0.1 Background

In my empirical analysis, I examine how changes in the cost of knowledge acquisition and communication across the hierarchy affect hierarchical structure and the work of OSS contributors. I study the impact of increases in the cost of knowledge acquisition by examining knowledge turnover caused by contributor departures. I study the impact of decreases in communication cost by examining the impact of issue and pull request template adoption by projects.

1. Contributor Turnover
2. Issue & Pull Request Templates

0.2 Knowledge Turnover

Question: What is the causal impact of knowledge turnover on OSS hierarchical structure and contributor time allocation?

0.2.1 Context

One of the major worries with OSS development is the reliance of projects on a few contributors, who, without a contractual obligation of a project, may disappear at a moments notice and bring with them the knowledge and skills necessary for key components of, or in some cases, the whole project, to function. One way to measure variation in the cost of knowledge acquisition in projects is to exploit OSS developer turnover. When a key contributor for an OSS project leaves, it is much harder for new contributors to acquire knowledge about projects because a source of information is gone.
It would be good for me to explore some examples of this in the data, so I can tell an interesting story about the impact of contributor turnover. Some initial questions

- How were areas where this contributor was involved in affected deferentially compared to other areas?
- Do we observe evidence of reduced discussion in issues related to a PR (less learning?)
- Do we observe evidence of less PR review comments & slower PR reviews?
- Do we observe evidence that future PRs by contributors who would be affected are rejected at higher rates?

0.2.2 Adoption and Treatment

1. We can consider variation in turnover by examining the "truck" factor of projects, which measures the number of key members. There are also variants to this measure, such as looking at the identity of the person who wrote the last line of code, as opposed to total lines written per person.

2. I should also control for the presence of knowledge preservation tools. The impact of contributors leaving will be lessened by the presence of good documentation. For example, if a departing contributor wrote a lot of documentation count, that means that they probably did preserve their knowledge for future contributors so their departure will not impact knowledge acquisition by as much. I can measure this by looking at the amount of code for '.md', '.rst' or other documentation files types that they wrote.

0.2.3 Good Descriptive Facts

- What is the rate of turnover? Contextualize this with information about average project size.

- What does the distribution of importance (for departing contributors) look like? Can contexualize using a variety of measures such as LOC written (truck factor), issues commented on, PRs reviewed, etc

0.2.4 Predicted Empirical Effects - Hierarchical Structure

I am interested in three metrics related to organizational structure

1. Span: Ratio of higher to lower ranked contributors
   Garicano's model predicts that there will be more write rank contributors
   What does my model predict?

2. Frequency: % of all problems solved at each layer
   Garicano's model predicts that more problems will be solved by highly ranked contributors
   What does my model predict?

3. Output: How many problems are being solved
   Garicano's model predicts that overall production will decrease.
   What does my model predict?

On span: Since some organizations don't actively manage hierarchies, I may want to filter by "activeness" (separately, there may be something interesting to say about "activeness" versus actual hierarchy, along the lines of security). I may also want to just consider the numerator (write rank contributors) as the project can only control the quantity of promotions, not new contributors.
On frequency: I define a problem as a pull request, although not all pull requests are created equal. In this case, I can weight pull requests by lines of code, files changed (removing moved files) or other measures (that I think of). Separately, since the flow of read-rank contributors cannot be controlled, I may want to just consider the % of all problems solved by write rank contributors. I can start by assuming problems are solved individually and validate this by examining the proportion of code written/PR by one individual. It will also be interesting to break down where the changes in % problems solved are coming from.
On output: This is my opportunity to connect organizational structure outcomes to OSS development outcomes such as quantity or % of PRs merged, and % of opened issues that are closed. There may also be non-code related development metrics of interest. What are outcomes that organizations care about?

0.2.5 Predicted Empirical Effects - Contributor Characteristics

I am interested in three metrics related to individual conrtibutors

1. Individual Output: How many problems is each contributor solving?
   Garicano's model predicts that each read rank contributor will solve less problems and write rank contributor will solve more
   What does my model predict?

2. Individual Skill: How knowledgeable are they?
   Garicano's model predicts that each read rank contributor will lose skill and write rank contributors will gain skill
   What does my model predict?

3. Individual Value Added: What is the value of the problems they're solving Garicano's model doesn't have a prediction for this
   I don't have a prediction for this either but I think it would be fun to answer. It's probably not the most important thing to spend time on though.

