How does Division of Labor Affect Team Productivity? Evidence from GitHub

Jinci Liu

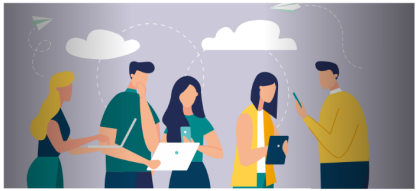
IIES, Stockholm University

March 12, 2025

Motivation

- Large differences in organizational structure between/within firms (Bloom et al., 2013, 2014; Giroud et al., 2022)
- Given a set of workers and tasks, how to allocate them?
- Seminal theory: **division of labor increases productivity** (Smith, 1776)
  - To make a pin, divide into 18 distinct operations, each carried out by different people
  - Gain: human capital accumulation (Rosen, 1983; Becker and Murphy, 1992; Young, 1928)

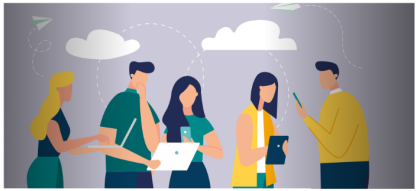
Routine versus Non-routine Production



- Explicit rules & repeat procedures Autor et al. (2003)
- Problem-solving & complex communication

Autor et al. (2003)

Routine versus Non-routine Production

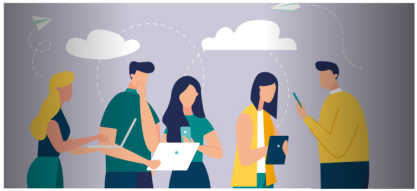


- Explicit rules & repeat procedures Autor et al. (2003)
- Empirical evidence of specialization benefits
  - Gong and Png (2024): **cashiers**
  - Kohlhepp (2024): **hair salon**

- Problem-solving & complex communication

  Autor et al. (2003)

Routine versus Non-routine Production



- Explicit rules & repeat procedures Autor et al. (2003)
- Empirical evidence of specialization benefits
  - Gong and Png (2024): **cashiers**
  - Kohlhepp (2024): **hair salon**

- Problem-solving & complex communication

  Autor et al. (2003)
- Cross-task feedback and knowledge sharing are important

Research Q: How does Team Specialization Affect Productivity?

- Effect of specialization on team productivity is ex-ante **ambiguous**
  - Increase member's task-specific human capital
  - Suffer from coordination cost Becker and Murphy (1992) and learning myopia Levinthal and March (1993)
- This project:
  - **New data**: millions of task assignments for software developers
  - **New measures**: task allocation specialization and productivity
  - **New result**: specialization is detrimental for software development team productivity

Roadmap

1. Data
2. Measures
3. Facts
4. Empirical
5. Conclusion

Data Construction

- Use data from the **largest online coding platform** GitHub
- **64,400** software development **teams** (public repository) under firms (organization) Detail
  - E.g., Teams in Microsoft, Meta, Google
- **35 million** code files Detail
  - Unsupervised learning algorithm to classify into **10 task types** (E.g., frontend, backend) Detail
- **292,840** team members (defined by GitHub) Detail
- **2017-2023**

Measuring Team Specialization

Example

- Each member has inelastic 1 unit of labor supply (row sum)
- Each task requires a different unit of labor supply (column sum: **task share**)

Task

	1	2	3	
A	1/2	0	1/2	1
B	1/2	1/2	0	1
C	1/2	0	1/2	1
	3/2	**1/2**	**1**	

Actual(A)

Example

- Each member has inelastic 1 unit of labor supply (row sum)
- Each task requires a different unit of labor supply (column sum: **task share**)

Task

	1	2	3	
A	1/2	0	1/2	1
B	1/2	1/2	0	1
C	1/2	0	1/2	1
	3/2	**1/2**	**1**	

Actual(A)

Task

	1	2	3	
A	1/2	1/6	1/3	1
B	1/2	1/6	1/3	1
C	1/2	1/6	1/3	1
	3/2	**1/2**	**1**	

Generalized(G)

Example

- Each member has inelastic 1 unit of labor supply (row sum)
- Each task requires a different unit of labor supply (column sum: **task share**)

Task

	1	2	3	
A	1/2	0	1/2	1
B	1/2	1/2	0	1
C	1/2	0	1/2	1
	3/2	**1/2**	**1**	

Actual(A)

Task

	1	2	3	
A	1/2	1/6	1/3	1
B	1/2	1/6	1/3	1
C	1/2	1/6	1/3	1
	3/2	**1/2**	**1**	

Generalized(G)

Task

	1	2	3	
A	1	0	0	1
B	1/2	1/2	0	1
C	0	0	1	1
	3/2	**1/2**	**1**	

Specialized(S)

Example

- Each member has inelastic 1 unit of labor supply (row sum)
- Each task requires a different unit of labor supply (column sum: **task share**)

Task

	1	2	3	
A	1/2	0	1/2	1
B	1/2	1/2	0	1
C	1/2	0	1/2	1
	3/2	**1/2**	**1**	

Actual(A)

Task

	1	2	3	
A	1/2	1/6	1/3	1
B	1/2	1/6	1/3	1
C	1/2	1/6	1/3	1
	3/2	**1/2**	**1**	

Generalized(G)

Task

	1	2	3	
A	1	0	0	1
B	1/2	1/2	0	1
C	0	0	1	1
	3/2	**1/2**	**1**	

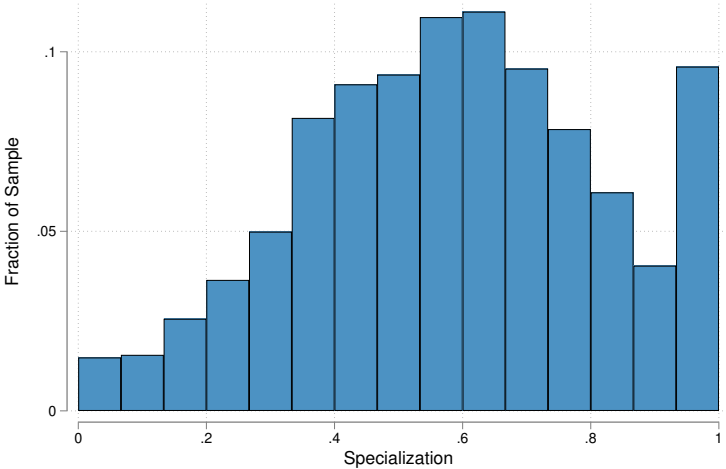
Specialized(S)

