded engineered OSS repositories. Complementary, a follow-up user survey shows that 70% of studied participants ended up making contributions to an OSS repos-itory. Newcomer OSS-Candidates strongly agreed that they face the barrier of *finding a way to start*, while *social interaction* received the most mixed responses as a barrier.

The remainder of this paper is organized as follows: Section 2 describes the identification procedure for Newcomer OSS-Candidates. Section 3 reports the approaches and results of our empirical study, while Section 4 discusses the deviations, lesson learned and our find-ings. Section 5 discloses the threats to validity, Section 6 presents related work and finally, we conclude the paper in Section 7. To facilitate replication and future work in the area, we have prepared a replication package, which includes the studied 171 Newcomer OSS-Candidates' repositories, manually labeled datasets, the scripts for the quantitative analyses, and the survey materials. The package is available online at https://github.com/NAIST-SE/NewcomerCandidate.

2 Identifying newcomer OSS-candidates

In this section, we describe the process of identifying Newcomer OSS-Candidates. As per our registered report (Rehman et al. 2020), we used the first-contribution community[2] in GitHub as our data source for collecting Newcomer OSS-Candidates. The community is an initiative established to help beginners make their first contributions on GitHub and cur-rently has over 5,000 contributors, over 39.7 thousand forks, and over 21 thousand stars as of October 2021. To extract the survey respondent candidates, we used command `"git log --pretty=format:%ae"`[3] on Contributors.md file provided by the community and were able to get 17,507 respondent candidates. We sent our online survey invitation[4]

Table 1 Survey Questions sent to potential respondents

Survey Questions for Newcomer OSS-Candidate

Q1) What is your motivation to make a contribution to GitHub?

  (a) Learning to Code.

  (b) Assignment or Experiment Project.

  (c) Intend to contribute to an Open Source.

  (d) Use to showcase my programming skills.

  (e) Others.

Q2) Did you have prior experience contributing to an OSS before GitHub?

  (Yes/No)

to reach up to 4,000 respondent candidates through email and a slack channel.[5] Our survey was open from March 3, 2020 to March 31, 2020 (around a four-week period). We received 208 responses, allowing us to mine their repositories and contributions by providing their GitHub IDs. In the survey, we validate the definition of our Newcomer OSS-Candidate by asking two questions. The two questions are presented in Table 1. Besides, respondents were also asked about their interests, and their perception rank of their programming skills.

171 Identified Newcomer OSS-Candidates Table 2 presents the survey answers that are related to prior OSS experience of respondents and their motivations to contribute. Table 2b shows that 82% of respondents (i.e., 171 responses) intend to contribute to an OSS project. Furthermore, these respondents claim that they have not had any prior OSS experience. Henceforth, according to our definition of Newcomer OSS-Candidate that is described in the Introduction, we used these 171 participants to further track their repositories and contributions for our subsequent analyses.

3 Findings

We follow the protocol that is highlighted in our registered report (Rehman et al. 2020) to answer all RQs. Each research question comprises of the approach and their results. Deviations to the protocol are highlighted in Section 4.1 (Discussion).

3.1 Target repositories (RQ1)

Approach To answer RQ1, we first construct the *(D1) Newcomer OSS-Candidate Repository Dataset*, which is a mapping of our selected Newcomer OSS-Candidate information (as described in Section 2) with their GitHub repository contributions. Using the GitHub REST API (GitHub, 2020) and the credentials of the 171 survey participants, we retrieved 2,392 unique contributed repositories, consisting of 936 fork[6] and 1,456 upstream[7] repositories. Under the guidance of (Borges et al. 2016; Kalliamvakou et al. 2014), we classify the

---

[5]https://firstcontributions.slack.com/

[6]https://docs.github.com/en/get-started/quickstart/fork-a-repo

[7]https://docs.github.com/en/get-started/quickstart/github-glossary#upstream

Table 2 Two questions in our survey

Have you had any prior OSS experience?	Percent	
No	85%	
Yes	15%	

(a) Answers to Q1 of the survey

What is the motivation to contribute?	Percent	
(a) Learning to Code.	58%	
(b) Assignment or Experiment Project.	21%	
(c) Intend to contribute to an Open Source.	82%	
(d) Use to showcase my programming skills.	42%	
(e) Others	5%	

(b) Answers to Q2 of the survey

repositories into software and non-software. The definitions of software and non-software repositories are described below:

– *(Software) Application Software:* systems that provide functionalities to end-users, like browsers and text editors.
– *(Software) System Software:* systems that provide services and infrastructure to other systems, like operating systems, middleware, servers, and databases.
– *(Software) Web libraries and frameworks.*
– *(Software) Non-web libraries and frameworks.*
– *(Software) Software tools:* systems that support software development tasks, like IDEs, package managers, and compilers.
– *(Software) Documentation:* repositories with documentation, tutorials, source code examples.
– *(Software) Experimental:* repositories include demos, samples, test code, and tutorial examples.
– *(Non-Software) Storage:* category includes repositories documents and files for personal use, such as presentation slides, resumes, e-books, music files etc.
– *(Non-Software) Academic:* class and university research projects come under this category.
– *(Non-Software) Web:* under this category we classify websites and blogs.
– *(Others) No longer accessible/Empty:* repositories that gave 404 error, containing only a license file, a gitignore file, a README file, or no files at all were placed under this category.

