World of Code: Enabling a Research Workflow for Mining and Analyzing the Universe of Open Source VCS Data

Audris Mockus
University of Tennessee
audris@utk.edu

DAMSS'21 *[2021-12-03 Fri]*

Outline

About the speaker

Where/Who/What

- Moscow Physical Technical Institute, Carnegie Mellon University
- 23 years at Bell Labs/Avaya Labs
- Since Fall'14 at the University of Tennessee
  - Professor of Digital Archeology and Evidence Engineering
  - or Data Science/Big data/Software Engineering
- Looking for interested students, postdocs, visitors

Open Source Software

- 1980-90s: a mere curiosity
  - Free Software GNU (35 years);
  - Linux (28 years);
  - Slackware (26 years);
  - Apache (24 years);
  - Mozilla (25 years)
- Attacked by critics then as
  - Expensive to maintain
  - Unsafe
  - Unsustainable

Open Source Software Miracle

- 2021:
  - Over 170M projects
  - 60M contributors
  - RedHat 3.4B revenue / 15 percent annual growth
- Why:
  - Reuse that works
  - Distributed decision making
  - Code can be trusted to be there
  - Sophisticated collaboration tools

Future is clouded

- What worked over past 20+ years may break
  - Supply chain attacks
    - Easy to exploit OSS-wide dependencies
  - Maintainer exhaustion
    - Aging out in critical projects
    - Increasing complexity
    - Increasing user base
    - Commercial participants not contributing back
  - Move to the cloud
    - No need for software when using a service
    - Amazon appropriating open source to provide competing services

Two outcomes

- To believers:
  - I am scared, tell me how my research can save OSS (and the world)?
- To skeptics:
  - I hear emotional BS, are these problems real?

To skeptics

- E.g., NPM has over 1.7M packages
  - Median number of direct dependents is 2, recursive 200 (emberjs)
  - supply chain (introducing malicious dependencies) attacks are common
- Skeptic: well...
  - NPM is a house of cards built on sandcastles
  - gone with a gust of wind
  - and good riddance...
- Response:
  - Much of web infrastructure is built on NPM
  - Many developers rely/work on it
  - Would be very difficult to replace

Press Releases: arstechnica.com

A rash of supply chain attacks hitting open source software over the past year shows few signs of abating, following the discovery this week of two separate backdoors slipped into a dozen libraries downloaded by hundreds of thousands of server administrators.

. . . someone compromised the server used to develop new versions of the program. The attacker then used the access to distribute a backdoor that was downloaded more than 900,000 times . . .

