When dispersed teams are more successful: Evidence from software

Gábor Békés, Julian Hinz, Miklós Koren, and Aaron Lohmann

WU Vienna

2025-06-18

Research questions

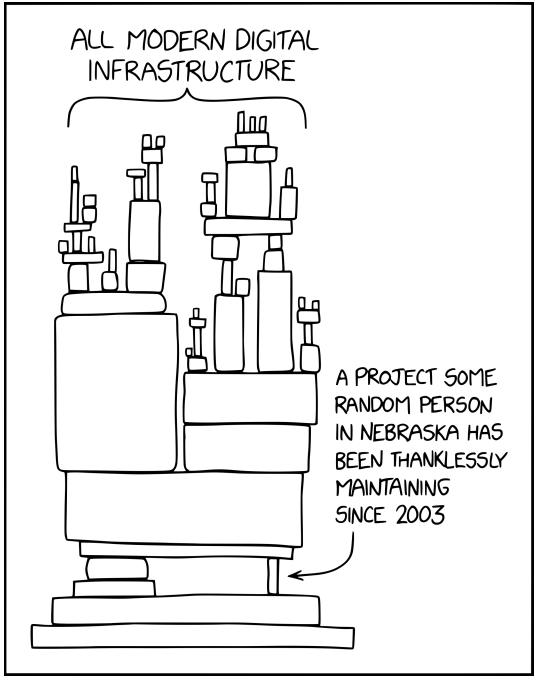
1. How and where is open source software developed?
2. Do spatially dispersed developers produce quality software?
3. How do team formation and collaboration affect the success of OSS projects?

Research questions

1. How and where is open source software developed?
2. Do spatially dispersed developers produce quality software?
3. How do team formation and collaboration affect the success of OSS projects?

Economics of open source

- Incentives to produce and contribute to open source
- Spatial dispersion and its interactions with quality


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

Open Source Software

- Software is everywhere and more specifically OSS is everywhere
  - 98% of commercial software uses OSS (Synopsys 2023).
  - OSS is a global phenomenon

Open Source Software

- Software is everywhere and more specifically OSS is everywhere
  - 98% of commercial software uses OSS (Synopsys 2023).
  - OSS is a global phenomenon
- OSS is huge
  - Hoffmann, Nagle, and Zhou (2024) estimate demand side as 8.8 trilion USD;
  - GitHub nowadays has over 100 million developers

Open Source Software

OSS plays a hugely important roles in

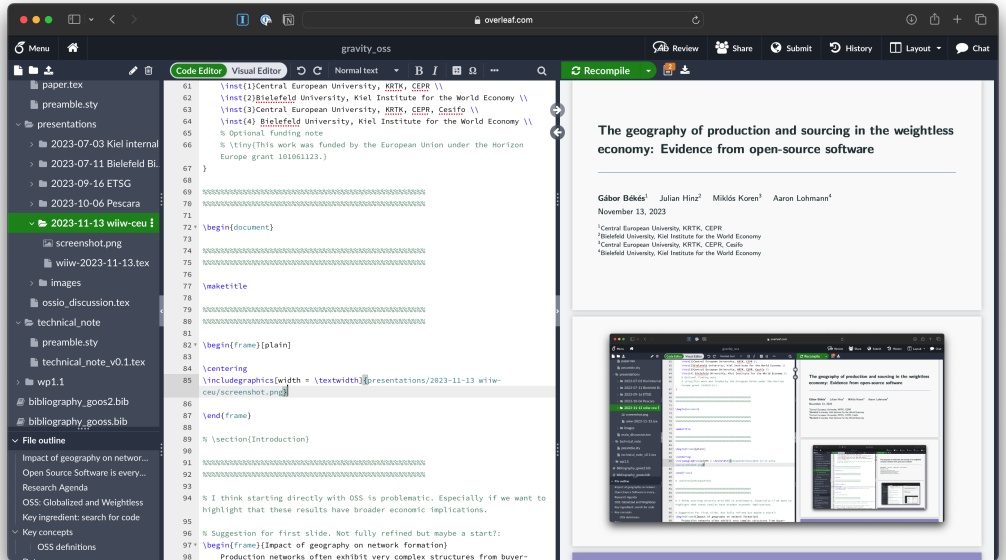
- Websites (JavaScript)
- Operating systems (Linux, Android)
- Data Science (R Tidyverse, Python Pandas, Julia)
- Machine Learning and AI (PyTorch, LLaMA)