As per the registered report, we use a qualitative method to manually classify the different kinds of repositories. Following the protocol, with a confidence level of 95% and a confidence interval of 5[8], we draw a statistically representative sample from *(D1)* to end up with 273 fork repositories and 304 upstream repositories. To evaluate the validity of our manual coding, we randomly selected 30 repositories from the representative sample, and then the first three authors independently coded these repositories. The three authors then measured the inter-rater agreement using Cohen's Kappa (Viera et al. 2005) as the measure of agreement. In the end, the Kappa agreement for fork repositories was nearly perfect (i.e.,

---

[8]https://www.surveysystem.com/sscalc.htm

Table 3 Proportion of software and non-software repositories targeted by Newcomer OSS-Candidates. Around 66% of Newcomer OSS-Candidates target Software repositories

Category	Percent (%)	Fork & Upstream (%)
Software	66	Upstream (52)
		Fork (48)
Non-Software	24	Upstream (55)
		Fork (45)
Others	10	-
		-

0.91), while the score for upstream repositories was substantial (i.e., 0.76). Based on this encouraging result, the first author then completed the manual coding for the rest of the representative sample.

For our significance testing, different from the registered report,[9] we validate our hypothesis *(H1) Newcomer OSS-Candidates are more likely to target software repositories*, using the one proportion Z-test (Paternoster et al. 1998) as it compares an observed proportion to a theoretical one when the categories are binary.

Proportion of software and non-software repositories Table 3 shows the proportion of software and non-software based repositories that Newcomer OSS-Candidates target. We see that 66% of Newcomer OSS-Candidates target repositories are software based and follow sound software engineering practices in each dimensions. Furthermore, Newcomer OSS-Candidates are less likely to target non-software based repositories, accounting for 24%. Specifically, we observe that 10% of repositories are classified as Others. Through the manual analysis, these repositories are either "No longer accessible" or "Empty". Upon in-depth analysis of repositories (i.e., Fork and Upstream), we observe that the dominant repositories for software and non-software are upstream i.e., 52% and 55%.

Frequency of contributed repository kinds Figure 1 shows that Documentation (21%), Experimental (15%), Web-based-applications, libraries and frameworks (15%) are the most frequently targeted upstream software repositories kinds. The other kinds of repositories that Newcomer OSS-Candidates frequently target are Academic (12%), Web (11%), and Application Software (9%). On the other side, we find that Experimental (24%) and Web-based-application, libraries, and frameworks (17%) are the most commonly targeted fork repositories kinds. The other kinds of fork repositories commonly targeted are Documentation (13%) and Academic (12%).

Our statistical test validates a significant difference between the proportion of software and non-software repositories that Newcomer OSS-Candidates target, with a p-value < 0.001. The result indicates that our proposed hypothesis, i.e., *(H1) Newcomer OSS-Candidates are more likely to target software repositories*, is established.

> **RQ1 Summary:** Results show that 66% of Newcomer OSS-Candidates target software based repositories. Our proposed hypothesis that *(H1) Newcomer OSS-Candidates are more likely to target software repositories* is established. Furthermore, Experimental and Documentation are the most frequently targeted software repository kinds for both fork and upstream repositories with 24% and 21%, respectively.

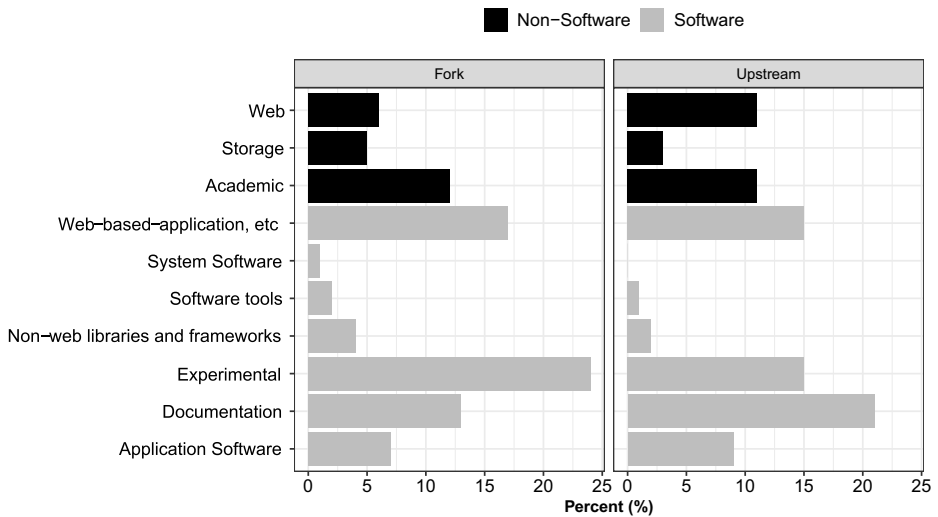[9] Please refer to Section 4.1 for deviations explanations

Fig. 1 Frequency for contributed repository kinds with Fork and Upstream. Experimental and Documentation are the most frequently targeted software repository kinds, i.e., 24% and 21%, respectively

3.2 Kinds of contributions (RQ2)

Approach To answer RQ2, different from the registered report, we analyze the first contributions with two types, i.e., first commit and first PR. As such, we constructed a new dataset from RQ1, which is *(D2) First Contribution Dataset*. To do so, we first obtain the earliest GitHub repositories each of the 171 Newcomer OSS-Candidates. For the quality purpose, we ignore any *test and not meaningful* commits by filtering out experimental repositories that have been identified in RQ1. Furthermore, from our initial list of 171 participants, we remove another five participants. Three participants had not made any contributions to their fork or upstream repositories, and another two participants had become inactive since the initial survey. Hence, we ended up with a total of 166 first commits and 97 PRs from 166 Newcomer OSS-Candidates. As per the registered report, we then classify the contributions according to Hattori and Lanza ([2008]):

- *Development (forward engineering and non-software):* based on the forward-engineering type proposed by Hattori and Lanza (2008), the development activities relate to incorporation of new features and implementation of new requirements for both software and non-software. Examples of development for non-software repositories include adding new content for websites or documentation.
- *Repository Initializing (sub-category of development):* derived from the forward-engineering category, we identify any first commits as the initializing commits to a new repository.
- *Re-engineering:* maintenance activities are related to refactoring, redesign and other actions to enhance the quality of the code without properly adding new features.
- *Corrective Engineering:* maintenance activities handle defects, errors and bugs in the software.
- *Management:* maintenance activities are those unrelated to codification, such as formatting code, cleaning up, and updating documentation.