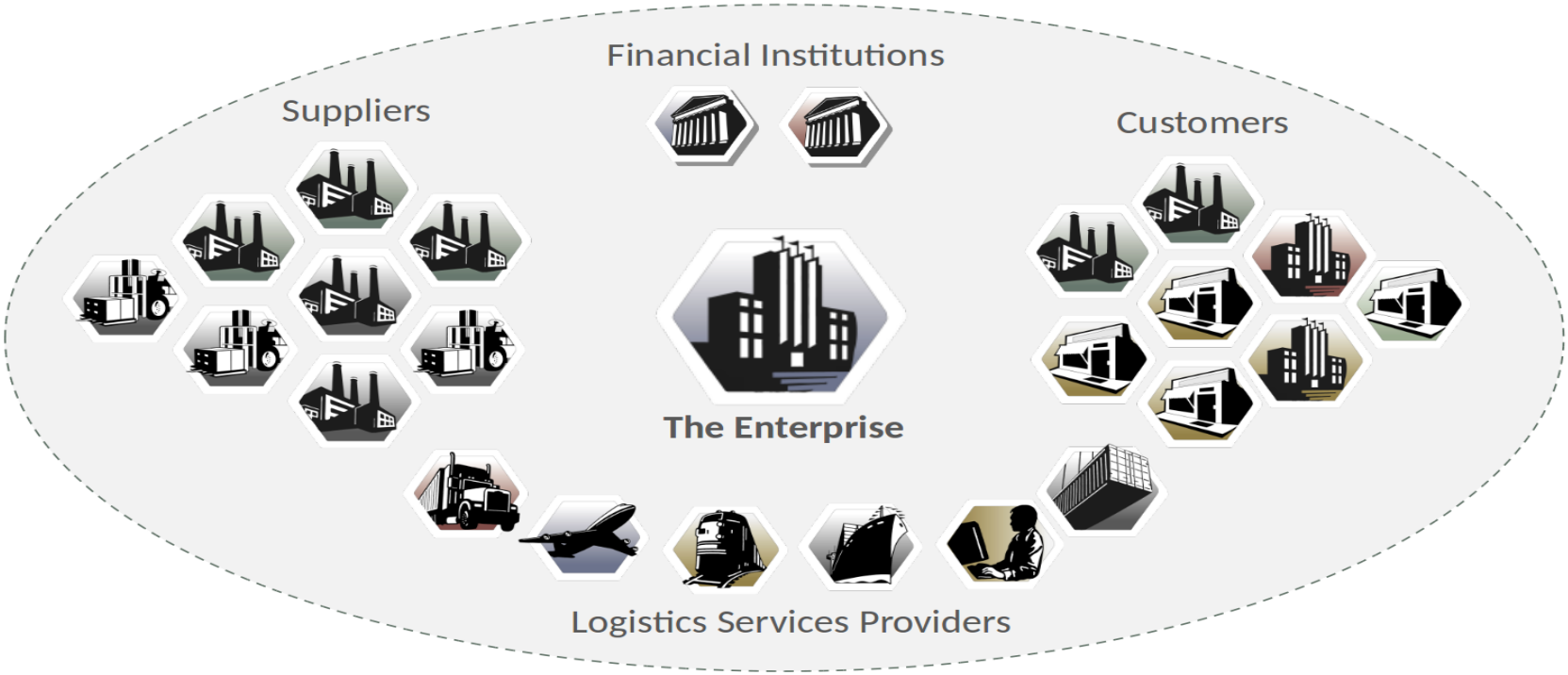
The problem may be real, but what to do?

- What is Open Source Software Supply Chain?
- What are its Risks and How to Manage Them?
- How to do Research on it?
  - World of Code or WoC
    - Current, Complete, Curated, Cross-referenced Collection of Open Source Version Control Data (CCCCCosvCd)
    - "Research Ready" for Supply Chain Research

Definition (What is Supply Chain)

is a set of three or more companies directly linked by one or more of the upstream or downstream flows of products, services, finance, and information from a source to a customer
Key features:

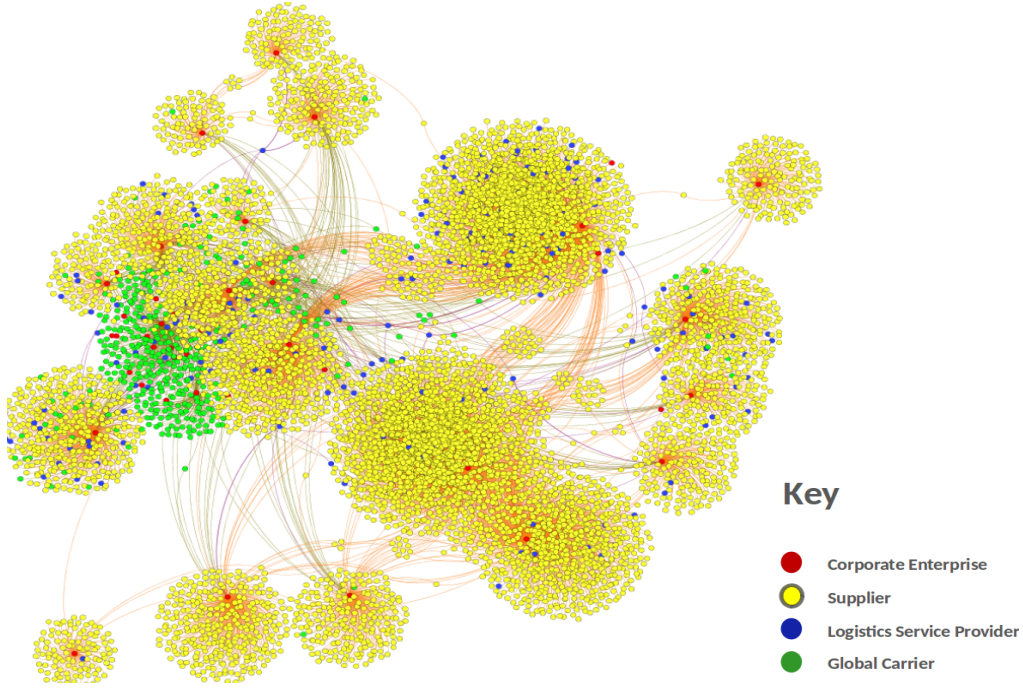
- Complex interdependencies
- Distributed decisions



