

The Standard of Rigor for MSR Research

A 10-Year Evolution

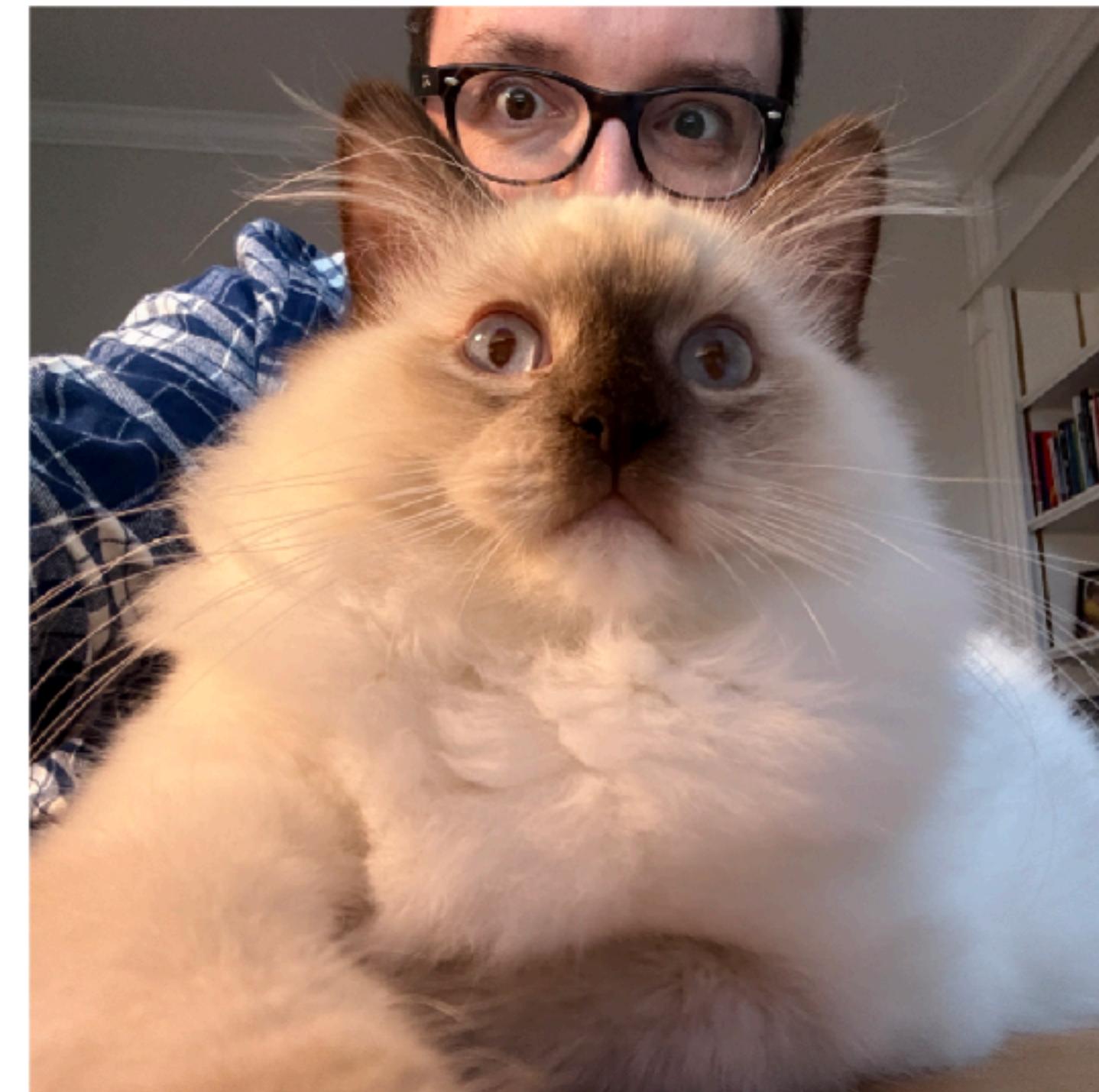