To validate the understanding of the taxonomy of contribution kinds, we randomly selected 30 contributions of first commits and PRs, and then the first three authors independently coded these contributions, similar to RQ1. Since Hattori and Lanza (2008) used a set of keywords, we applied the keywords as an initial guide. However, when deciding the classification, we consider the commit and PR attributes (i.e., title, message, and description) to have a better understanding of the context. Similar to RQ1, we use Cohen's Kappa. The Kappa agreement scores for classifying contribution kinds of first commits and PRs were both substantial (i.e., 0.72 and 0.79, respectively). After the agreement measurement, the first author then completed the remaining sample.

To validate our hypothesis *(H2) Contributions to GitHub repositories from Newcomer OSS-Candidates are more likely to do development activities*, similar to RQ1, we use the one proportion Z-test (Paternoster et al. 1998). To fit the formula of the statistical test, we merge *Development* and *Repository Initializing* into the *Development* category, and we merge *Re-engineering*, *Corrective Engineering*, and *Management* into the *Maintenance* category.

Frequency of Contribution's Kinds Table 4 depicts the distribution for kinds of contributions made by Newcomer OSS-Candidates. For the first commit contributions, as shown in the table, 31% and 43% of Newcomer OSS-Candidates engage in development activities and repository initializing activities in the first commits. The result suggests that Newcomer OSS-Candidates are more likely to engage in development activities (i.e., 31% + 43% = 74%) when submitting first commits. Upon closer inspection, we find that 98% and 77% of development activities and repository initializing activities involve code related changes. For the first PR contributions, our manual classification shows that 60% of Newcomer OSS-Candidates engage in management activities when submitting their PRs, indicating that

Table 4 Frequency for Contribution's Kinds of Newcomer OSS-Candidates

First Contributions	Kinds	Percent (%)	Code (%)	Doc (%)
First Commit :	Development	31	98	2
	Repository Initializing	43	77	23
	Re-engineering	7	100	0
	Corrective Engineering	2	100	0
	Management	13	5	95
	Others	4	100	0
sum		100		
Pull Request :	Development	9	89	11
	Repository Initializing	3	33	67
	Re-engineering	17	76	24
	Corrective Engineering	6	100	0
	Management	60	45	55
	Others	4	100	0
sum		100		

In the first commits, 43% of Newcomer OSS-Candidates are typically engaged in repository initializing activities, and 60% are engaged in the management activities of the PRs

Newcomer OSS-Candidates are more likely to target maintenance activities. Furthermore, we find that 45% of management activities are related to formatting code, and 55% are associated with cleaning up and updating documentation. More specifically, 4% of their first commits and 4% of first PRs contributions are classified as Others. Through our manual analysis, we find that these contributions are inaccessible (i.e., 404 errors), not be classified into any category based on our taxonomy, or not written in English.

Our statistical tests confirm statistically significant differences between the proportion of development and maintenance activities for both types of contributions (first commit and PR), with a p-value < 0.001. For the type of first commit contributions, the test result validates that Newcomer OSS-Candidates are more likely to engage in development activities. However, for the type of first PR contributions, the test result confirms that Newcomer OSS-Candidates are more likely to be involved in maintenance activities. To conclude, our raised hypothesis, *(H2) Contributions to GitHub repositories from Newcomer OSS-Candidates are more likely to do development activities*, is established in first commit contributions, while it is not established in first PR contributions.

> **RQ2 Summary:** For the first commit contributions, we find that 74% of contributions from Newcomer OSS-Candidates are related to development activities. For the first PR contributions, our results show that 60% of contributions are associated with management activities. Furthermore, statistical tests confirm that *(H2) Contributions to GitHub repositories from Newcomer OSS-Candidates are more likely to do development activities* is established in first commit contributions, but it is not established in first PR contributions.

3.3 Social coding in terms of multiple authorship (RQ3)

Approach Social coding is a very loose term (Dabbish et al. 2012) used to describe the ability for developers to advertise (openly share and allow modification) their code on social platforms such as GitHub. In our paper, as shown in Fig. 2, we select one social coding practice in terms of multiple authorship to analyze where a contributor modifies either someone else's codes or others may modify this contributor's codes in the future. In the example, there are two authors (i.e., author A for lines 1–3 and author B for line 4) that contribute to a single file (i.e., git.gemspec) in a repository (i.e., ruby-git). To do so, we use the *D2* dataset

Fig. 2 An example of how we define developers practice social coding, where more than one author contributes to the git.gemspec file

from RQ2, which contains first commit and first PR contributions. We identify social coding using Algorithm 1 and the `git-blame`[10] command on each contained file in the commit to check whether the files receive changes from more than one author (lines 3–4 in Algorithm 1). Considering that one PR may include multiple commits, we analyze all commits inside each PR with Algorithm 1. Specifically, we found that 21 out of 97 PRs (22%) have multiple commits.

Algorithm 1 Identify social coding in terms of whether a contribution is modified by a single author or multiple authors.