Definition (What is Supply Chain)

is a set of three or more companies directly linked by one or more of the upstream or downstream flows of products, services, finance, and information from a source to a customer
Key features:

- Complex interdependencies
- Distributed decisions



Definition (Elements of Software Supply Chains (SSCs))

- Developers and groups ("companies")
- Relationships among software projects or packages
- Changes to the source code (e.g., to files, modules, frameworks, or entire distributions)

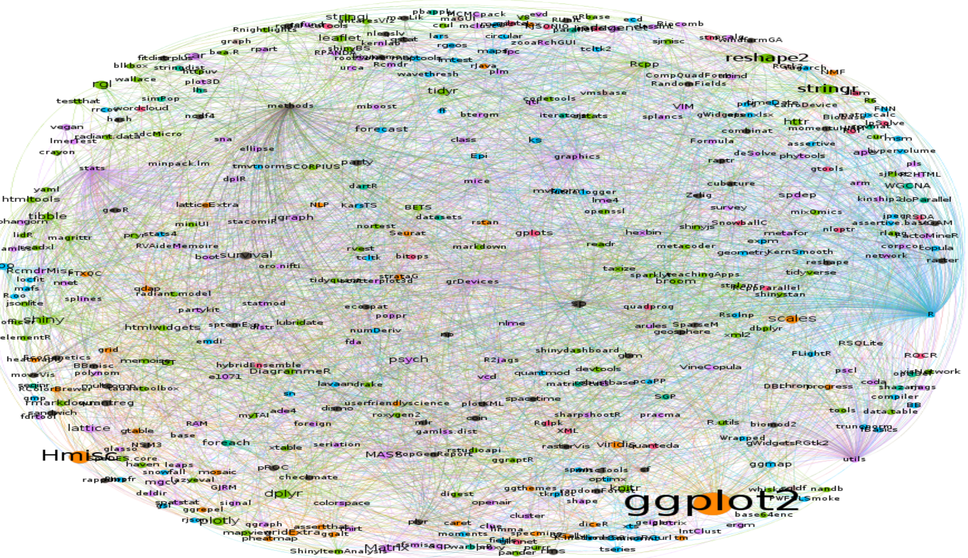
SSC of the 1st kind

- Technical dependencies among projects with change effort as product flow
- Primary risks: unknown vulnerabilities, breaking changes, lack of maintenance, lack of popularity

Examples of SSC of the first kind

- Python: import re
- Java: import java.util.Collection;
- JavaScript: package.json

e.g., CRAN


ggplot2
reshape2
stringr
stringi
leaflet
tidyr
Rcpp
Hmisc
scales
shiny
tibble
plotly
dplyr
lattice
Matrix
psych
MASS
survival
graphics
methods
stats
utils
R
forecast
viridis
httr
spdep
igraph
gplots
htmlwidgets
foreach
rgl
ggmap
broom
DiagrammeR

SSC of the 2nd kind

- Copying of the source code from project to project as product flow
- Primary risks: license compliance, unfixed vulnerabilities/bugs, missing updated functionality