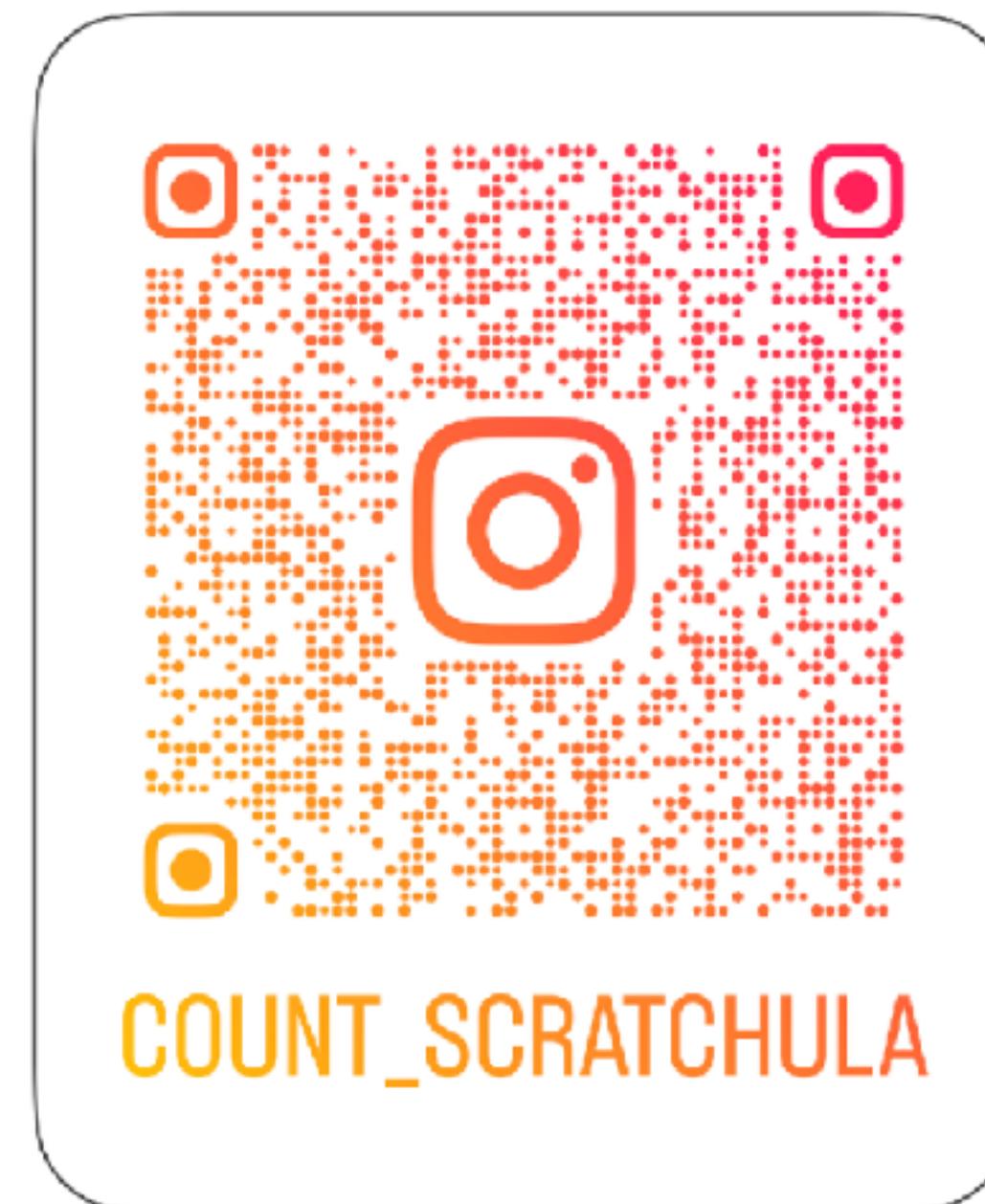
Bogdan Vasilescu
@b_vasilescu