   Input   : First $Commit/PR$ performed by an author au
   Output : Contribution type of First $Commit/PR$: single or multiple authors
1 $F \leftarrow$ A set of files modified by First $Commit/PR$;
2 $Type(F) =$single author;
3 **for** $f \in F$ **do**
4     $D \leftarrow extract\_authors(\texttt{git-blame}(f));$
5   **if** $au \in D$ & $|D| > 1$ **then**
6         $Type(f) =$ multiple author;
7   **end**
8 **end**
9 **return** $Type$;

To validate our hypothesis *(H3) Newcomer OSS-Candidates are more likely to contribute to a file with multiple authorship*. Similar to RQ1, we use the one proportion Z-test (Paternoster et al. 1998).

Social coding (Multiple Authorship) Table 5 presents the frequency of social and non-social contributions in terms of authorship done by Newcomer OSS-Candidates. As shown in the table, the majority of Newcomer OSS-Candidates do not practice social coding after joining GitHub. For instance, we find that 73% of the first commits and 59% of the first PRs are contributed by a single author. Such results suggest that Newcomer OSS-Candidates are less likely to practice social coding in terms of sharing multiple authorship, when placing their first GitHub contributions.

Our statistical test validates that for the first commits, there is a statistically significant difference between the proportion of social and non-social contributions, with a p-value $<$ 0.001, where Newcomer OSS-Candidates are likely to practice non-social coding. For the first PRs, there are no statistically significant difference, with a p-value $>$ 0.05. To conclude, our proposed hypothesis *(H3) Newcomer OSS-Candidates are more likely to contribute to a file with multiple authorship*, is not established in both first commits and PRs.

> **RQ3 Summary:** Our results show that after joining GitHub, a majority of Newcomer OSS-Candidates (i.e., 73% of first commits and 59% of PRs) do not share code with other authors. Furthermore, statistical tests validate that *(H3) Newcomer OSS-Candidates are more likely to contribute to a file with multiple authorship*, is not established in both first commit and PR contributions.

Table 5 Frequency of social and non-social contributions from Newcomer OSS-Candidates in terms of single/multiple authorship

Social coding practice (First Commit)	Percent (%)	
multiple	27	
single	73	
Social coding practice (Pull Request)	Percent (%)	
multiple	41	
single	59	

After joining GitHub, 73% and 59% of Newcomer OSS-Candidates have non-social based contributions in their first commits and PRs

3.4 Onboarding of newcomer OSS-candidates (RQ4)

Approach To answer RQ4, we perform both quantitative and qualitative analyses. Different from the registered report, we find that making contributions to an OSS project is not trivial, and involves a process that follows two steps:

– *Fork an OSS repository*. The first step for any Newcomer OSS-Candidate is to fork an OSS repository. Hence, we extracted 936 fork repositories out of a total of 2,392 repositories from the *D1* dataset. Then, to identify whether this repository is an engineered software project, we matched each fork repository against a curated dataset by Munaiah et al. (2016).
– *Identify contributions*. During step one, we found that many participants who only fork the repository, without contributing back to either the fork or upstream repository. Hence, we performed an in-depth analysis through two particular ways of onboarding i.e., either the fork or upstream repositories.

For the qualitative analysis, we conducted a follow-up survey[11] to acquire the perception of our participants. We sent our online survey invitation to Newcomer OSS-Candidates through emails and ended up receiving 27 responses. The survey is split into two questions, confirming whether participants had contributed to an OSS repository. The first question is related to whether the participant had onboarded an OSS project (i.e., Since joining GitHub, did you successfully make a contribution to any Open Source Software project?). In the second question, we explore the barriers faced by OSS newcomers (Steinmacher et al. 2014). Hence, we asked participants to rate each barrier (i.e., Social Interaction, Newcomer Previous Knowledge, Finding a Way to Start, Technical Hurdles, and Documentation) on a five-point Likert scale.

Onboarding Process in GitHub Table 6 presents the distribution of how Newcomer OSS-Candidates onboard OSS projects in terms of the quantitative analysis. We show that 49% of Newcomer OSS-Candidates onboard OSS projects, while 51% do not. Furthermore, 51% of Newcomer OSS-Candidates only fork the OSS repositories not making any contributions (Fork an OSS repository), and 22% have contributed in the form of making commits to their own fork OSS repositories (Contributed to fork OSS repository). Meanwhile, 30% of Newcomer OSS-Candidates eventually onboard by submitting PRs directly to the original OSS repositories (Contributed to original OSS repository). On the other hand, for the qualitative

---

[11]Survey details are available at https://forms.gle/JQiVamovUXdJiy8z5

Table 6 Frequency of Newcomer OSS-Candidates that started the onboarding process for OSS repositories

Match to the Munaiah(2016) dataset	Onboarding Steps	Count (#)	Percent (%)
Started Onboarding			
Process :		81	49
	Fork an OSS repository (51%)		
	Contribute to fork OSS repository (22%)		
Eventually Onboarded:	Contribute to original OSS repository (30%)		
Not Onboard:		85	51
Sum		166	100

analysis, the survey results show that 19 out of 27 Newcomer OSS-Candidates (70%) claim that they have made contributions to OSS repositories. Figure 3 (a) shows the distribution of Newcomer OSS-Candidates onboarding OSS projects by means of qualitative analysis.