Examples of SSC of the second kind

- Implementation of a complex algorithm
- Useful template
- Build configuration

SSC of the 3rd kind

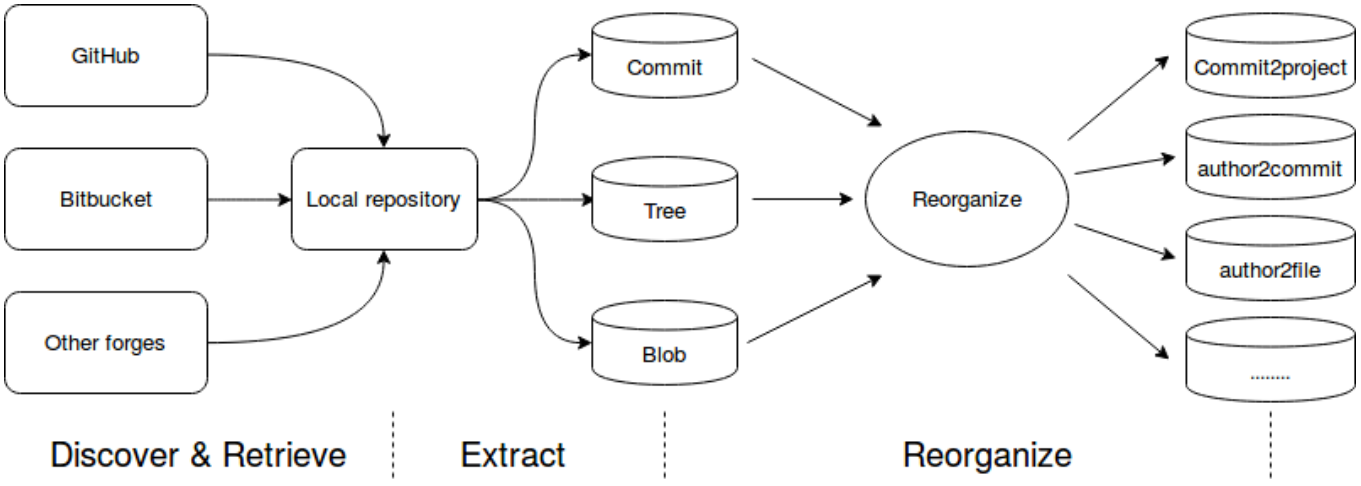
- Knowledge (product) flow through code changes as developers learn from and impart their knowledge to the source code
- Primary risks: developers may leave, companies may discontinue support

Examples of SSC of the third kind

- Developers gaining skills with tools/packages/practices
- Developers spreading practices, e.g., testing frameworks

Skeptic: OK, but how to measure?

- WoC: Discover, Retrieve, Store, Analyze, Update [10, 8]



- 170+M projects; 60+M Authors; 3.1B Commits; 12B Blobs and Trees
- Simply cloning: $\geq$\$3PB
- Specialized database without redundancies: 200TB

WoC: What is it?

- Complete: capture data from all public git repos (approx 50 forges)
- Current: quarterly releases
- Curated/Research Ready: e.g., author aliasing, deforking, bot identification, ...
- Cross-referenced: First-class entities mapped to other first-class entities
- WoC version U numbers are at bitbucket.org/swsc/overview/
- How to use: github.com/woc-hack/tutorial
- Web interface: worldofcode.org

Skeptic: but OSS data are rather bad

"productive" authors

Number of commits	Author
10960000	one-million-repo < *mikigal.acc@gmail.com* >
4400778	datakit < *datakit@docker.com* >
2463758	greenkeeper[bot] < *greenkeeper[bot]@users.noreply.github.com* >
2063212	Auto Pilot < *noreply@localhost* >
1864730	Your Name < *you@example.com* >

Silly competitions

- Fake commits with everyone's name to have most contributors

Authors	Repo
389993	cirosantilli/imagine-all-the-people

- The longest chain of commits

Length of commit chain	Repo
9959999	github.com/one-million-repo/biggest-repo-ever

More examples at bitbucket.org/swsc/overview/fun/

Data That Needs to be Curated

- Author IDs
- Repository forks
- Code dependencies (dozens of languages)
- Project types
- Link to external data sources
- Many other challenges

Example: Identifying Authors

- We have 34M strings
- Are these two the same person?
  - Aaron Lee < aaron.lee@rackspace.com>
  - Aaron Lee < wwkeyboard@gmail.com>
- Text similarity (adjusted for common names)
- Behavioral fingerprints:
  - Similarity of written text (Doc2Vec embedding)
  - Change the same source code files
  - Work in the similar time-zones
- Trained machine learning: 99.99 accuracy

Skeptic: but why would I use it?