About me

- Raj Reddy Associate Professor of Software and Societal Systems
- Societal Computing PhD program director
- Socio-Technical Research Using Data Excavation Lab

About me

- Raj Reddy Associate Professor of Software and Societal Systems
- Societal Computing PhD program director
- Socio-Technical Research Using Data Excavation Lab



About me

- Raj Reddy Associate Professor of Software and Societal Systems
- Societal Computing PhD program director
- Socio-Technical Research Using Data Excavation Lab
- Reading since MSR 2005, attending since MSR Summer School 2012



 Check for updates

When Do Changes Induce Fixes?

(On Fridays.)

Jacek Śliwerski
International Max Planck Research School
Max Planck Institute for Computer Science
Saarbrücken, Germany
sliwers@mpi-sb.mpg.de

Thomas Zimmermann Andreas Zeller
Department of Computer Science
Saarland University
Saarbrücken, Germany
{tz, zeller}@acm.org

ABSTRACT
As a software system evolves, programmers make changes that sometimes cause problems. We analyze CVS archives for *fix-inducing changes*—changes that lead to problems, indicated by fixes. We show how to automatically locate fix-inducing changes by linking a version archive (such as CVS) to a bug database (such as BUGZILLA). In a first investigation of the MOZILLA and ECLIPSE history, it turns out that fix-inducing changes show distinct patterns with respect to their size and the day of week they were applied.

Which change properties may lead to problems? We can investigate which properties of a change correlate with inducing fixes, for instance, changes made on a specific day or by a specific group of developers.

How error-prone is my product? We can assign a *metric* to the product—on average, how likely is it that a change induces a later fix?

How can I filter out problematic changes? When extracting the

 Check for updates

Mining Email Social Networks*

Christian Bird, Alex Gourley,
Prem Devanbu, Michael Gertz
Dept. of Computer Science, Kemper Hall,
University of California, Davis,
Davis, California Republic.
cabird,devanbu@ucdavis.edu

Anand Swaminathan
Graduate School of Management,
University of California, Davis,
Davis, California Republic.
aswaminathan@ucdavis.edu

ABSTRACT
Communication & Co-ordination activities are central to large software projects, but are difficult to observe and study in traditional (closed-source, commercial) settings because of the prevalence of informal, direct communication modes. OSS projects, on the other hand, use the internet as the communication medium, and typically conduct discussions in an open, public manner. As a result, the email archives of OSS projects provide a useful trace of the communication and co-ordination activities of the participants. However, there are various challenges that must be addressed before this data can be effectively mined. Once this is done,

1. INTRODUCTION
Large-scale software development projects invariably require a lot of communication and coordination (C&C) amongst the project workers. We distinguish these activities from engineering activities, where actual artifacts such as source code or documents are modified. The difficulty and intensity of the required coordination effort is quite high; this is often cited as the reason why adding more developers doesn't necessarily speed-up development [4]. C&C activities influence (and are influenced by) the design, structure and evolution of software systems. In traditional, commercial software organization, C&C activities may occur informally and would be difficult to study. Even if coordination

About me

- Raj Reddy Associate Professor
- Societal Computing
- Socio-Technical Engineering
- Reading since May 2012



ms

ol 2012



Check for updates

When Does Code Change Happen?

Jacek Śliwerski
International Max Planck Research School
Max Planck Institute for Computer Science
Saarbrücken, Germany
sliwers@mpi-sb.mpg.de

ABSTRACT

As a software system evolves, programmers make changes that sometimes cause problems. We analyze CVS archive history to find fix-inducing changes—changes that lead to problems, indicating a version archive (such as CVS) to a bug database (such as BUGZILLA). In a first investigation of the MOZILLA codebase, it turns out that fix-inducing changes show a clear pattern with respect to their size and the day of week they were made.

networks*

Anand Swaminathan
Graduate School of Management,
University of California, Davis,
Davis, California Republic.
swaminathan@ucdavis.edu