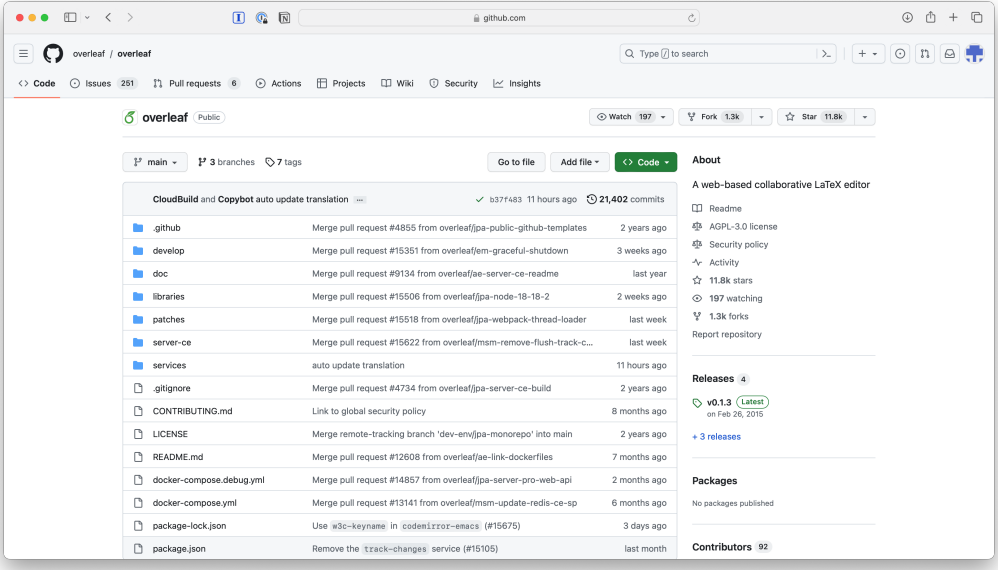
Open Source Software

- OSS is observable
  - Due to the `git` paradigm almost everything is recorded!

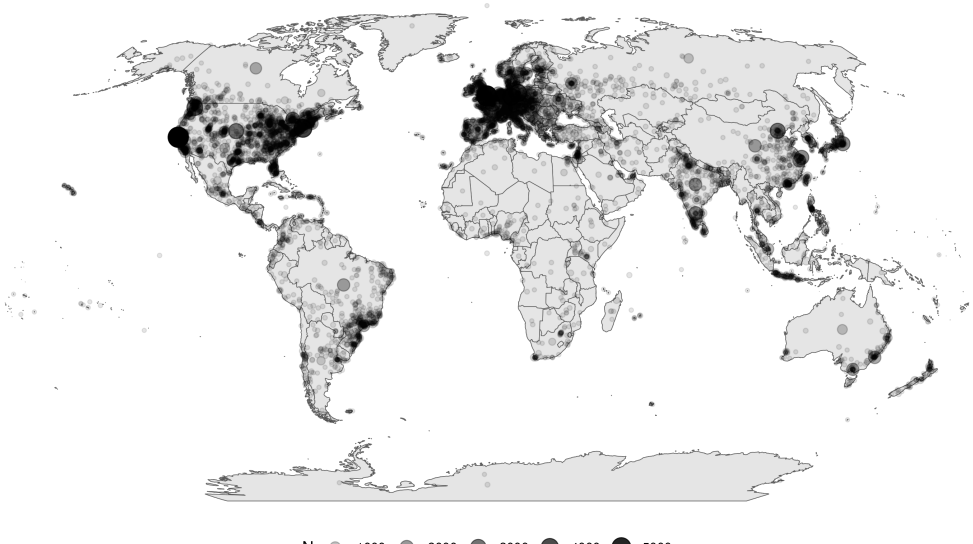
Open Source Software

- OSS is observable
  - Due to the `git` paradigm almost everything is recorded!
- OSS mostly free, but present in fee-based platforms
  - Overleaf, ...