- To improve research quality
  - Most software development is not for isolated projects
  - But current research practices ignore SSC relationships and lead to inadequate models/tools/practices
- To do entirely new types of research
  - What made OSS so successful so far?
  - Create fundamental theories in software engineering enabled by WoC and similar observatories

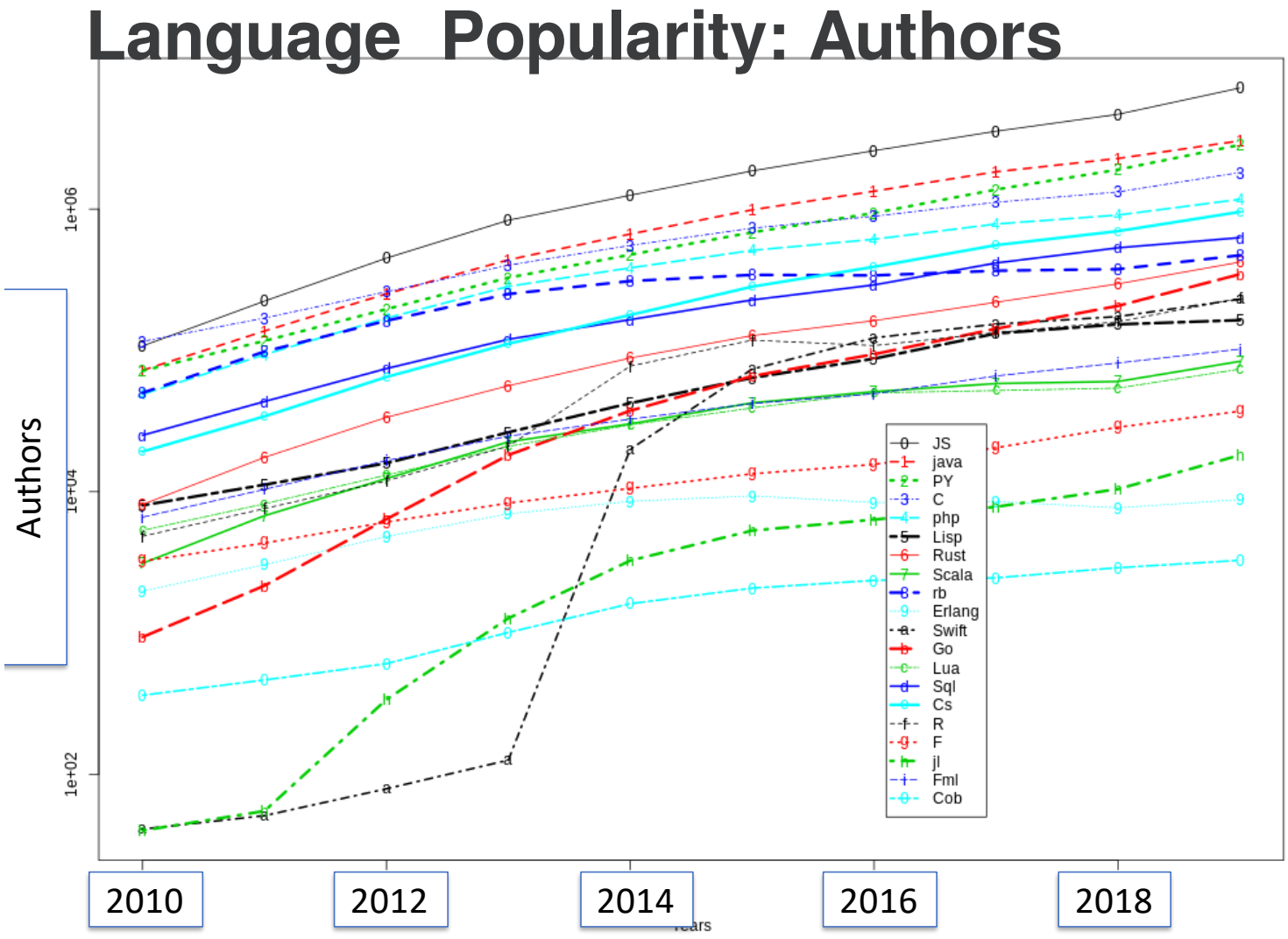
Skeptic: what exactly is presently inadequate?