Barriers faced by Newcomer OSS-Candidates Figure 3 (b) shows the results of our Likert-scale question related to barriers. The figure shows that *finding a way to start* is the most crucial barrier, with 22 responses being positive (i.e., 12 agree and 10 strongly agree responses). The second most crucial barrier is *technical hurdles*, receiving 18 positive responses (i.e., 15 agree and 3 strongly agree responses). *Newcomer previous knowledge* is considered the third most crucial barrier with 16 responses (i.e., 10 agree and 6 strongly agree responses). On the other hand, the respondents are more likely to disagree with the statement that *social interaction* and *documentation* can be barriers for them to onboard OSS projects (i.e., 7 negative responses for each barrier).

> **RQ4 Summary:** Our quantitative analysis shows that 30% of New-comer OSS-Candidates eventually onboarded OSS projects. Our follow-up user survey also shows that 19 out of the 27 participants (70%) claim that they have made contributions to OSS repositories. We find that *finding a way to start* is the most agreed barrier for Newcomer OSS-Candidates.
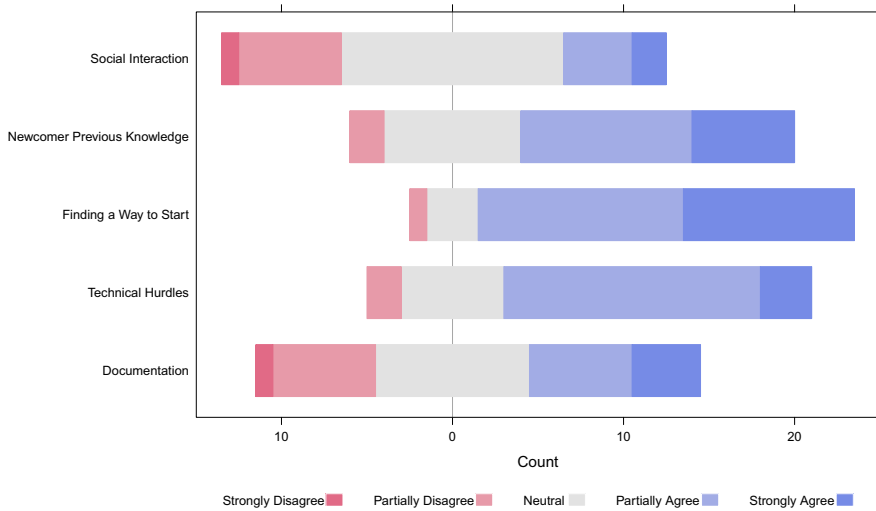
4 Discussions

In this section, we discuss deviations from the registered report, lessons learned and then revisit our expected implications listed in the registered report against the actual results.

4.1 Deviations

The execution of this registered report (RR) prompted unavoidable changes to our proto-cols. We list up the following four deviations below: *(i) Term Newcomer OSS-Candidate.* To generalize the definition of the term, Newcomer Candidate has been changed to Newcomer OSS-Candidates as "a developer that does not have any prior experience contributing to an OSS project, is a new user to a social coding platform, with the intention to onboard an OSS project", *(ii)Terminology Clarification.* In the registered report, our preliminary study

Response to "Since joining GitHub, did you successfully make a contribution to any OSS project?"	Count (#)	Percent (%)
Has made a contribution to an OSS project.	19	70%
Has never made a contribution to an OSS project.	8	30%
sum	27	100

(a) Answers to Q1 in the follow-up survey.

(b) Barriers faced by Newcomer OSS-Candidates. Most Newcomer OSS-Candidates (i.e., 22 out of 27 responses) strongly agree that *finding a way to start* is a barrier.

Fig. 3 Qualitative analysis using a follow-up survey to acquire the perception of Newcomer OSS-Candidates

is now a separate section in the full study. In terms of clarity, in the executed study, we specify the social coding practice as the number of authors on a shared file, and realize that onboard is an ongoing process, *(iii) Research Design.* The statistical test has been changed to one proportion Z-test (Paternoster et al, 1998). After revising the categories, we realized that the statistical test in the RR was not appropriate. We modified the statistical tests based on the binary result categories of RQ1, RQ2, and RQ3. The one proportion Z-test compares an observed proportion to a theoretical one when the categories are binary, and last *(iv) Hypothesis.* We adjusted the hypotheses H2 and H3. For H2, we changed it to *(H2) Contributions to GitHub repositories from Newcomer OSS-Candidates are more likely to do development activities*, to be aligned with our motivation. For H3, we narrowed down the aspect of social coding and adjusted it to *(H3) Newcomer OSS-Candidates are more likely to contribute to a file with multiple authorship.*

4.2 Lessons learned

This paper discusses two lessons learned that would be useful for future replication or improvements of the study. In the first lesson, we acknowledge that extracting the first

contribution is not as trivial as we first envisioned. This is because the actual first commit might be just an ad-hoc test for the user, and not an actual meaningful contribution to a repository. In this research, we manually filtered out such contributions, but future work should consider a more systematic approach.

The second lesson to acknowledge is the process of onboarding may take a long time as it may be tied with the process of making a contribution to GitHub. As shown in the results for RQ4, different Newcomer OSS-Candidates are at different stages of the onboarding process and may take time before they decide to submit the PR. Thus, we need to take into consideration a long enough time-window to evaluate whether or not a Newcomer OSS-Candidate will end up onboarding an OSS project.

4.3 Implications (expectations vs. actual results)

Based on our results, we revisit our expected implications against the actual results of the study.