- Define team specialization index *SPE*: $\frac{d(A,G)}{d(S,G)}$ (d: Euclidean distance)
- *SPE* is **higher (more specialized)** if A is far from G

▸ Math formula ◂ Distance between two matrices

Distribution of Team Specialization Index

Team-month level



▸ By Team Size ▸ Over Time

Measuring Productivity

Outcome Measures

Team-month level

- **Output Quality**: stars per month (Users' Appreciation) (Borges and Valente, 2018)
  - 75% developers check stars before using Link
  - Stars have monetary value ≈$0.88/star Link
- **Output Quantity**: lines-of-code (Vasilescu et al., 2015),(Casalnuovo et al., 2015),(Wagner and Ruhe, 2018)
- **Problem-Solving Speed**: time from users' bug report to solve Example
- **Code Acceptance Rate**: success rate of member code submissions Manage to merge Fail to merge
- **Discussion**: comments sent by team members

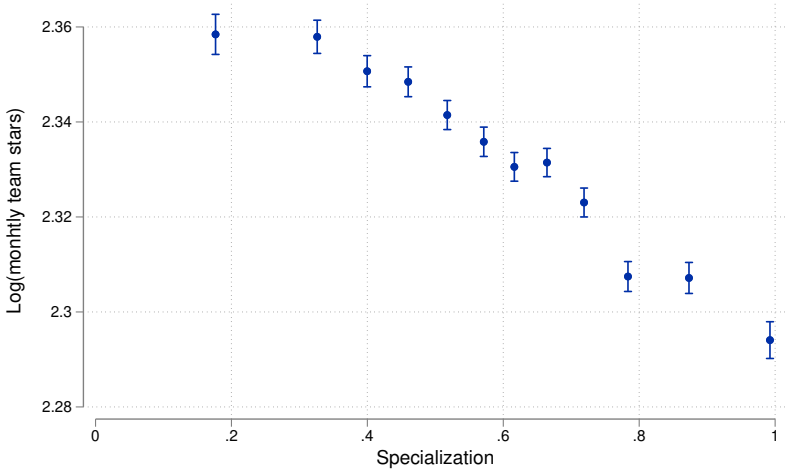
Summary statistics

Team Specialization and Productivity

$$Y_{mt} = \beta_1 SPE_{mt} + \theta_i + \gamma_m + \phi_{mt} + \beta_2 \mathbf{X}_{mt} + \varepsilon_{mit}$$

- Y_{mt}: outcome for team m in month t
- SPE_{mt}: degree of specialization for team m in t
- β_1: coefficient of interest
- θ_i: member fixed effect
- γ_m: team fixed effect
- ϕ_{mt}: team age, team size fixed effect
- X_{mt}: 10 task type distribution
- Standard error is clustered at team level
- Weight by 1/team size

Fact 1 Higher Specialization, Lower Quality



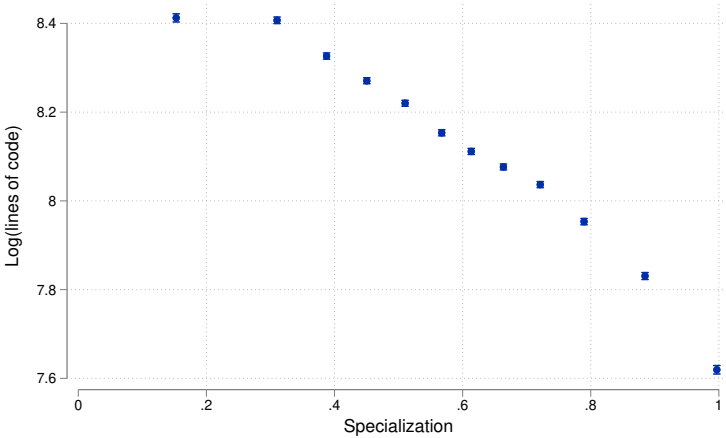
Controls: task distribution
Fixed effect: team age, team size, team, member

Output Quality

Dep Var: Log (stars per month)	(1)	(2)	(3)
SPE_{mt}	-0.320***	-0.078***	-0.086***
	(0.021)	(0.007)	(0.007)
Dependent mean	2.13	2.13	2.13
R-squared	0.648	0.903	0.910
Observations	1,823,750	1,823,750	1,770,310
Task type share control	Y	Y	Y
Team age FE	Y	Y	Y
Team size FE	Y	Y	Y
Firm FE	Y		
Team FE		Y	Y
Member FE			Y

NOTE. Weighted by 1/team size.

Fact 2 Higher specialization, Lower Quantity

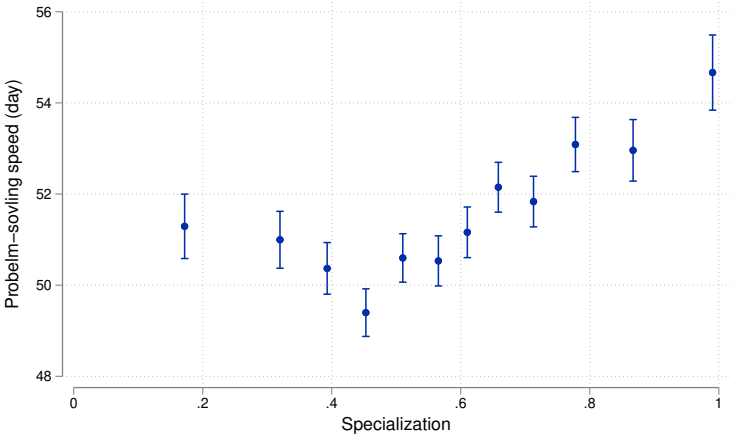


Controls: task distribution
Fixed effect: team age, team size, team, member

▸ Regression

Fact 3 Slower to Solve Users' Problems

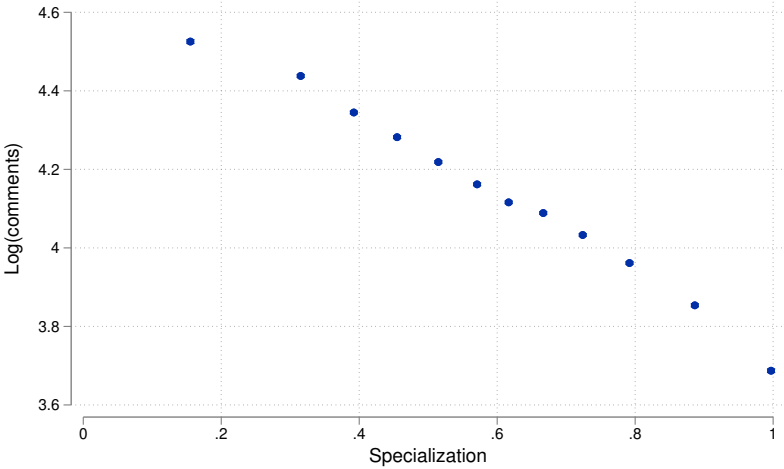
Cross-task knowledge



Controls: task distribution
Fixed effect: team age, team size, #issues, team, member