- Key advantages WoC-like infrastructure
- Completeness: proper instead of convenience sampling
- Cross-referencing:
  - first time ever measure/study Type II and III SSCs and downstream of Type I SSCs
- Curation: don't need to spend a lifetime cleaning data Not accounting for activities downstream

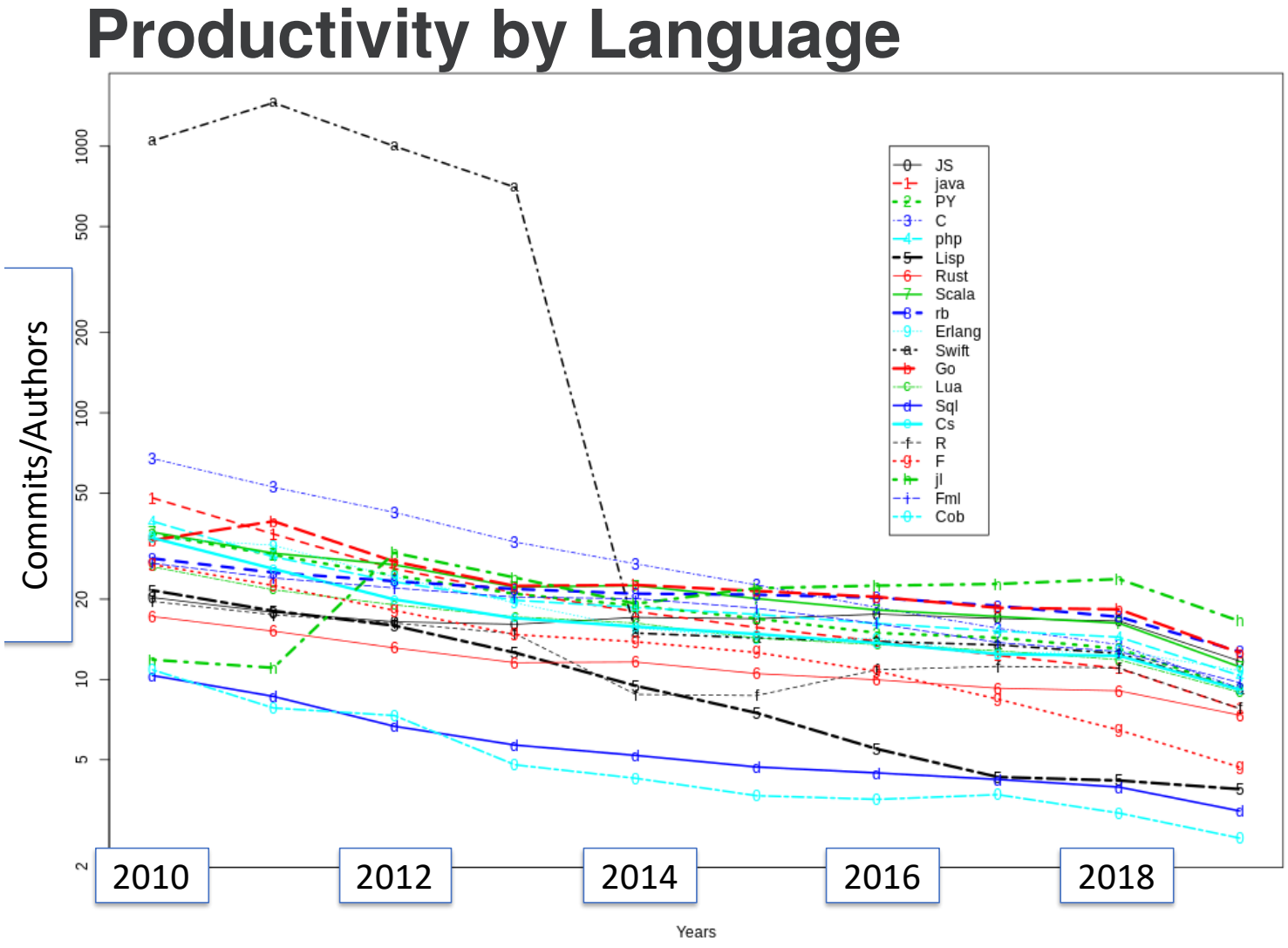
Skeptic: any track record?