Suggestions for Newcomers In our registered report, we speculated that our research would help Newcomer OSS-Candidates understand the kinds of contributions they target before onboarding a real OSS project. Actually, we found in Table 4 that Newcomer OSS-Candidates are not only engaged in adding new content, but 60% of them are also interested in management activities related to formatting code, cleaning up, and updating documentation through the submission of PRs. One example of this can be seen in the AEOL's repository[12], where a PR is submitted to add a new function to the project. Furthermore, RQ2 also reveals that after joining GitHub, 43% of Newcomer OSS-Candidates prefer to add new content in order to initialize or start a repository in their first commit. We found a common pattern is an initial commit that is uploading a website to the GitHub repository.[13] Finally, based on our RQ3 quantitative analysis, the majority of Newcomer OSS-Candidates have non-social based contributions in their contributions. As shown in Table 5 from RQ3 that after joining GitHub, Newcomer OSS-Candidates contributes in terms of single authorship are 73% of their first commits and 59% of their PRs, respectively. On the basis of evidence, we conclude that it is unlikely that Newcomer OSS-Candidates will be onboard to OSS projects immediately after joining GitHub.

We also speculated that we would reveal barriers on why some Newcomer OSS-Candidates never end up contributing to an OSS projects. According to our survey responses in RQ4, finding a way to start is one of the most challenging barriers, with 22 responses being positive (i.e., 12 agree and 10 strongly agree responses). Hence, inspired by these examples and combining all results, we recommend that Newcomer OSS-Candidates should not be afraid to individually contribute to their own code, contribute to upstream software repositories, or fork OSS projects before attempting to onboard. Last, regarding the most challenging barrier (i.e., finding a way to start), to this end, Newcomer OSS-Candidates should leverage suggestions provided by Subramanian et al. (2020), including minor feature additions (a change of around 36 lines of code), minor documentation changes, selecting bug fixes, and changing catering to revised dependencies as first-timer friendly, which may relieve this problem. In addition, there are online resources[14] that

---

[12]https://github.com/AE0L/round-robin/pull/1

[13]https://github.com/maanizfar/vanilla-js-web-projects/commit/e208d861b80762be8aa545567a300a7fad6aacf7

[14]https://hacktoberfest.digitalocean.com/

help Newcomer OSS-Candidates choose easy issues or opportunities to find ways to start contributing.

Suggestions for OSS Projects The registered report speculated that the findings would reveal insights into what contributions may attract a Newcomer OSS-Candidate. Through our qualitative analysis of RQ2, Table 4 shows that in the first commits, 43% of Newcomer OSS-Candidates are typically engaged in adding new content to initialize the repositories, and 60% are involved in management activities in their PRs. Hence, we suggest that Newcomer OSS-Candidates may not have required skills to make immediate contributions. Instead, they may start with software based upstream experimental repositories. Hence, for OSS project, it might start with tasks to update the documentation, formatting or cleaning up code. One example of this can be seen in Bviveksingh's upstream repository[15], where a PR is submitted to update a software version.

We also speculated that OSS projects may benefit from our study, by identifying and offering the right contributions for the right Newcomer OSS-Candidates. Based on the results, we could not be able to provide concrete examples of contributions that match a specific Newcomer OSS-Candidate as the majority is a mixture of management and development activities. A potential future venue for research could be to explore the kinds of OSS projects that these Newcomer OSS-Candidates end up onboarding. This would provide insights into matching the contributions to the onboarded OSS projects.

Suggestions for Researchers The registered report speculated that non-software repositories that are personal have always been regarded as a challenge and are often filtered out from the dataset. We find that the majority of targeted repositories are software based repositories. Results include experimental (24%), documentation (21%), and web-based-application-libraries-and-frameworks (17%). For researchers, this insight helps to understand the role of software based experimental, documentation, and web-based-application-libraries-and-frameworks repositories in platforms like GitHub, that should cater for developers. A potential avenue for research is to perform a finer-grain of analysis to understand the nature of these repositories.

5 Threats to validity

In this section, we now discuss threats to the validity of our study.

External Validity Two external threats are identified. We perform an empirical study on Newcomer OSS-Candidates that use GitHub the platform, and our observations may not be generalized to other platforms. Hence, we use GitHub as a case study. Another external threat is whether or not the 171 participants are representative of all Newcomer OSS-Candidates of the GitHub platform. Hence, we rely on the first contribution community. To represents the global population, future work should be conducted with other communities.

Construct Validity We summarize three threats regarding construct validity. First, our qualitative analysis of manually classifying repositories and contribution kinds (RQ1, RQ2) are prone to error. To mitigate this threat, we took a systematic approach to first test our

---

[15]https://github.com/Bviveksingh/angular-starter/pull/1

comprehension with 30 samples using Kappa agreement scores with three separate individuals. The second threat is to identified first contributions in RQ2 may not be actual contributions. To mitigate this, we perform a manual inspection to ignore any test, not meaningful contributions (i.e., commits or PRs) from any experimental repositories. The third potential threat exists in the quantitative analysis of matching engineered software projects using the curated database provided by Munaiah et al. (2016). We did contact the authors for assistance to help run the latest scripts, but were unsuccessful. Although the curated database might be outdated, we are confident that with the dataset, we were able to match 936 repositories.

Internal Validity We identify three internal threats. The first threat is the first contributions by Newcomer OSS-Candidates may not be meaningful; they just want to get into the GitHub way of doing things. To mitigate this, we applied our first filter. The second internal threat to validity is related to results obtained from the quantitative analysis of RQ3 adapted to data visualization. As per the result, 27% and 41% of social coding is done by Newcomer OSS-Candidates in their first commits and PRs. The final threat is regarding errors in our tracking of repositories, due to repositories being deleted or a user changing user ids, as studied by Wiese et al. (2016). We acknowledge this threat, however, during our manual inspection, we are confident that this was only for a few cases.