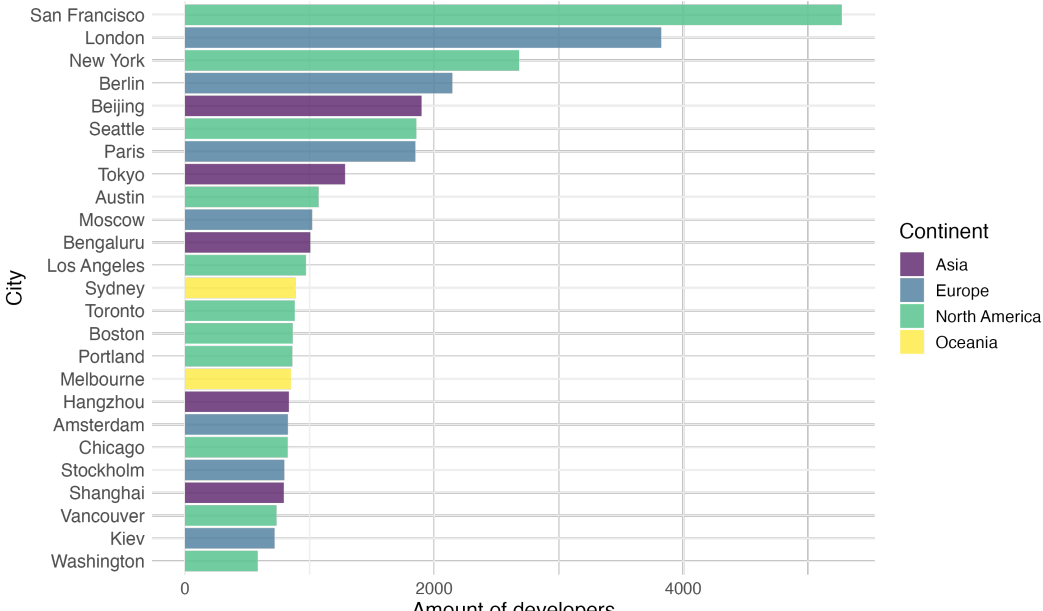

overleaf.com
Menu
gravity_oss
Review
Share
Submit
History
Layout
Chat
Code Editor
Visual Editor
Normal text
Recompile
paper.tex
preamble.sty
presentations
2023-07-03 Kiel internal
2023-07-11 Bielefeld Bi.
2023-09-16 ETSG
2023-10-06 Pescara
2023-11-13 wiiw-ceu
screenshot.png
wiiw-2023-11-13.tex
images
ossio_discussion.tex
technical_note
preamble.sty
technical_note_v0.1.tex
wp1.1
bibliography_goos2.bib
bibliography_goos.bib
File outline
Impact of geography on networ...
Open Source Software is every...
Research Agenda
OSS: Globalized and Weightless
Key ingredient: search for code
Key concepts
OSS definitions
\inst{1}Central European University, KRTK, CEPR \\
\inst{2}Bielefeld University, Kiel Institute for the World Economy \\
\inst{3}Central European University, KRTK, CEPR, CesIfo \\
\inst{4} Bielefeld University, Kiel Institute for the World Economy \\
% Optional funding note
% \tiny{This work was funded by the European Union under the Horizon Europe grant 101061123.}
}
\begin{document}
\maketitle
\begin{frame}[plain]
\centering
\includegraphics[width = \textwidth]{presentations/2023-11-13 wiiw-ceu/screenshot.png}
\end{frame}
% \section{Introduction}
% I think starting directly with OSS is problematic. Especially if we want to highlight that these results have broader economic implications.
% Suggestion for first slide. Not fully refined but maybe a start?:
\begin{frame}{Impact of geography on network formation}
Production networks often exhibit very complex structures from buyer-
The geography of production and sourcing in the weightless economy: Evidence from open-source software
Gábor Békés[1] Julian Hinz[2] Miklós Koren[3] Aaron Lohmann[4]
November 13, 2023
[1]Central European University, KRTK, CEPR
[2]Bielefeld University, Kiel Institute for the World Economy
[3]Central European University, KRTK, CEPR, CesIfo
[4]Bielefeld University, Kiel Institute for the World Economy


github.com
overleaf / overleaf
Type / to search
Code
Issues 251
Pull requests 6
Actions
Projects
Wiki
Security
Insights
overleaf
Public
Watch 197
Fork 1.3k
Star 11.8k
main
3 branches
7 tags
Go to file
Add file
Code
CloudBuild and Copybot auto update translation
b37f483 11 hours ago
21,402 commits
.github Merge pull request #4855 from overleaf/jpa-public-github-templates 2 years ago
develop Merge pull request #15351 from overleaf/em-graceful-shutdown 3 weeks ago
doc Merge pull request #9134 from overleaf/ae-server-ce-readme last year
libraries Merge pull request #15506 from overleaf/jpa-node-18-18-2 2 weeks ago
patches Merge pull request #15518 from overleaf/jpa-webpack-thread-loader last week
server-ce Merge pull request #15622 from overleaf/msm-remove-flush-track-c... last week
services auto update translation 11 hours ago
.gitignore Merge pull request #4734 from overleaf/jpa-server-ce-build 2 years ago
CONTRIBUTING.md Link to global security policy 8 months ago
LICENSE Merge remote-tracking branch 'dev-env/jpa-monorepo' into main 2 years ago
README.md Merge pull request #12608 from overleaf/ae-link-dockerfiles 7 months ago
docker-compose.debug.yml Merge pull request #14857 from overleaf/jpa-server-pro-web-api 2 months ago
docker-compose.yml Merge pull request #13141 from overleaf/msm-update-redis-ce-sp 6 months ago
package-lock.json Use w3c-keyname in codemirror-emacs (#15675) 3 days ago
package.json Remove the track-changes service (#15105) last month
About
A web-based collaborative LaTeX editor
Readme
AGPL-3.0 license
Security policy
Activity
11.8k stars
197 watching
1.3k forks
Report repository
Releases 4
v0.1.3 Latest
on Feb 26, 2015
+ 3 releases
Packages
No packages published
Contributors 92

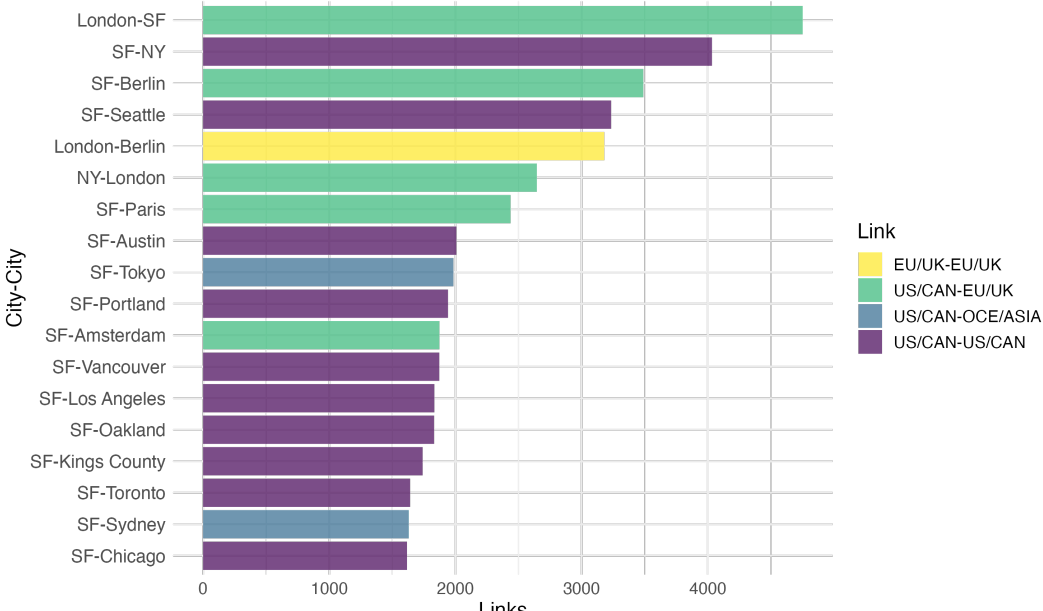
Global industry: Number of JavaScript developer per city