▸ Regression

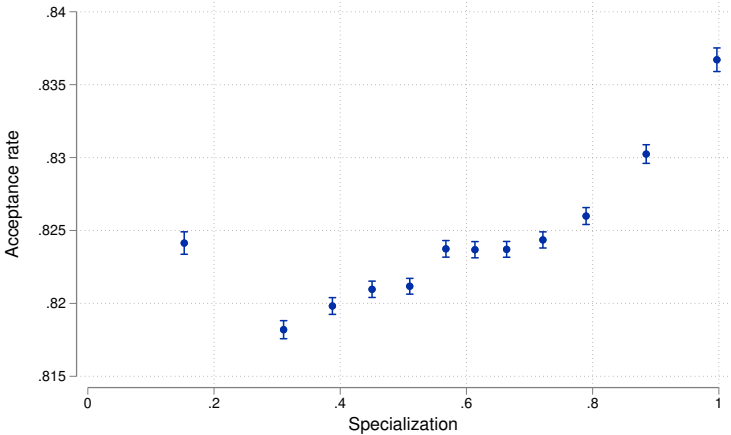
Fewer Discussions between Team Members



Controls: task distribution
Fixed effect: team age, team size, team, member

Fact 4 Higher Code Acceptance Rates

Task-specific knowledge



Controls: task distribution, team age
Fixed effect: team size, #submissions, team, worker

▸ Regression

Summary so far

- Negative correlation between specialization and output quality and quantity
- Specialized teams take longer to solve users' problems
- Specialized teams have higher code acceptance rate

Summary so far

- Negative correlation between specialization and output quality and quantity
- Specialized teams take longer to solve users' problems
- Specialized teams have higher code acceptance rate
- But task allocation is endogenous
- Ideal experiment:
  - Randomly change a team's task allocation to increase or decrease specialization
  - Impact on team productivity

Automatic Task Assignment

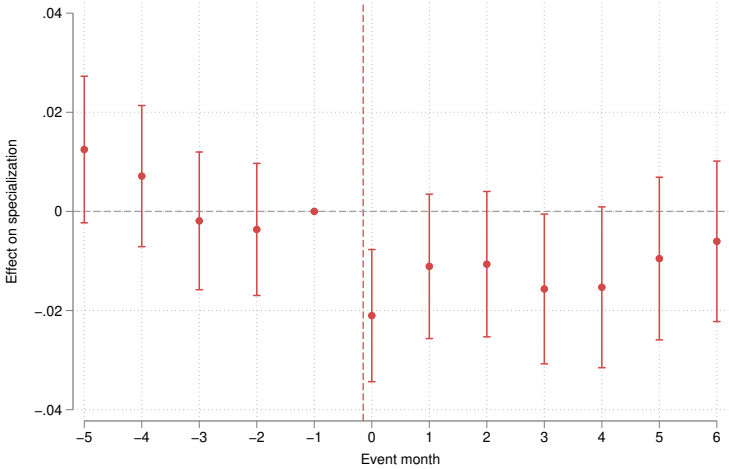
Research Design

- Automatic task assignment **decreases specialization**
  - **Evenly distributes** some **tasks** among team members
  - Teams enable it via configuration files

Research Design

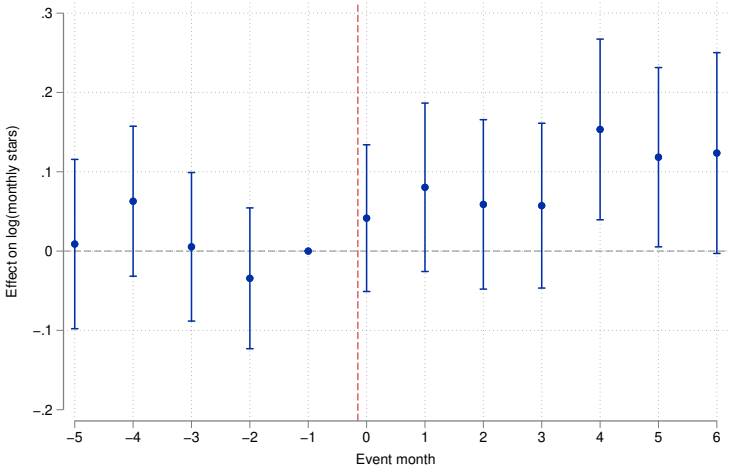
- Automatic task assignment **decreases specialization**
  - **Evenly distributes** some **tasks** among team members
  - Teams enable it via configuration files
- **Concern**: Adoption is not exogenous
  - **Solution**: Create a control group for teams that adopted
    1. 1:1 match on team size, task types, and activities in $t-1$ to $t-5$
    2. Use matched groups to construct treatment and control groups (98.2% matching rate)
  - **Validate** with empirical test of parallel trends assumption and compare outcomes using **diff-in-diff**

Specialization Decreased by 1.7%



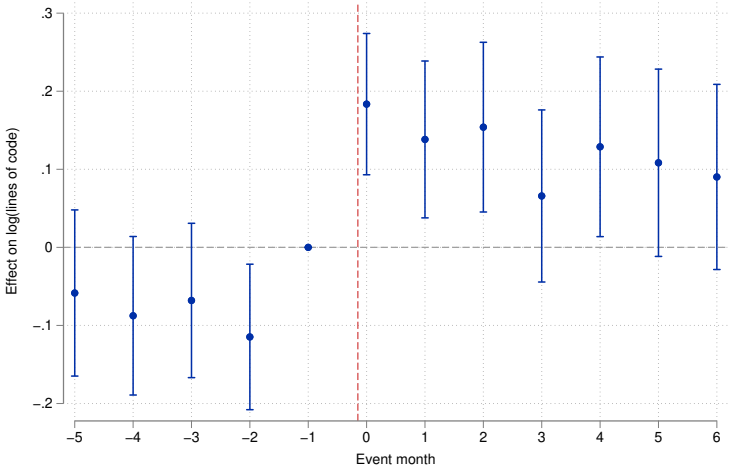
Implied constant treatment effect: -0.017*** (se=0.004)