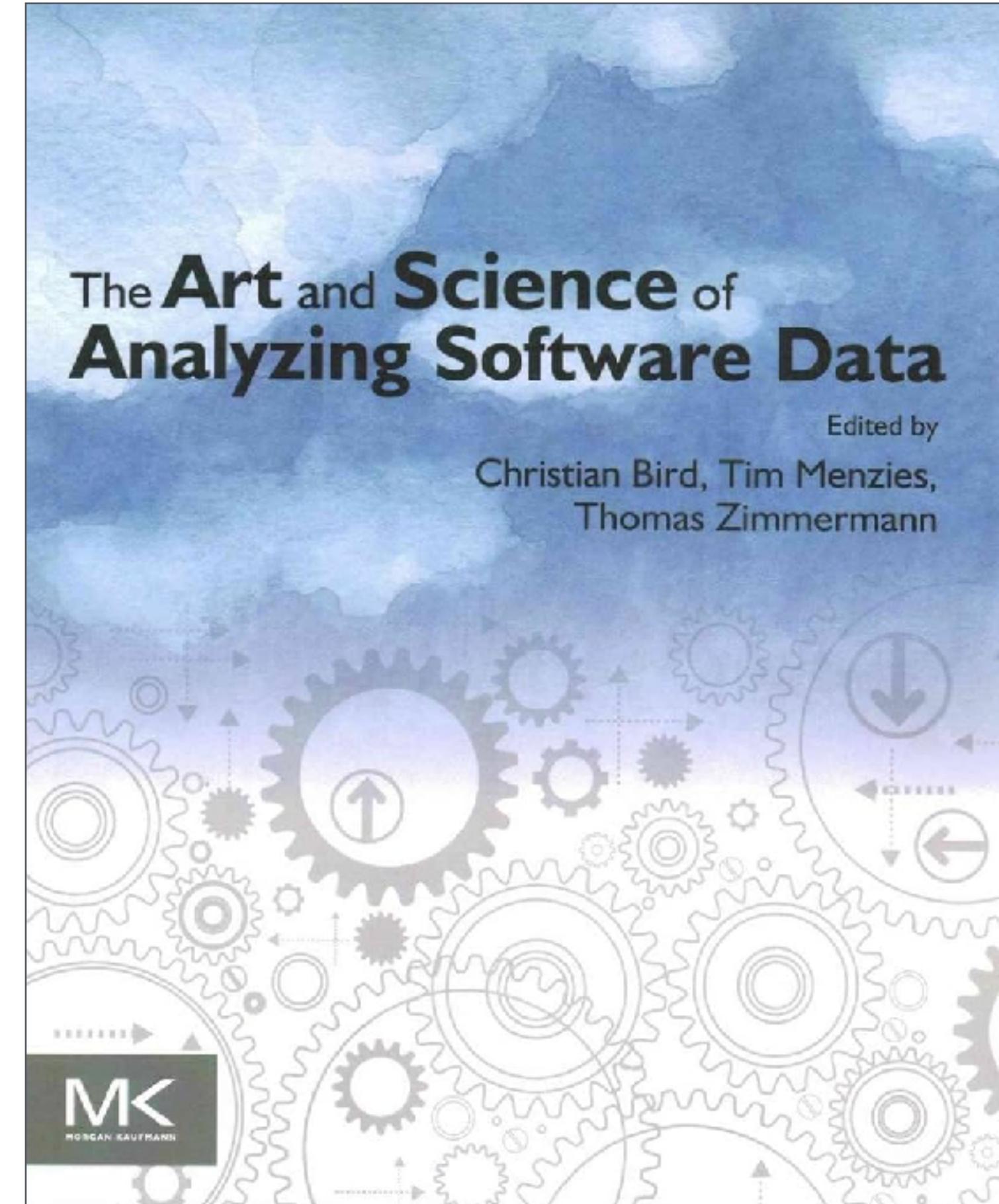
INTRODUCTION

Large scale software development projects invariably result in a complex web of communication and coordination (C&C) among the project workers. We distinguish these activities from engineering activities, where actual artifacts such as code or documents are modified. The difficulty and complexity of the required coordination effort is quite high; this has often been cited as the reason why adding more developers does not necessarily speed-up development [4]. C&C activities are influenced by the design, structure and evolution of software systems. In traditional, commercial organization, C&C activities may occur informally and would be difficult to study. Even if coordination

There is no such thing as MSR!