Dispersion and concentration: top cities per number of developers



Collaboration across space



Git and GitHub

- Git is a version control system, allowing to track changes in code.
- GitHub is a platform for hosting Git repositories, allowing collaboration and sharing of code.
- GitHub has over 100 million developers and hosts over 400 million repositories.

Open Source vocabulary

`Package`: A unit of software, provision of a (bundle of) functionality

`Project`: A software project offering solution to a use case. Typically one package, but may be more.

`Repository`: A storage for one project (what we observe)

`Commit`: The smallest unit of contribution

`Dependency`: An imported package that provides a functionality

What we see in the data: ggplot2-project as an example



Figure 1: Hadley Wickham

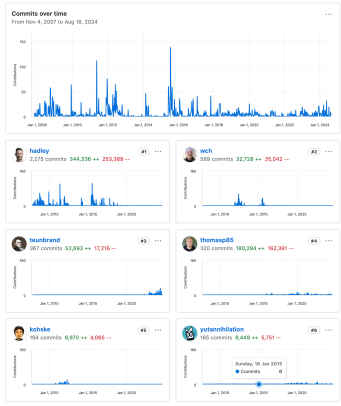


Figure 2: Commits in ggplot2

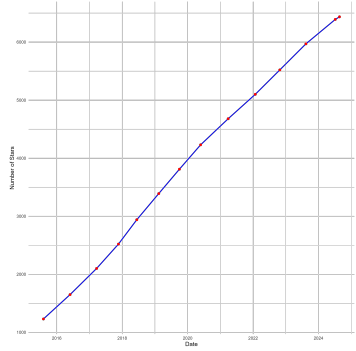


Figure 3: ggplot2 stars over time

Team formation follows gravity

Dependent Variable:	Collaboration count	
Model:	(1)	(2)
Variables		
Same Country	1.837***	0.9497***
	(0.0522)	(0.0878)
Log Distance	-0.8720***	-0.8634***
	(0.0118)	(0.0119)
Common Official Language		1.000***
		(0.0764)
Common Religion		0.8163***
		(0.1077)
Fixed-effects		
Developer 1 City	Yes	Yes
Developer 2 City	Yes	Yes
Fit statistics		
Observations	21,687,649	21,242,884
Squared Correlation	0.96379	0.96530
Pseudo R^2	0.90664	0.90990
BIC	4,566,297.8	4,378,415.1

Clustered (Developer 1 City) standard-errors in parentheses
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Good software is produced in teams

Dependent Variable:	num_stars				
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
log_dev_1_exp	0.1035***	0.0530***	0.0140*	0.0164**	0.0258
	(0.0088)	(0.0081)	(0.0078)	(0.0078)	(0.0224)
log_dev_2_exp	0.1561***	0.1017***	0.0992***	0.0997***	0.0999***
	(0.0072)	(0.0062)	(0.0060)	(0.0060)	(0.0145)
log_distance		0.2678***	0.2475***	0.2557***	0.2339***
		(0.0066)	(0.0066)	(0.0066)	(0.0121)
log_dev_1_effort			0.5487***		
			(0.0170)		
log_dev_2_effort			-0.4102***		
			(0.0183)		
log_commits_6m				0.3066***	0.3075***
				(0.0187)	(0.0314)
dev_1_share				2.598***	2.305***
				(0.1296)	(0.2255)
Fixed-effects					
language_time_fe	Yes	Yes	Yes	Yes	Yes
Fit statistics					
Observations	1,637,399	1,636,722	1,636,722	1,636,722	614,681
Squared Correlation	0.00687	0.01269	0.02378	0.02028	0.02549
Pseudo R^2	0.14658	0.20801	0.25819	0.24941	0.26197
BIC	22,526,679.5	20,904,939.1	19,584,430.7	19,815,448.8	7,717,733.2

Clustered (language_time_fe) standard-errors in parentheses
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Literature

- **Production in teams:** Jarosch, Oberfield, and Rossi-Hansberg (2021) ; Herkenhoff et al. (2024) ; Freund (2022) ; Kerr and Kerr (2018)
  Our contribution: A model for global team formation which has selection as a main mechanism.
- **Gravity/International Trade:** Eaton and Kortum (2002) ; Atkin, Chen, and Popov (2022) ; Head, Li, and Minondo (2019)
  Our contribution: Gravity estimates for team formation in OSS.
- **OSS**: Lerner and Tirole (2002) ; Fackler and Laurentsyeva (2020) ; Wachs et al. (2022)
  Our contribution: Providing more descriptive statistics, making use of data and combining several data sources.

This paper

This paper

- Model global *team formation*, *collaboration*, and *performance* for open-source software projects.

This paper

- Model global *team formation*, *collaboration*, and *performance* for open-source software projects.
- Test model predictions using data from GitHub 2018-2024.