Output Quality (Star) Increased after 3 months



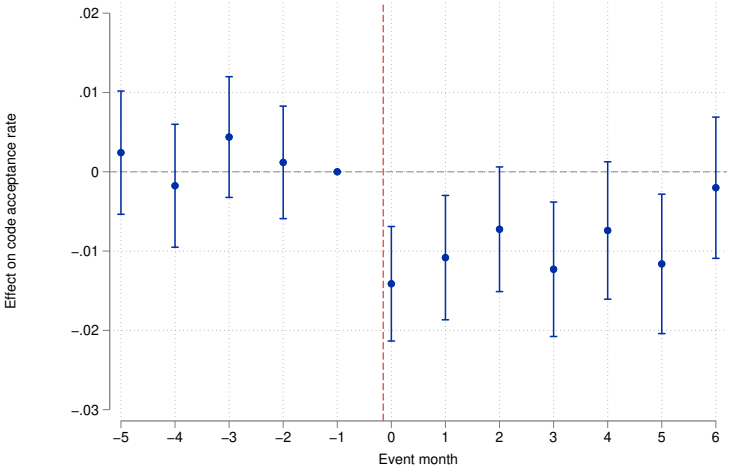
Implied constant treatment effect: 0.021 (se=0.022)

Output Quantity (Code) Increased by 17.7 %



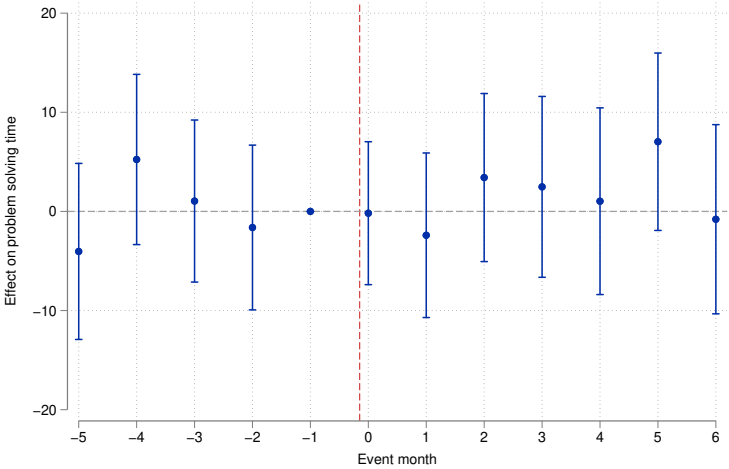
Implied constant treatment effect: 0.177*** (se=0.027)

Code Acceptance Rate Decreased by 2.4%



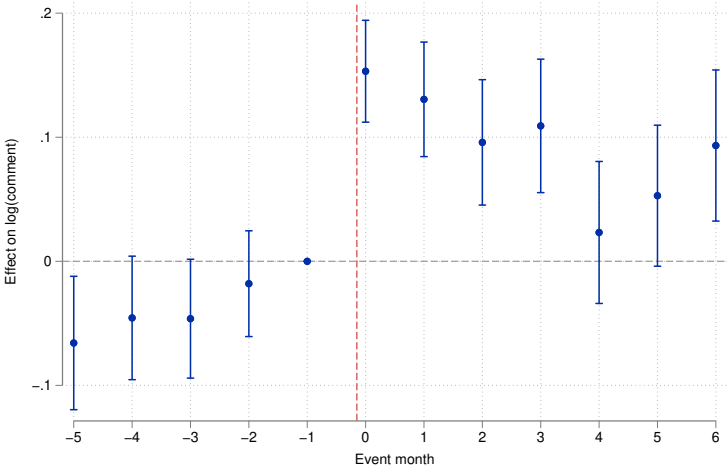
Implied constant treatment effect: -0.01*** (se=0.0.02)

No Effect for Problem Solving Speed



Implied constant treatment effect: 0.384 (se=1.970)

Potential Mechanism: Discussion Increased by 14.2%



Implied constant treatment effect after : 0.142*** (se=0.0.017)

Conclusion

- Measure task allocation specialization and team productivity
- Specialization is **negatively** associated with output **quality** and **quantity**
- Automatic assignment ↓ specialization, ↑ productivity
- Potential Mechanism: **loss of cross-task knowledge**
  - Reduced specialization increases team discussions
  - **Future**: text analysis to capture team communication patterns and knowledge spillover
- Specialization restricts knowledge exchange for innovation

Thank you

jinci.liu@iies.su.se

Outcome: Code acceptance rate

	(1)	(2)	(3)	(4)
SPE	-0.010***	0.020***	0.015***	0.015***
	(0.002)	(0.001)	(0.001)	(0.001)
Task type share control	Y	Y	Y	Y
Team age fixed effect	Y	Y	Y	Y
Team size fixed effect	Y	Y	Y	Y
Firm fixed effect	Y	Y		
Project type fixed effect		Y		
# code submission		Y	Y	Y
Team fixed effect			Y	Y
Member fixed effect				Y
R-squared	0.217	0.214	0.417	0.442
Dependent mean	0.824	0.825	0.824	0.824
Observations	3,213,677	2,982,846	3,213,677	3,125,570

▸ Graph version

Note. Weighted by 1/team size

Outcome: Problem Solving Speed

	(1)	(2)	(3)	(4)
SPE	17.655***	11.247***	4.411**	4.155**
	(1.496)	(1.540)	(1.385)	(1.332)
Task type share control	Y	Y	Y	Y
Team age fixed effect	Y	Y	Y	Y
Team size fixed effect	Y	Y	Y	Y
Firm fixed effect	Y	Y		
Project type fixed effect		Y		
# question fixed effect		Y	Y	Y
Team fixed effect			Y	Y
Member fixed effect				Y
R-squared	0.166	0.166	0.355	0.398
Dependent Mean	51.182	51.372	51.182	51.579
Observations	2,364,271	2,267,892	2,364,271	2,294,737

▸ Graph version

Note. Weighted by 1/team size

Outcome:Log (Lines of Code)

	(1)	(2)	(3)	(4)
SPE	-1.224***	-1.260***	-0.954***	-0.934***
	(0.021)	(0.022)	(0.015)	(0.015)
Dependent mean	8.124	8.128	8.124	8.118
R-squared	0.398	0.403	0.597	0.616
Observations	3,212,871	2,982,107	3,212,871	3,124,751
Task type share control	Y	Y	Y	Y
Team age fixed effect	Y	Y	Y	Y
Team size fixed effect	Y	Y	Y	Y
Firm fixed effect	Y	Y		
Project type fixed effect		Y		
Team fixed effect			Y	Y
Member fixed effect				Y