6 Related work

A steady of influx of new developers to an OSS project is crucial for its sustainability. In this section, we compare and contrast our work to the prior studies in three parts: first, we introduce the studies that are related to motivation for newcomers and OSS projects; second, we consider the studies regarding onboarding OSS projects; third, we discuss the studies with respect to the barriers that newcomers face.

Studies on Onboarding Motivators There is a complete body of work that explored OSS developer's motivation and project's attractiveness (Meirelles et al. 2010; Santos et al. 2013; Shah 2006; Ye and Kishida 2003). Studies have also investigated the progression from newcomer to a core project member (Ducheneaut 2005; Fang and Neufeld 2009; Krogh et al. 2003; Marlow et al. 2013; Nakakoji et al. 2003). On the other hand, Choi et al. (2010) identified the seven most frequently used socialization tactics which have impact on newcomers' commitment to online groups. Other parts of the literature focus on the forces of motivation and attractiveness that drive newcomers towards projects. For example, (Lakhani 2003) have found that external benefits (e.g., better jobs, career advancement) motivate primarily new contributors, along enjoyment-based intrinsic, code-based challenges, and improving programming skills. Compared to these, our study investigates how Newcomer OSS-Candidates contribute to both software (e.g., experimental, documentation, and web-based-application-libraries-and-frameworks) and non-software (e.g., academic, Web, and storage) repositories. Different to prior work, our goal is to study potential Newcomer OSS-Candidates that have the intention to onboard an OSS project.

Studies on the Onboarding Process There have been several studies that investigated the onboarding process. Fagerholm et al. (2013) presented preliminary observations and results of in-progress research that studied the process of onboarding into virtual OSS teams. Commercial software development settings are also affected by newcomers onboarding towards

OSS projects, as described by Dagenais et al. (2010) and Begel and Simon (2008). Ducheneaut (2005) approached onboarding from a sociological point of view by considering the perspective of individual developers. Previously, mentorship activity is recognized as an important factor for effective onboarding of newcomers towards OSS projects (Fagerholm et al. 2013; Fagerholm et al. 2014; Musicant et al. 2011). Swap et al. (2001) described mentoring in their study as a basic knowledge transfer mechanism in the enterprise. A joining script is proposed in another study by Krogh et al. (2003) for developers who want to take participate in OSS project. Nakakoji et al. (2003) also studied the OSS project and proposed eight possible joining roles comprise of concentric layers called "the onion patch". Zhou and Mockus (2015) found that the willingness of individual and project's climate were associated with odds that an individual would become a long-term contributor. Different from previous research, our study looks at the activities of potential newcomers before they onboard.

Studies on the barriers to Onboarding Newcomers are important to the survival, long-term success, and continuity of OSS projects (Kula and Robles 2019). However, newcomers face many difficulties when making their first contributions to a project. According to (Ye and Kishida 2003), learning is one of the motivational forces that motivates people to participate in OSS communities. Conversely, newcomers to a project send contributions which are not incorporated into the source code and give up trying (Steinmacher et al. 2015). As discussed by Zhou and Mockus (2010), the transfer of entire projects to offshore locations, aging and renewal of core developers in legacy products, recruiting in fast growing Internet companies, and the participation in open source projects, presents similar challenges of rapidly increasing newcomer competence in software projects. Several research activities are performed to reduce the barriers for newcomers previously. Steinmacher et al. (2014) proposed a developer joining model that represents the stages that are common and the forces that are influential to newcomers being drawn or pushed away from a project. Steinmacher et al. (2016) created a portal called FLOSScoach based on a conceptual model of barriers to support newcomers. The evaluation shows that FLOSScoach played an important role in guiding newcomers and in lowering barriers related to the orientation and contribution process. In terms of barriers, our research complements the work of Steinmacher et al. (2014), which highlights the most crucial barrier among others, i.e., finding a way to start, due to which newcomers face difficulty in contributing to OSS projects. Furthermore, our work takes a first look at potential Newcomer OSS-Candidates before they onboard. Hence, insights show that learning the social platform contribution process (i.e., PR process) may co-inside with onboarding.

7 Conclusion

In this work, we studied the activities of a particular category of potential contributors (i.e., Newcomer OSS-Candidates) towards OSS projects on GitHub. To do that, we (i) analyze what kinds of repositories they target, (ii) investigate what kinds of contributions come from them, (iii) analyze to what extent they practice social coding with their contributions, and (iv) explore what proportion of them eventually onboard an OSS project.

We observe that (i) 66% of Newcomer OSS-Candidates target software based repositories; (ii) the majority of their contributions are related to development activities and maintenance activities, respectively, for commits and PRs; (iii) Newcomer OSS-Candidates are less likely to practice social coding in their contributions in terms of multiple authorship; and (iv) 70% of them eventually onboarded OSS projects in a follow-up survey and

cited that *finding a way to start* is the most crucial barrier. As GitHub continues to grow, so does the possibility to attract potential contributors to OSS projects. Our work presents the first step towards understanding these potential contributors and reveals insights to provide a guidance for them to onboard an OSS project.

Declarations

Conflict of Interests The third author (Raula Gaikovina Kula) is an Editorial board member.