This paper

- Model global *team formation*, *collaboration*, and *performance* for open-source software projects.
- Test model predictions using data from GitHub 2018-2024.
- (later) Structurally estimate the model. Conduct counterfactuals about digital fragmentation, AI, and the Global South.

Team formation, collaboration, and performance

Timing

1. Developers have an idea (uncertain quality)
2. Two developers meet at random (incomplete information)
3. Decide whether to do a project together
4. Software quality depends on ideas and effort
5. Users download it, distributing kudos to developers

Production function

Software quality depends on *design* z and *implementation effort* e:

$$y_p = z_p^{\nu} e_1^{1-\nu}$$

Diminishing returns to implementation effort for a given design $\rightarrow$ rents to good design.

Benchmark 1: Solo project

- Developers care about software quality and receive "kudos" proportional to quality
- Simple case with linear cost $c_1 e_1$:

$$\max_e z_p^\nu e_1^{1-\nu} - c_1 e_1$$

Benchmark 1: Solo project

- Developers care about software quality and receive "kudos" proportional to quality
- Simple case with linear cost $c_1 e_1$:

$$\max_e z_p^\nu e_1^{1-\nu} - c_1 e_1$$

leads to optimal effort

$$e_1 = z_p c_1^{-1/\nu} (1-\nu)^{1/\nu}$$

Benchmark 1: Solo project

- Developers care about software quality and receive "kudos" proportional to quality
- Simple case with linear cost $c_1 e_1$:

$$\max_e z_p^\nu e_1^{1-\nu} - c_1 e_1$$

leads to optimal effort

$$e_1 = z_p c_1^{-1/\nu} (1-\nu)^{1/\nu}$$

and project quality

$$y_p = z_p c_1^{-(1-\nu)/\nu} (1-\nu)^{(1-\nu)/\nu}$$

Benchmark 1: Solo project

- Developers care about software quality and receive "kudos" proportional to quality
- Simple case with linear cost $c_1 e_1$:

$$\max_e z_p^\nu e_1^{1-\nu} - c_1 e_1$$

leads to optimal effort

$$e_1 = z_p c_1^{-1/\nu} (1-\nu)^{1/\nu}$$

and project quality

$$y_p = z_p c_1^{-(1-\nu)/\nu} (1-\nu)^{(1-\nu)/\nu}$$

- Effort is increasing in design quality and decreasing in coding costs

Benchmark 2: Market-based collaboration

- Hire another developer, paying her effort costs, to maximize

$$z_p^{\nu}(e_{1p} + e_{2p})^{1-\nu} - c_{2p}e_{2p}$$

Benchmark 2: Market-based collaboration

- Hire another developer, paying her effort costs, to maximize

$$z_p^{\nu}(e_{1p} + e_{2p})^{1-\nu} - c_{2p}e_{2p}$$

- Invite developer 2 only if her effective cost is lower: $\tau_{2p}c_2 < c_1$.
- Remote collaborators often face higher costs $\tau_{2p} > 1$ due to frictions.

Benchmark 2: Market-based collaboration

- Hire another developer, paying her effort costs, to maximize

$$z_p^{\nu}(e_{1p} + e_{2p})^{1-\nu} - c_{2p}e_{2p}$$

- Invite developer 2 only if her effective cost is lower: $\tau_{2p}c_2 < c_1$.
- Remote collaborators often face higher costs $\tau_{2p} > 1$ due to frictions.

- Working in a team may be better. Developer 1 can increase the rents to her design.
- Good model for in-house, proprietary software.
- Hard in OSS, where no money is involved and effort is not contractible.

Collaborative development

- Design quality improves when the best idea prevails:

$$z_p = \max\{z_1, z_2\}$$

- assume developer 2 can be contracted to provide exactly e_2 effort for price $\tau_{2p} c_2 e_2$.
- lead developer 1 maximizes

$$\max_{e_2} z_p^{\nu} (e_1 + e_2)^{1-\nu} - \tau_{2p} c_2 e_2$$

Collaborative development

- optimal effort of developer 2 from point of view of dev 1 is then

$$e_2 = z_p^{\nu}(\tau_{2p} c_2)^{-1/\nu}(1-\nu)^{1/\nu} - e_1$$

Collaborative development

- optimal effort of developer 2 from point of view of dev 1 is then

$$e_2 = z_p^{\nu}(\tau_{2p}c_2)^{-1/\nu}(1-\nu)^{1/\nu} - e_1$$

- Invite a second developer when her idea is strong enough to offset higher cost:

$$\frac{z_2}{z_1} > \left(\frac{\tau_{2p}c_2}{c_1}\right)^{1/\nu}$$

Collaborative development

- rents to developer 1 increase for two reasons:

Collaborative development

- rents to developer 1 increase for two reasons:

1. only pay developer 2 for marginal cost

Collaborative development

- rents to developer 1 increase for two reasons:

1. only pay developer 2 for marginal cost
2. new design makes code of developer 1 more valuable

Collaborative development

- rents to developer 1 increase for two reasons:

1. only pay developer 2 for marginal cost
2. new design makes code of developer 1 more valuable

- Because of higher rent, developer 1 willing to engage in rent sharing
- Developer 1 (owner of project) can offer fraction $\phi \in (0, 1)$ of rents to developer 2