▸ Graph version

Note. Weighted by 1/team size

Team specialization index

- $\mathbb{K}$ the set of all tasks
- Each team m consists of $N_m \in \mathbb{N}$ members, handles a set of task $K_m \subseteq \mathbb{K}$
- Define A_m to be task allocation matrix of team m
  - $A_m(i,j)$ represents member i's labor input on task j
-

$$A_m = \begin{pmatrix} A_m(1,1) & A_m(1,2) & \cdots & A_m(1,|K_m|) \\ A_m(2,1) & A_m(2,2) & \cdots & A_m(2,|K_m|) \\ \vdots & \vdots & \ddots & \vdots \\ A_m(N_m,1) & A_m(N_m,2) & \cdots & A_m(N_m,|K_m|) \end{pmatrix}$$

- Member i's labor share: $l_i := \sum_j A_m(i,j)$
- Task share: $\alpha_j := \sum_i A_m(i,j)$

Back

Team specialization index

- Euclidean distance

$$d(A_m - G_m) = \sqrt{(A_m(i,j) - G_m(i,j))^2}$$

$d(S_m - G_m) =$

$$\sqrt{\sum_j \left(\underbrace{\lfloor \alpha_j \rfloor}_{\text{\# of 1s}} \cdot \left(1 - \frac{\alpha_j}{N_m}\right)^2 + \underbrace{\left(N_m - \lceil \alpha_j \rceil\right)}_{\text{\# of 0s}} \cdot \left(\frac{\alpha_j}{N_m}\right)^2 + \left(\lceil \alpha_j \rceil - \lfloor \alpha_j \rfloor\right) \cdot \left(\underbrace{\alpha_j - \lfloor \alpha_j \rfloor}_{\text{residual workload}} - \frac{\alpha_j}{N_m} \right)^2 \right)}$$

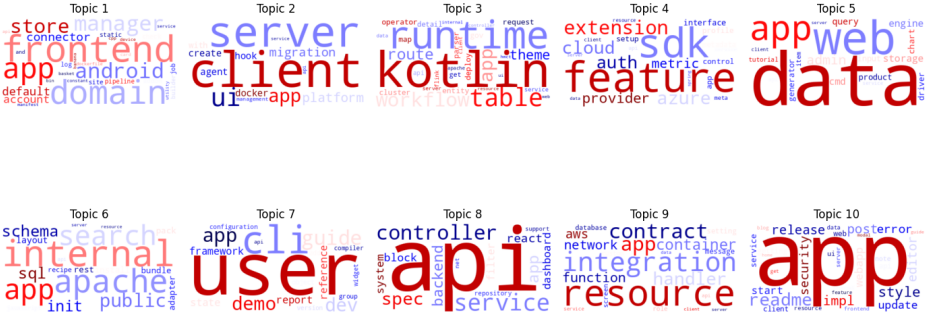
- Kullback-Leibler divergence

$$d(A_m - G_m) = \sum_i \sum_j A_m(i,j) \log\left(\frac{A_m(i,j)}{G_m(i,j)}\right)$$

$$d(S_m - G_m) = \sum_j \left(\lfloor \alpha_j \rfloor \cdot \log\left(\frac{1}{\frac{\alpha_j}{N_m}}\right) + \left(\lceil \alpha_j \rceil - \lfloor \alpha_j \rfloor\right) \cdot \left(\alpha_j - \lfloor \alpha_j \rfloor\right) \cdot \log\left(\frac{\alpha_j - \lfloor \alpha_j \rfloor}{\frac{\alpha_j}{N_m}}\right) \right)$$

Back

Task Type



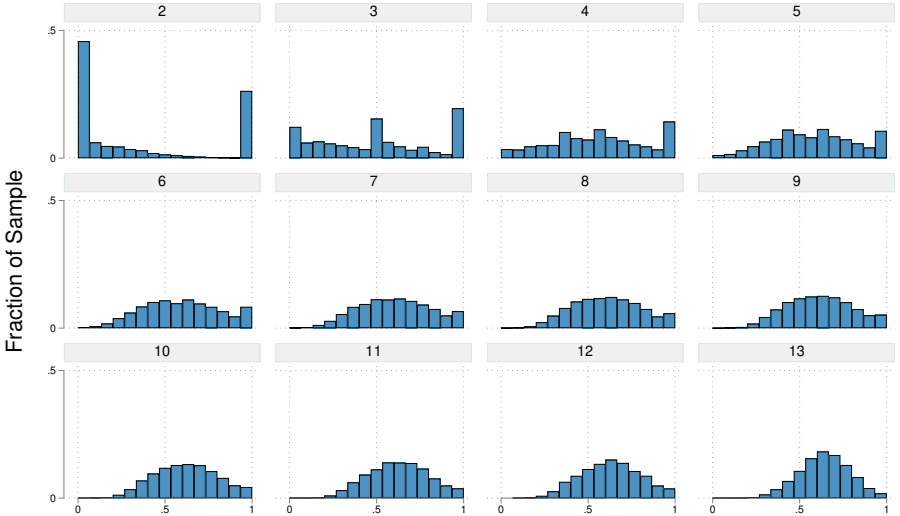
Back

Project Type

Back

Specialization by Team Size

Final sample only includes teams with size >3



Back

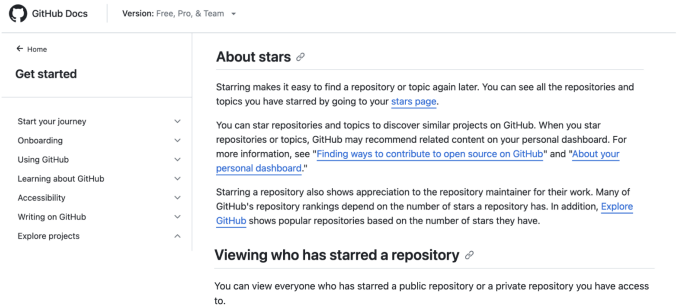
Summary Statistics

	N	Mean	St. Dev.	Min	Median	Pctl(75)	Pctl(95)
SPE	439079	0.59	0.24	0.00	0.59	0.75	1.00
Team size	439079	7.39	8.77	4	5	7.0	17
Task type	439079	4.69	2.24	2	4	6	9
Lines of code	439079	43558.76	372784.61	0.00	3325	12926	119168
Activities	439079	750.20	2322.29	0.00	381	784	2379
Monthly Stars	439079	44.55	254.21	0.00	3	18	197
Comments	439079	144.98	362.40	0.00	54	145	533
Solving time	300034	51.18	122.35	0.00	13.86	41.36	224.82
Edited files	439079	570.47	2985.87	0.00	123	363	1917
Code acceptance rate	433833	0.82	0.17	0.00	0.86	0.94	1.00
Create year	439079	2018.14	2.82	2011	2018	2020	2022

Notes: This table provides the summary statistics for the main variables of interest at the team-month level. Data is from 2017-01 to 2023-12.