References

Begel A, Simon B (2008) Novice software developers, all over again. In: ICER'08 - Proceedings of the ACM workshop on international computing education research

Understanding the factors that impact the popularity of GitHub repositories (2016)

Choi B, Alexander K, Kraut RE, Levine JM (2010) Socialization tactics in wikipedia and their effects. In: Proceedings of the 2010 ACM conference on Computer supported cooperative work, pp 107–116

Coelho J, Valente MT (2017) Why modern open source projects fail. In: FSE

Dabbish L, Stuart C, Tsay J, Herbsleb J (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, pp 1277–1286

Dagenais B, Ossher H, Bellamy RKE, Robillard MP, de Vries JP (2010) Moving into a new software project landscape. In: Association for computing machinery, pp 275–284

Ducheneaut N (2005) Socialization in an open source software community: a socio-technical analysis. Comput Supported Cooperative Work (CSCW) 14:323–368

Fagerholm F, Johnson P, Guinea A, Borenstein J, Münch J (2013) Onboarding in open source software projects: A preliminary analysis. In: 2013 IEEE 8th international conference on global software engineering workshops

Fagerholm F, Guinea AS, Münch J, Borenstein J (2014) The role of mentoring and project characteristics for onboarding in open source software projects. In: Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement, pp 1–10

Fang Y, Neufeld D (2009) Understanding sustained participation in open source software projects. J Manage Inf Syst

Hattori LP, Lanza M (2008) On the nature of commits. In: ASE

Kalliamvakou E, Gousios G, Blincoe K, Singer L, German DM, Damian D (2014) The promises and perils of mining GitHub. In: MSR

Krogh G, Spaeth S, Lakhani K (2003) Community, joining, and specialization in open source software innovation: a case study. Res Policy 32:1217–1241

Kula RG, Robles G (2019) The life and death of software ecosystems. Springer, Berlin, pp 97–105

Lakhani K (2003) Wolf r. Understanding motivation and effort in free/open source software projects. Perspectives on Free and Open Source Software, Why hackers do what they do

Marlow J, Dabbish L, Herbsleb J (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. Association for Computing Machinery, New York, NY, USA, CSCW '13, p 117–128

Meirelles P, Santos JrC, Miranda J, Kon F, Terceiro A, Chavez C (2010) A study of the relationships between source code metrics and attractiveness in free software projects. In: 2010 Brazilian symposium on software engineering, pp 11–20

Munaiah N, Kroh S, Cabrey C, Nagappan M (2016) Curating github for engineered software projects. EMSE

Musicant DR, Ren Y, Johnson JA, Riedl J (2011) Mentoring in wikipedia: a clash of cultures. In: Proceedings of the 7th international symposium on Wikis and Open Collaboration, pp 173–182

Nakakoji K, Yamamoto Y, NISHINAKA Y, Kishida K, Ye Y (2003) Evolution patterns of open-source software systems and communities. International Workshop on Principles of Software Evolution (IWPSE)

Park Y, Jensen C (2009) Beyond pretty pictures: Examining the benefits of code visualization for open source newcomers. In: VISSOFT

Paternoster R, Brame R, Mazerolle P, Piquero A (1998) Using the correct statistical test for the equality of regression coefficients. Criminology 36(4):859–866

Rehman I, Wang D, Kula RG, Ishio T, Matsumoto K (2020) Newcomer candidate: Characterizing contributions of a novice developer to github. In: 2020 IEEE international conference on software maintenance and evolution (ICSME), pp 855–855

Santos C, Kuk G, Kon F, Pearson J (2013) The attraction of contributors in free and open source software projects. J Strateg Inf Syst 22(1):26–45

Shah S (2006) Motivation, governance, and the viability of hybrid forms in open source software development. Manag Sci 52:1000–1014

Steinmacher I, Gerosa MA, Redmiles D (2014) Attracting, onboarding and retaining newcomer developers in open source software projects. In: CSCW

Steinmacher I, Conte T, Gerosa MA, Redmiles DF (2015) Social barriers faced by newcomers placing their first contribution in open source software projects. In: CSCW

Steinmacher I, Conte TU, Gerosa MA (2015) Understanding and supporting the choice of an appropriate task to start with in open source software communities. In: HICSS

Steinmacher I, Conte TU, Treude C, Gerosa MA (2016) Overcoming open source project entry barriers with a portal for newcomers. In: ICSE

Subramanian VN, Rehman I, Nagappan M, Kula RG (2020) Analyzing first contributions on github: What do newcomers do. IEEE Softw, 0–0

Swap W, Leonard D, Shields M, Abrams L (2001) Using mentoring and storytelling to transfer knowledge in the workplace. J of Management Information Systems 18:95–114

Valiev M, Vasilescu B, Herbsleb J (2018) Ecosystem-level determinants of sustained activity in open-source projects: A case study of the pyPI ecosystem. In: FSE

Viera AJ, Garrett JM et al (2005) Understanding interobserver agreement: The kappa statistic. Fam Med 37(5):360–363

Wiese IS, Da Silva JT, Steinmacher I, Treude C, Gerosa MA (2016) Who is who in the mailing list? comparing six disambiguation heuristics to identify multiple addresses of a participant. In: 2016 IEEE International conference on software maintenance and evolution (ICSME). IEEE, pp 345–355

Ye Y, Kishida K (2003) Toward an understanding of the motivation open source software developers. In: Proceedings of the 25th international conference on software engineering, IEEE Computer Society, USA, ICSE '03, p 419–429

Zhou M, Mockus A (2010) Growth of newcomer competence: Challenges of globalization. In: FoSER

Zhou M, Mockus A (2015) Who will stay in the floss community? modeling participant's initial behavior. TSE

Affiliations

Ifraz Rehman[1] · Dong Wang[2] · Raula Gaikovina Kula[1] · Takashi Ishio[1] · Kenichi Matsumoto[1]

Dong Wang
wang.dong.vt8@is.naist.jp

Raula Gaikovina Kula
raula-k@is.naist.jp

Takashi Ishio
ishio@is.naist.jp

Kenichi Matsumoto
matumoto@is.naist.jp

[1]   Nara Institute of Science and Technology, Ikoma, Japan

[2]   Kyushu University, Fukuoka, Japan