Rent sharing

- if both developer 1 and 2 know quality of design and agree on rent sharing before, then first-order-conditions for effort:

$$(1-\nu)(1-\phi)z_p^{\nu}(e_1+e_2)^{-\nu} = c_1$$
$$(1-\nu)\phi z_p^{\nu}(e_1+e_2)^{-\nu} = \tau_{2p}c_2$$

Effort cost assumptions

We analyze two different cost functions:

1. **Linear cost:** $c_{ip}e_{ip}$.
2. **Convex cost:** $\tau_{ip}c_i e_{ip}^{1+\gamma}/(1+\gamma)$

The convex specification implies increasing marginal cost

$$\frac{\partial c_{ip}}{\partial e_{ip}} = \tau_{ip}c_i e_{ip}^{\gamma}$$

with $\gamma > 0$, which avoids corner solutions and leads to interior optima.

Benefits of collaboration

1. Decaying effort costs: getting tired, bored, diminishing returns to specialized expertise.
2. Complementarity of skills (not modeled): developers specialize in different parts of the code.
3. Knowledge sharing: design quality may improve.

Kudos sharing and participation constraints

$$\max_{e_1} \ (1-\phi) z_p^{\nu} (e_1+e_2)^{1-\nu} - c_1 \frac{e_1^{1+\gamma}}{1+\gamma} \quad \geq \kappa_{1p}$$

$$\max_{e_2} \ \phi z_p^{\nu} (e_1+e_2)^{1-\nu} - c_2 \frac{e_2^{1+\gamma}}{1+\gamma} \quad \geq \kappa_{2p}$$

with both developers earning more than the outside option κ_{ip}.

Frictions in collaboration

Communication

Hard to work on someone else's code without frequent communication (may depend on distance):

$$c_{2p} := \tau_{12} c_2$$

Kudos value

Distant developers may enjoy less of the kudos (e.g., cannot turn kudos into job opportunities, may depend on distance)

$$\kappa_{2p} := d_{12} \kappa_2$$

Equilibrium team formation

Given kudos sharing rules and frictions, team formation satisfies participation constraints

$$\mathsf{E}_1[(1-\phi)y_p^*] \geq y_{1p}^*$$
$$\mathsf{E}_2(\phi y_p^*) \geq d_{12}\tilde{\kappa}_2$$

with $\tilde{\kappa}_2 = \kappa_2(1+\gamma)/(\gamma+\nu)$.

Distance frictions to team formation

Far-away developers less likely to join project:

- less expected rent due to miscommunication / lower effort
- higher outside option due to rent appropriation frictions

Selection of projects by quality

Only better projects get produced.

Selection of developers by distance

Remote developers (higher d_{12}) have to expect

- higher project quality or
- higher kudos share to join.

Kudos sharing rule: Credit where credit due

$$z_p := \max\{z_1, z_2\}$$

Developer with the winning design gets higher share:

$$\phi := \begin{cases} \phi_H & \text{if } z_2 \geq z_1 \\ \phi_L & \text{if } z_2 < z_1 \end{cases}$$

with $\phi_H > \frac{1}{2} > \phi_L$

Good developers have *disproportionately* higher expected payoff from collaborative projects $\rightarrow$ positive assortative matching.

Testable predictions

Testable predictions

Testable predictions

Team formation

1. More skilled developers more likely to join collaborative projects.
2. Distant developers less likely to join collaborative projects.
3. Distant developers on collaborative projects have higher skill.
4. (if $\phi_H \gg \phi_L$) Positive assortative matching of developers by skill.

Testable predictions

Team formation

1. More skilled developers more likely to join collaborative projects.
2. Distant developers less likely to join collaborative projects.
3. Distant developers on collaborative projects have higher skill.
4. (if $\phi_H \gg \phi_L$) Positive assortative matching of developers by skill.

Collaboration

5. Collaborative projects more successful than solo projects.
6. Lead developer works more on project than follower.
7. Skilled developers work more.
8. Followers work more on project with a skilled leader.
9. Distant followers work less.

Testable predictions

Team formation

1. More skilled developers more likely to join collaborative projects.
2. Distant developers less likely to join collaborative projects.
3. Distant developers on collaborative projects have higher skill.
4. (if $\phi_H \gg \phi_L$) Positive assortative matching of developers by skill.

Collaboration

5. Collaborative projects more successful than solo projects.
6. Lead developer works more on project than follower.
7. Skilled developers work more.
8. Followers work more on project with a skilled leader.
9. Distant followers work less.

Project success

10. Skill and effort both matter for project success.
11. Projects with distant developers are more successful.

Measurement

Lead developer

- Person starting the project
- If developers join in same month, person with most commits in the first month