Back

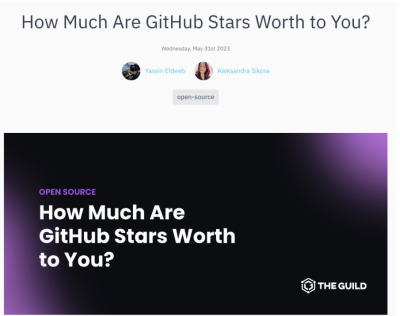
What is star

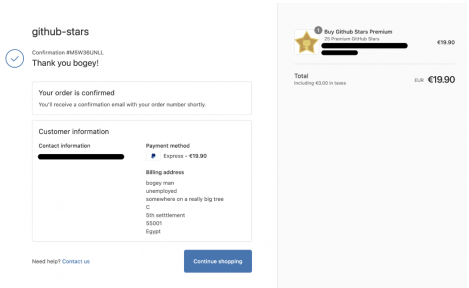


- **Show appreciation**
- **GitHub will recommend related content on your dashboard**
- **Many Github repo rankings depend on stars**
- **Not anonymized**

Back

Buying GitHub Stars





19.9 Euros for 25 stars ~ $0.88/star

And after a month, they are all gone. GitHub detected and banned them.

Back

75% developers check stars metric before using it

What's in a GitHub Star? Understanding Repository Starring Practices in a Social Coding Platform

Hudson Borges, Marco Tulio Valente

Show more

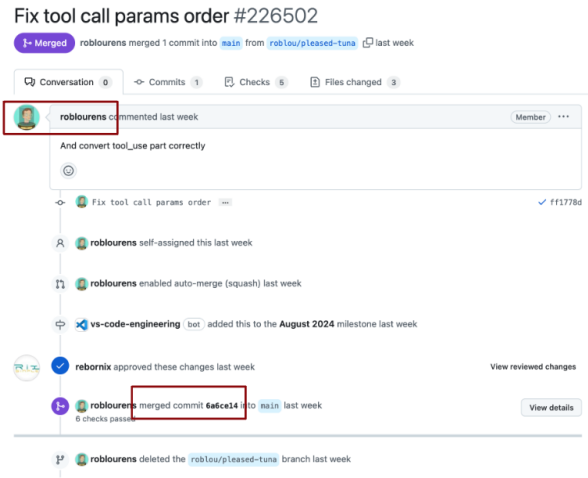
Add to Mendeley Share Cite

https://doi.org/10.1016/j.jss.2018.09.016 Get rights and content

Highlights

- Developers star repositories mainly to show appreciation or to bookmark projects
- 3 out of 4 developers check the stars metric before using or contributing to projects
- But developers also evaluate other factors, such as code quality and documentation
- Fast growth in the number of stars is often a result of promotion in social sites
- When ranking projects, we should check whether stars are result of active promotion