MSR = Data Science + Software Engineering

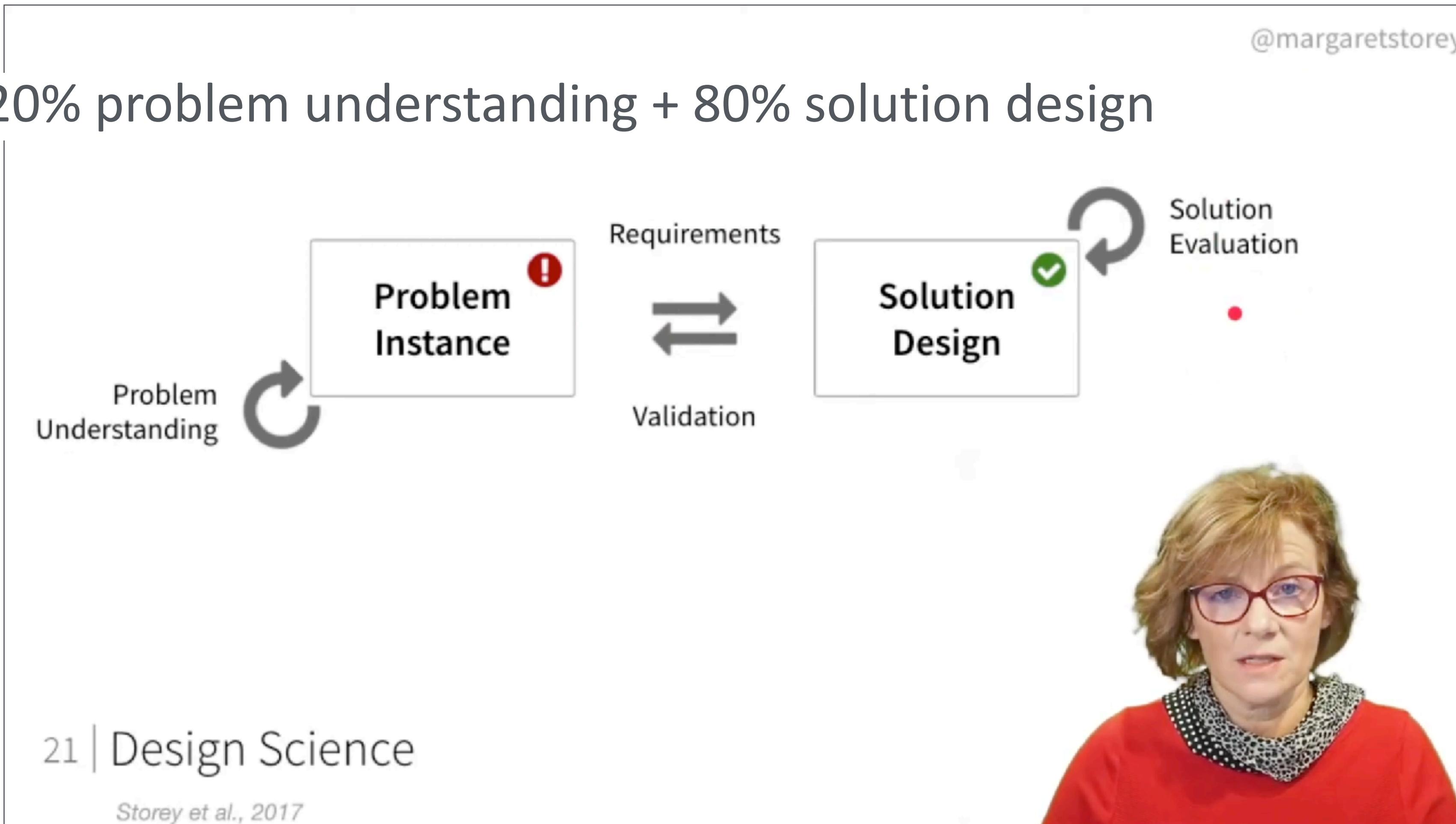
Table 1.3 Mapping Information Needs (Left) to Automatic Technique (Right)			
Information Need	Description	Insight	Relevant Techniques
Summarization	Search for important or unusual factors associated with a time range.	Characterize events, understand why they happened.	Topic analysis, NLP
Alerts (& Correlations)	Continuous search for unusual changes or relationships in variables	Notice important events.	Statistics, Repeated measures
Forecasting	Search for and predict unusual events in the future based on current trends.	Anticipate events.	Extrapolation, Statistics
Trends	How is an artifact changing?	Understand the direction of the project.	Regression analysis
Overlays	What artifacts account for current activity?	Understand the relationships between artifacts.	Cluster analysis, repository mining
Goals	How are features/artifacts changing in the context of completion or some other goal?	Assistance for planning	Root-cause analysis
Modeling	Compares the abstract history of similar artifacts. Identify important factors in history.	Learn from previous projects.	Machine learning
Benchmarking	Identify vectors of similarity/difference across artifacts.	Assistance for resource allocation and many other decisions	Statistics
Simulation	Simulate changes based on other artifact models.	Assistance for general decisions	What-if? analysis



MSR is, more or less, quantitative empirical software engineering?