Skill

- Experience: Number of commits before project start

Quality

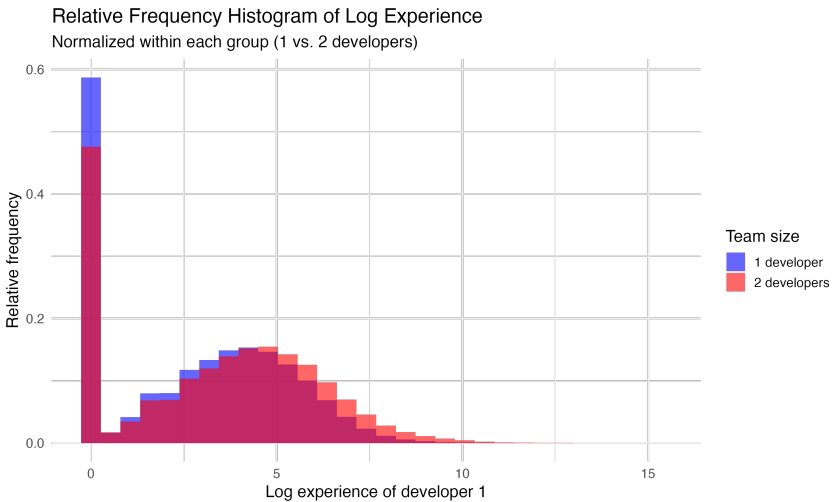
- Number of stars after 6 months
- (Number of forks, number of downloads)

Effort

- Number of commits in first 6 months

Selection into collaboration

More skilled developers are overrepresented in collaborative projects.

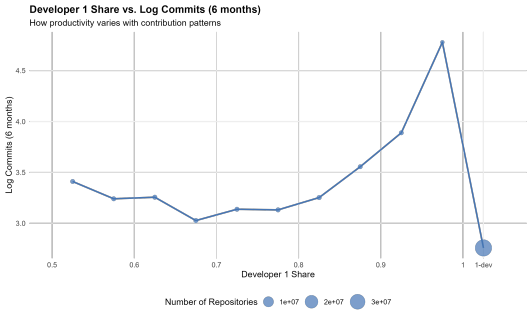


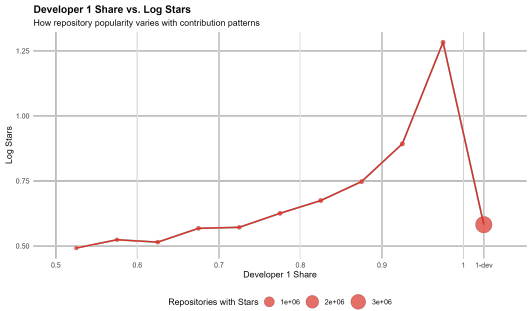
More experienced developers are joined by more experienced developers

Dependent Variable:	log_dev_2_exp					
Model:	(1)	(2)	(3)	(4)	(5)	(6)
Variables						
log_dev_1_exp	0.1079***	0.0896***	0.0897***	0.0809***	0.0804***	0.3108***
	(0.0001)	(0.0009)	(0.0009)	(0.0009)	(0.0009)	(0.0017)
log_commits_6m			0.1143***	0.1300***	0.1323***	0.1326***
			(0.0025)	(0.0024)	(0.0024)	(0.0046)
log_dev_1_effort			-0.1000***	-0.1142***	-0.1159***	-0.1275***
			(0.0024)	(0.0023)	(0.0023)	(0.0044)
log_distance				0.0362***	0.0488***	-0.0042***
				(0.0004)	(0.0006)	(0.0007)
dev_1_to_dev_2_same_city					0.0808***	
					(0.0029)	
Fixed-effects						
language_time_fe		Yes	Yes	Yes	Yes	Yes
Fit statistics						
Observations	2,340,003	2,339,777	2,339,777	2,338,828	2,338,828	862,462
Squared Correlation	0.11041	0.14524	0.14695	0.15897	0.15917	0.28787
Pseudo R^2	0.04672	0.06355	0.06411	0.06988	0.07010	0.17988
BIC	11,109,367.1	11,008,745.1	11,002,287.6	10,930,464.8	10,927,942.3	3,009,542.6

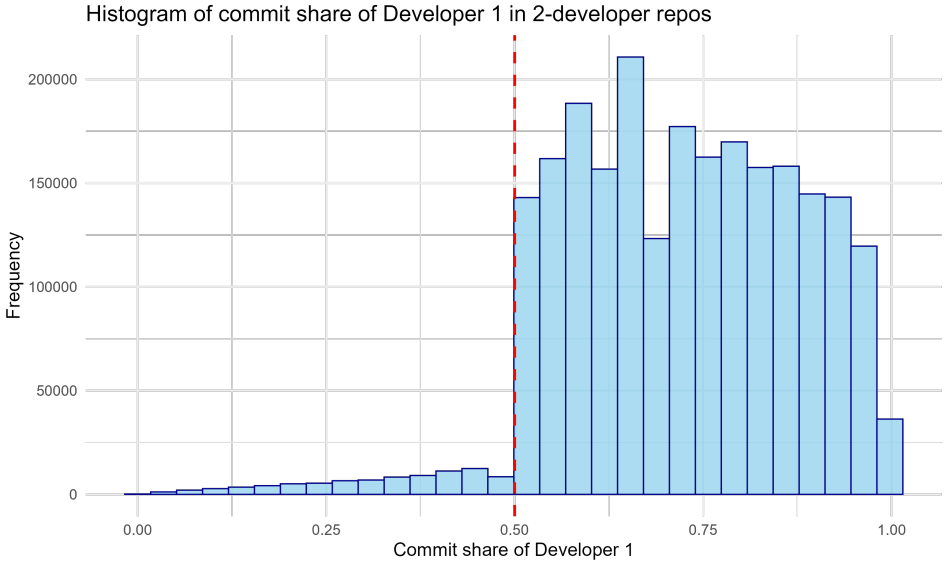
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Collaborative projects with strong lead developers are the most successful