Manage to merge code

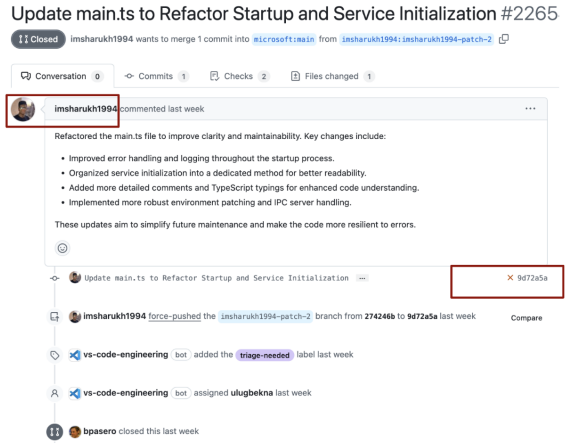


Submit code change

Manage to merge

Back

Fail to merge code

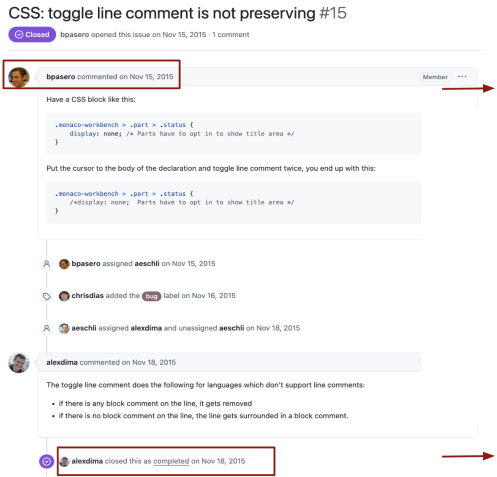


Submit code change

Fail to merge

Back

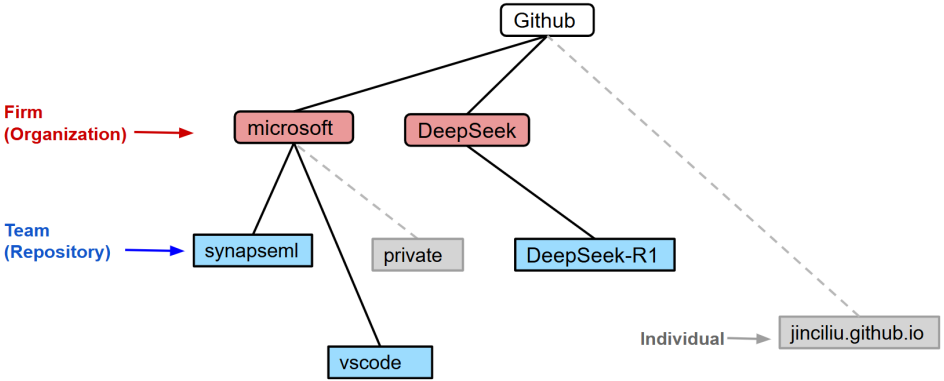
Time to solve problem



Back

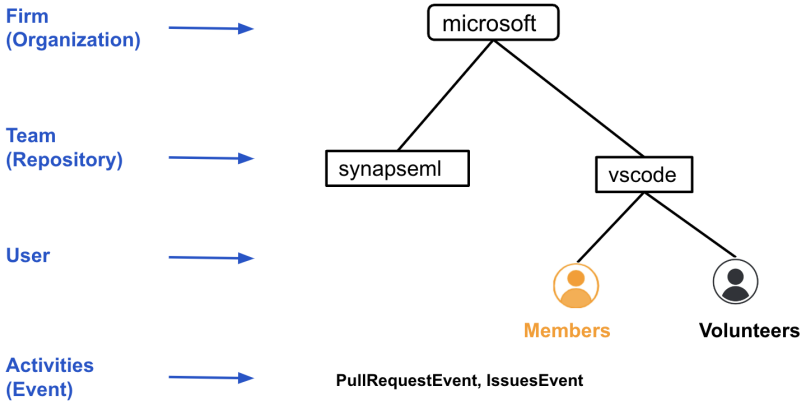
What is GitHub?

- World's largest open source platform for software development



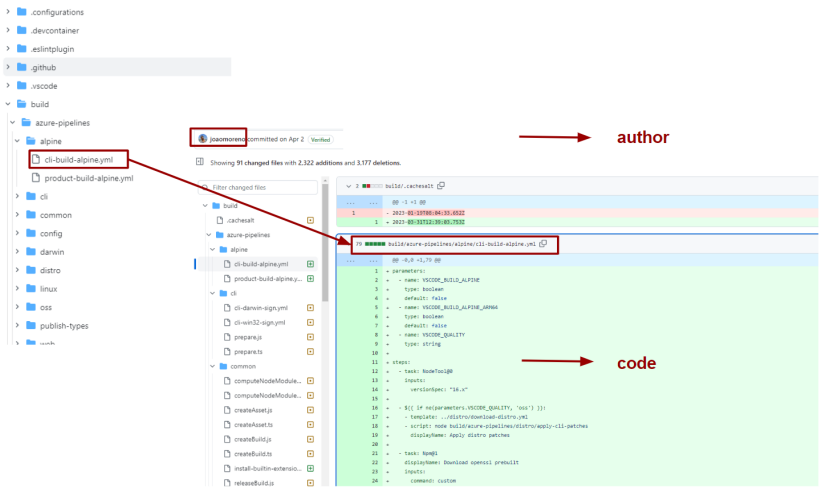
- Focus on public teams under firms Back

Team Member



- Task allocation within team members Back

Code File


.configurations
.devcontainer
.eslintplugin
.github
.vscode
build
azure-pipelines
alpine
cli-build-alpine.yml
product-build-alpine.yml
cli
common
config
darwin
distro
linux
oss
publish-types
joaomoreno committed on Apr 2
Verified
Showing 91 changed files with 2,322 additions and 3,177 deletions.
Filter changed files
build
.cachesalt
azure-pipelines
alpine
cli-build-alpine.yml
product-build-alpine.y...
cli
cli-darwin-sign.yml
cli-win32-sign.yml
prepare.js
prepare.ts
common
computeNodeModule...
computeNodeModule...
createAsset.js
createAsset.ts
createBuild.js
createBuild.ts
install-builtin-extensio...
releaseBuild.js
build/.cachesalt
@@ -1 +1 @@
- 2023-03-19T08:04:33.652Z
+ 2023-03-31T12:39:05.753Z
build/azure-pipelines/alpine/cli-build-alpine.yml
@@ -0,0 +1,79 @@
+ parameters:
+ - name: VSCODE_BUILD_ALPINE
+ type: boolean
+ default: false
+ - name: VSCODE_BUILD_ALPINE_ARM64
+ type: boolean
+ default: false
+ - name: VSCODE_QUALITY
+ type: string
+
+ steps:
+ - task: NodeTool@0
+ inputs:
+ versionSpec: "16.x"
+
+ - ${{ if ne(parameters.VSCODE_QUALITY, 'oss') }}:
+ - template: ../distro/download-distro.yml
+ - script: node build/azure-pipelines/distro/apply-cli-patches
+ displayName: Apply distro patches
+
+ - task: Npm@1
+ displayName: Download openssl prebuilt
+ inputs:
+ command: custom
author
code
Back

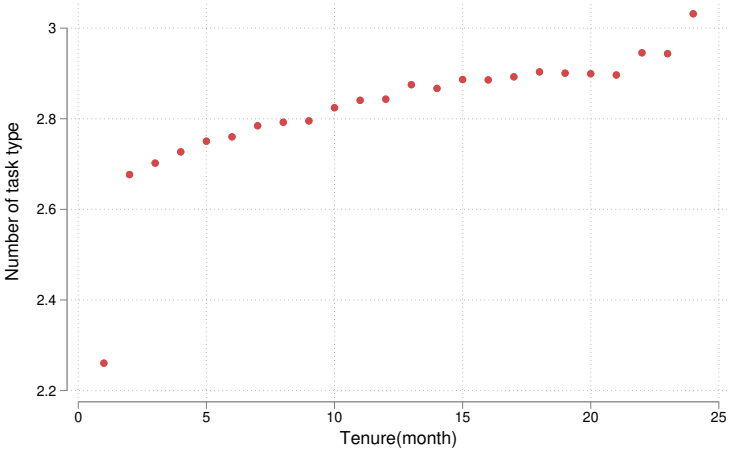
Task Type

Use Latent Dirichlet allocation (LDA) to classify code files into 10 task types

	Task type	Key words (LDA result)	Occupational role
1	Frontend development	Frontend, UI	Front-end engineer
2	Server management, Platform migration	Client, server	DevOps engineer
3	Android mobile development	Kotlin, runtime	Mobile engineer
4	Cloud feature implementation	Feature, sdk	Cloud engineer
5	Data management	Data, web	Data engineer
6	Internal system management	Internal, apache	System Administrator
7	CLI Development and Framework	User, cli	Technical Writer
8	API and Backend Services	API, controller	Back-end engineer
9	Integration system	Integration, function	System Integrator
10	App Development and UI Design	App, style	App Developer