- Contextualize/Correct/Impute [11]
  - Author aliasing/Bot detection: via behavioral fingerprinting [1, 5, 6]
  - De-forking/de-cloning via shared commits [12]
- Type I: dependencies
  - Models of spread [9]
  - Models of popularity [4]
  - Patterns of effort contribution [3]
- Type II: copying
  - Orphan vulnerabilities [13]
- Type III: knowledge
  - Knowledge at loss [14]
  - Developer impact [7]
  - Skill spaces [2]
  - Eight implemented use cases
    - Relationships: code flow, technical and tool dependencies, knowledge flow [3, 4, 13]

What Languages are Popular?



How Difficult Each Language is?



Can we predict which technology will prosper/fail?

- Two data-science technologies: abstraction of data.frame in R
  - tidy vs data.table
- Sample: all R-language files in public repositories
  - 1.5M files, 5M versions
- When was the first time tidy or data.table included?
  - 17,536 projects use data.table
  - 7,032 projects use tidy

Results (Choice and Decision Maker)

- Choice
  - Exposure: Recent and Cumulative deployments, Mentions on StackOverflow
  - Qualities: Activity, developers,responsiveness, open issues
- Decision maker
  - Activity, developers
  - Performance needs
  - Social network
  - Technical network

Note: Red- negative, Green- positive

Skeptic (turned into believer): ok ok, how do I sign up for this?

- bitbucket.org/swsc/overview:
  - related publications / data / analysis
- Hackathon Nov 1-3
  - woc-hack.slack.com : sign up at `http://bit.ly/WoC-Hack`
  - github.com/woc-hack/tutorial
  - Signup form `http://bit.ly/WoC-Signup`

Takeaways

- OSS: why it is worthy of study
- SSC: why relevant for present software, types and risks
- OSS Observatory (WoC): help speed discovery in this novel area
- You can benefit too!

References

Sadika Amreen, Yuxia Zang, Chris Bogart, Russell Zaretzki, and Audris Mockus.

Alfaa: Active learning fingerprint based anti-aliasing for correcting developer identity errors in version control systems.
International Journal of Empirical Software Engineering, 2019.

Tapajit Dey, Andrey Karnauch, and Audris Mockus.

Representation of developer expertise in opensource software.
In *ICSE 2021*. ACM Press, May 2021.

Tapajit Dey, Yuxing Ma, and Audris Mockus.

Patterns of effort contribution and demand and user classification based on participation patterns in npm ecosystem.
In *Proceedings of the 15th International Conference on Predictive Models and Data Analytics in Software Engineering*. ACM, 2019.

Tapajit Dey and Audris Mockus.

Are software dependency supply chain metrics useful in predicting change of popularity of npm packages?
In *Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering*, pages 66–69. ACM, 2018.

Tapajit Dey, Sara Mousavi, Eduardo Ponce, Tanner Fry, Bogdan Vasilescu, Anna Filippova, and Audris Mockus.

Detecting and characterizing bots that commit code.
In *IEEE Working Conference on Mining Software Repositories*, May 2020.

Tanner Fry, Tapajit Dey, Andrey Karnauch, and Audris Mockus.

A dataset and an approach for identity resolution of 38 million author ids extracted from 2b git commits.
In *IEEE Working Conference on Mining Software Repositories: Data Showcase*, May 2020.

Andrey Karnauch, Sadika Amreen, and Audris Mockus.

Developer reputation estimator (dre).
In *ASE'19*, 2019.

Yuxing Ma, Chris Bogart, Sadika Amreen, Russell Zaretzki, and Audris Mockus.