Lead developers commit much more



Dependent Variable:			num_stars		
Model:	(1)	(2)	(3)	(4)	(5)
Variables					
log_dev_1_exp	0.1035***	0.0530***	0.0140*	0.0164**	0.0258
	(0.0088)	(0.0081)	(0.0078)	(0.0078)	(0.0224)
log_dev_2_exp	0.1561***	0.1017***	0.0992***	0.0997***	0.0999***
	(0.0072)	(0.0062)	(0.0060)	(0.0060)	(0.0145)
log_distance		0.2678***	0.2475***	0.2557***	0.2339***
		(0.0066)	(0.0066)	(0.0066)	(0.0121)
log_dev_1_effort			0.5487***		
			(0.0170)		
log_dev_2_effort			-0.4102***		
			(0.0183)		
log_commits_6m				0.3066***	0.3075***
				(0.0187)	(0.0314)
dev_1_share				2.598***	2.305***
				(0.1296)	(0.2255)
Fixed-effects					
language_time_fe	Yes	Yes	Yes	Yes	Yes
Fit statistics					
Observations	1,637,399	1,636,722	1,636,722	1,636,722	614,681
Squared Correlation	0.00687	0.01269	0.02378	0.02028	0.02549
Pseudo R^2	0.14658	0.20801	0.25819	0.24941	0.26197
BIC	22,526,679.5	20,904,939.1	19,584,430.7	19,815,448.8	7,717,733.2

Clustered (language_time_fe) standard-errors in parentheses
*Signif. Codes: ***: 0.01, **: 0.05, *: 0.1*

Counterfactuals

Counterfactuals (for future work)

Estimate γ, ν, ϕ_H, ϕ_L, τ_{12} and d_{12} structurally.

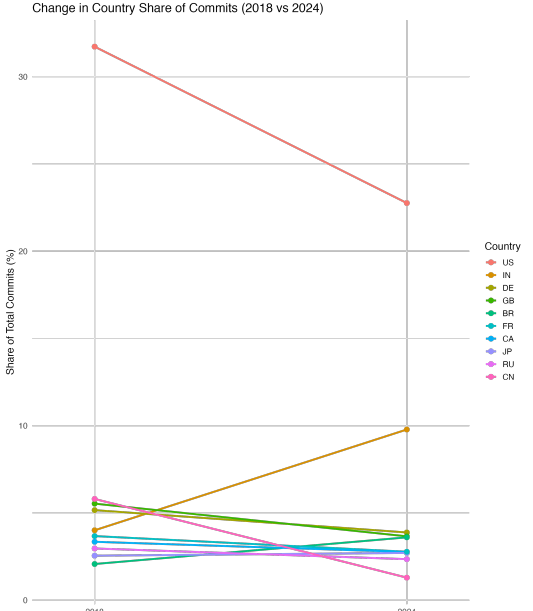
Study the impact of

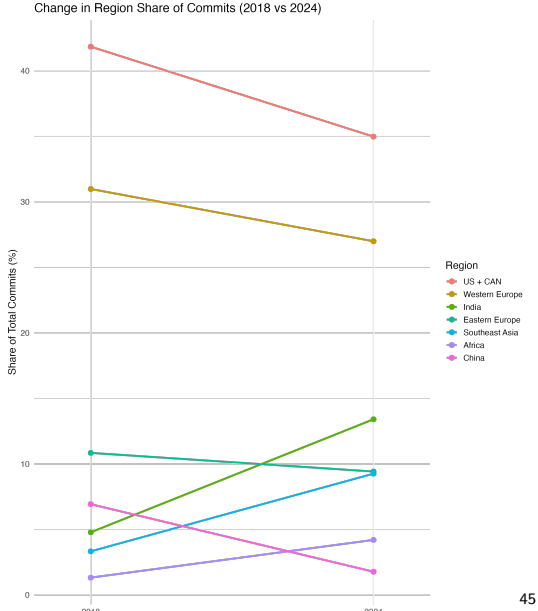
1. frictions in digitial services
2. generative AI

on

1. global software output
2. regional distribution of developers
3. inequality among developers

The Global South is increasingly important





Conclusion

Conclusion

Atkin, David, M Keith Chen, and Anton Popov. 2022. "The Returns to Face-to-Face Interactions: Knowledge Spillovers in Silicon Valley." National Bureau of Economic Research.

Eaton, Jonathan, and Samuel Kortum. 2002. "Technology, Geography, and Trade." *Econometrica* 70 (5): 1741–79.

Fackler, Thomas, and Nadzeya Laurentsyeva. 2020. "Gravity in Online Collaborations: Evidence from Github." In *CESifo Forum*, 21:15–20. 03. München: ifo Institut-Leibniz-Institut für Wirtschaftsforschung an der ….

Freund, Lukas. 2022. "Superstar Teams: The Micro Origins and Macro Implications of Coworker Complementarities." *Available at SSRN 4312245*.

Head, Keith, Yao Amber Li, and Asier Minondo. 2019. "Geography, Ties, and Knowledge Flows: Evidence from Citations in Mathematics." *Review of Economics and Statistics* 101 (4): 713–27.

Herkenhoff, Kyle, Jeremy Lise, Guido Menzio, and Gordon M Phillips. 2024. "Production and Learning in Teams." *Econometrica* 92 (2): 467–504.