▸ Word Cloud10 ▸ Data Summary

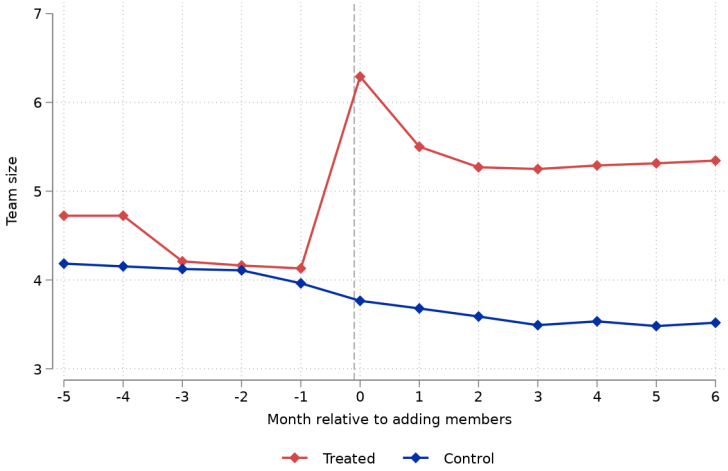
New members start with fewer tasks



Fixed effect: team, team size

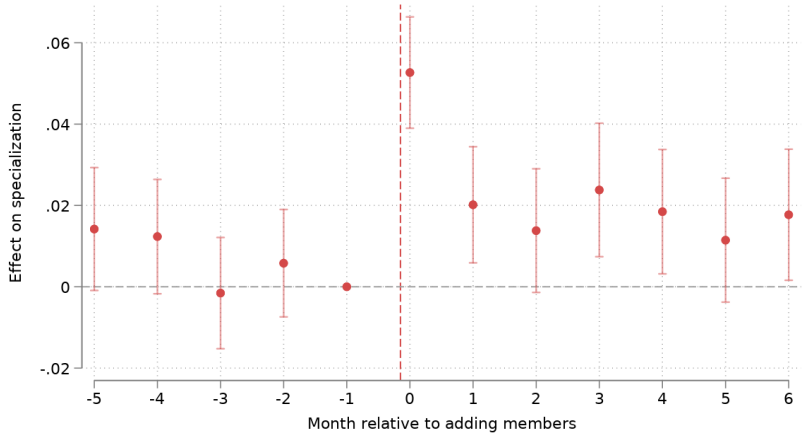
Back

Adding New Members



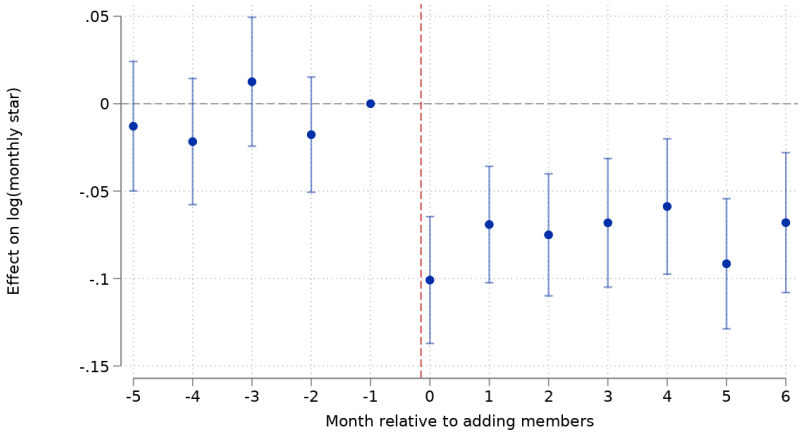
Back

Specialization Increased by 5% in the first month



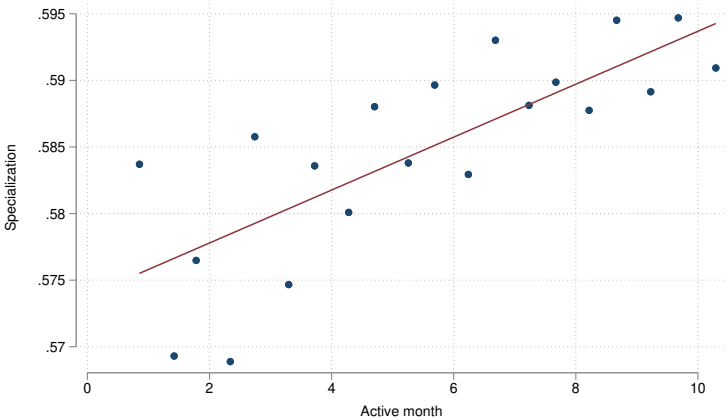
Back

Output Quality Decreased by 10%



Back

Teams are slightly more specialized over time



Teams active at least 10 months
Controls: task distribution
Fixed effect: Team size

▸ Distribution

References I

Autor, D. H., Levy, F., and Murnane, R. J. (2003). The skill content of recent technological change: An empirical exploration. *The Quarterly journal of economics*, 118(4):1279–1333.

Becker, G. S. and Murphy, K. M. (1992). The division of labor, coordination costs, and knowledge. *The Quarterly journal of economics*, 107(4):1137–1160.

Bloom, N., Eifert, B., Mahajan, A., McKenzie, D., and Roberts, J. (2013). Does management matter? evidence from india. *The Quarterly journal of economics*, 128(1):1–51.

Bloom, N., Garicano, L., Sadun, R., and Van Reenen, J. (2014). The distinct effects of information technology and communication technology on firm organization. *Management Science*, 60(12):2859–2885.

Borges, H. and Valente, M. T. (2018). What's in a github star? understanding repository starring practices in a social coding platform. *Journal of Systems and Software*, 146:112–129.

Casalnuovo, C., Vasilescu, B., Devanbu, P., and Filkov, V. (2015). Developer onboarding in github: the role of prior social links and language experience. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering*, pages 817–828.

Giroud, X., Matvos, G., Seru, A., and Silva, R. C. (2022). Managing resource (mis) allocation.

References II

Gong, J. and Png, I. P. (2024). Automation enables specialization: Field evidence. *Management Science*, 70(3):1580–1595.

Kohlhepp, J. (2024). The inner beauty of firms.

Levinthal, D. A. and March, J. G. (1993). The myopia of learning. *Strategic management journal*, 14(S2):95–112.

Rosen, S. (1983). Specialization and human capital. *Journal of Labor Economics*, 1(1):43–49.

Smith, A. (1776). *The wealth of nations [1776]*, volume 11937. na.

Vasilescu, B., Yu, Y., Wang, H., Devanbu, P., and Filkov, V. (2015). Quality and productivity outcomes relating to continuous integration in github. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering*, pages 805–816.

Wagner, S. and Ruhe, M. (2018). A systematic review of productivity factors in software development. *arXiv preprint arXiv:1801.06475*.

Young, A. (1928). Increasing returns and economic progress. *The economic journal*, 38(152):527–542.