On individual output: I may encounter measurement issues due to integer constraints with "problems". This is a good place to incorporate ideas relating to problem

weighting from the hierarchical structure section.
On individual skill: Some examples of skill measures are the length of time required to solve a problem, the number of LOC someone has written (for a project), their quantity of GH badges. The problem is that problem difficulty is an unobserved confounder. Two fun ideas to measure problem difficulty are how close ChatGPT can answer the question and the cyclomatic complexity of the problem (former is hard to implement, latter has endogeneity issues).
On value: One fun idea is to examine the text that's commonly observed in SO for this python library, and see how similar it is to a PR/issue - high value indicates high similarity (solving something people are asking about).

0.2.6 Literature

Economics
OSS
Rashid, Clarke, and O'Connor 2017 has a fairly good literature review on knowledge loss in OSS. Nassif and Robillard 2017 provides interesting comparisons about different statistics for measuring knowledge loss.
Some papers that I'm hoping to read are

1. https://ieeexplore.ieee.org/abstract/document/8870181?casa_token=1rqBePn8XnoAAAAA:wCuI
   CNViaN8n-_20B3PaSKabeFN2vp5ZX

2. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4755634 (need to cite)

3. https://doras.dcu.ie/29119/1/RashidIJIMPaper-RevisionFollowingPeerReview.pdf (need to cite)

4. Managing knowledge assets for open innovation: a systematic literature review (need to cite)

5. Exploring the Foundations of Cumulative Innovation: Implications for Organization Science (need to cite)

0.3 Issue & Pull Request Templates

0.3.1 Summary

In February of 2016 (https://github.blog/?s=issue+templates), GitHub released issue and pull request templates. GitHub stated that "now project maintainers can add templates for Issues and Pull Requests to projects, helping contributors share the right details at the start of a thread". This is important because it ensures that OSS contributors have completed required checklist items before opening an issue/PR, and also makes it easier for reviewers to identify to find particular key pieces of information.

0.3.2 Adoption and Treatment Notes

I can measure initial adoption of PR and Issue templates by looking at when an ISSUE_TEMPLATE (optional ".md" extension?) file or PULL_REQUEST_TEMPLATE (optional ".md" extension?) was created. I can compare projects that adopted these templates during similar timeframes because while the particular timing of adoption might be adoption, adoption within a certain timerange tells us a lot about the attitudes of the OSS contributors towards improving their projects. In May 2018, users acquired the option to choose a template from (presumably) a variety of options when making a new issue (source). However, while this implies that there may be multiple treatments, this is still easier to analyze because there are less possible combinations (less templates) and adoption is more spread out throughout time. Are there any interesting stories I can tell wrt to issue/pr templates?

0.3.3 Predicted Empirical Effects - Hierarchical Structure

I am interested in three metrics related to organizational structure

1. Span: Ratio of higher to lower ranked contributors
   Garicano's model predicts that there will be more write rank contributors
   What does my model predict?

2. Frequency: % of all problems solved at each layer
   Garicano's model predicts that more problems will be solved by highly ranked contributors
   What does my model predict?

3. Output: How many problems are being solved
   Garicano's model predicts that overall production will increase
   What does my model predict?

0.3.4 Predicted Empirical Effects - Contributor Characteristics

I am interested in three metrics related to individual conrtibutors

1. Individual Output: How many problems is each contributor solving?
   Garicano's model predicts that each read rank contributor will solve less problems and write rank contributor will solve more
   What does my model predict?

2. Individual Skill: How knowledgeable are they?
   Garicano's model predicts that each read rank contributor will lose skill and write rank contributors will gain skill
   What does my model predict?

3. Individual Value Added: What is the value of the problems they're solving
   Garicano's model doesn't have a prediction for this
   I don't have a prediction for this either but I think it would be fun to answer. It's probably not the most important thing to spend time on though.

0.3.5 Literature

Economics OSS Some papers that I'm hoping to read are

1. An Empirical Analysis of Issue Templates Usage in Large-Scale Projects on GitHub

2. Consistent or not? An investigation of using Pull Request Template in GitHub

3. To Follow or Not to Follow: Understanding Issue/Pull-Request Templates on GitHub

4. An empirical examination of newcomer contribution costs in established OSS communities: a knowledge-based perspective

1 Other questions

1. How does task distribution efficiency change? How does task choice change?

2. What differences do I observe b/w corporate and non-corporate OSS?

3. Control for software project maturity, individual length of affiliation with project

4. What % of assignees end up writing code/participating in discussion?

5.