ICSE 2017: 20% problem understanding + 80% solution design

@margaretstorey



- Storey, Ernst, et al. "The who, what, how of software engineering research: a socio-technical framework." EMSE 2020. Talk: <https://youtu.be/fs2XhM5-zXI>

Where is MSR now?

How have we changed in the last 10 years?

How to increase the impact of our work?

MSR 2015 (Florence) vs MSR 2025 (Ottawa)

The image shows two side-by-side screenshots of conference websites. The top half displays the MSR 2015 website for Florence, Italy, featuring a blue header with the title 'MSR 2015', the date 'May 16–17. Florence, Italy', and the subtitle 'The 12th Working Conference on Mining Software Repositories'. It includes a cartoon character of a person mining data, a co-location logo for ICSE 2015, and a navigation menu with links to Home, Program, Mining Challenge, Data Showcase, Registration, Venue, Hall Of Fame, and Organization. The background features a scenic view of a European city at sunset. The bottom half displays the MSR 2025 website for Ottawa, Canada. It has a similar layout with a blue header, the title 'MSR 2025', the subtitle '22nd International Conference on Mining Software Repositories', the date 'April 28-29, Ottawa, Canada', and a co-location logo for ICSE 2025. It also features a cartoon character of a person mining data, a Canadian maple leaf, and a navigation menu with links to Attending, Program, Tracks, Organization, Search, Series, Sign in, and Sign up. The background features a stylized graphic of a person mining data.

MSR 2015
May 16–17. Florence, Italy
The 12th Working Conference on Mining Software Repositories

co-located with
ICSE 2015
Firenze, Italy

Home Program Mining Challenge Data Showcase Registration Venue Hall Of Fame Organization

MSR 2025
22nd International Conference on
Mining Software Repositories
April 28-29, Ottawa, Canada

co-located with ICSE 2025

Attending ▾ Program ▾ Tracks ▾ Organization ▾ Search Series ▾ Sign in Sign up

MSR 2015 (Florence) vs MSR 2025 (Ottawa)

No proceedings, no preprints -\(\times\)/-

The image shows two side-by-side conference websites: MSR 2015 (Florence) and MSR 2025 (Ottawa).

MSR 2015 (Florence): The top half features a banner image of a building at sunset. The header includes "MSR 2015", "May 16–17. Florence, Italy", "The 12th Working Conference on Mining Software Repositories", and a cartoon character. It also mentions "co-located with ICSE 2015 Firenze, Italy". The navigation menu includes Home, Program, Mining Challenge, Data Showcase, Registration, Venue, Hall Of Fame, and Organization.

MSR 2025 (Ottawa): The bottom half features a banner image of a building at sunset. The header includes "MSR 2025", "22nd International Conference on Mining Software Repositories", "April 28-29, Ottawa, Canada", and a logo featuring a beaver, a maple leaf, and a person. It also mentions "co-located with ICSE 2025". The navigation menu includes Attending, Program, Tracks, Organization, Search, Series, Sign in, and Sign up.

MSR 2015 (Florence) vs MSR 2024 (Lisbon)

The image shows two side-by-side screenshots of conference websites. The left screenshot is for MSR 2015 in Florence, Italy, featuring a banner with a sunset over a bridge, navigation links for Home, Program, Mining Challenge, Data Showcase, Registration, Venue, Hall Of Fame, and Organization, and logos for co-location with ICSE 2015. The right screenshot is for MSR '24 in Lisbon, Portugal, featuring a banner with a cartoon character, navigation links for Attending, Program, Tracks, Organization, Search, and Series, and text for the 21st International Conference on Mining Software Repositories.

MSR 2015
May 16–17. Florence, Italy
The 12th Working Conference on Mining Software Repositories

co-located with
ICSE 2015
Firenze, Italy

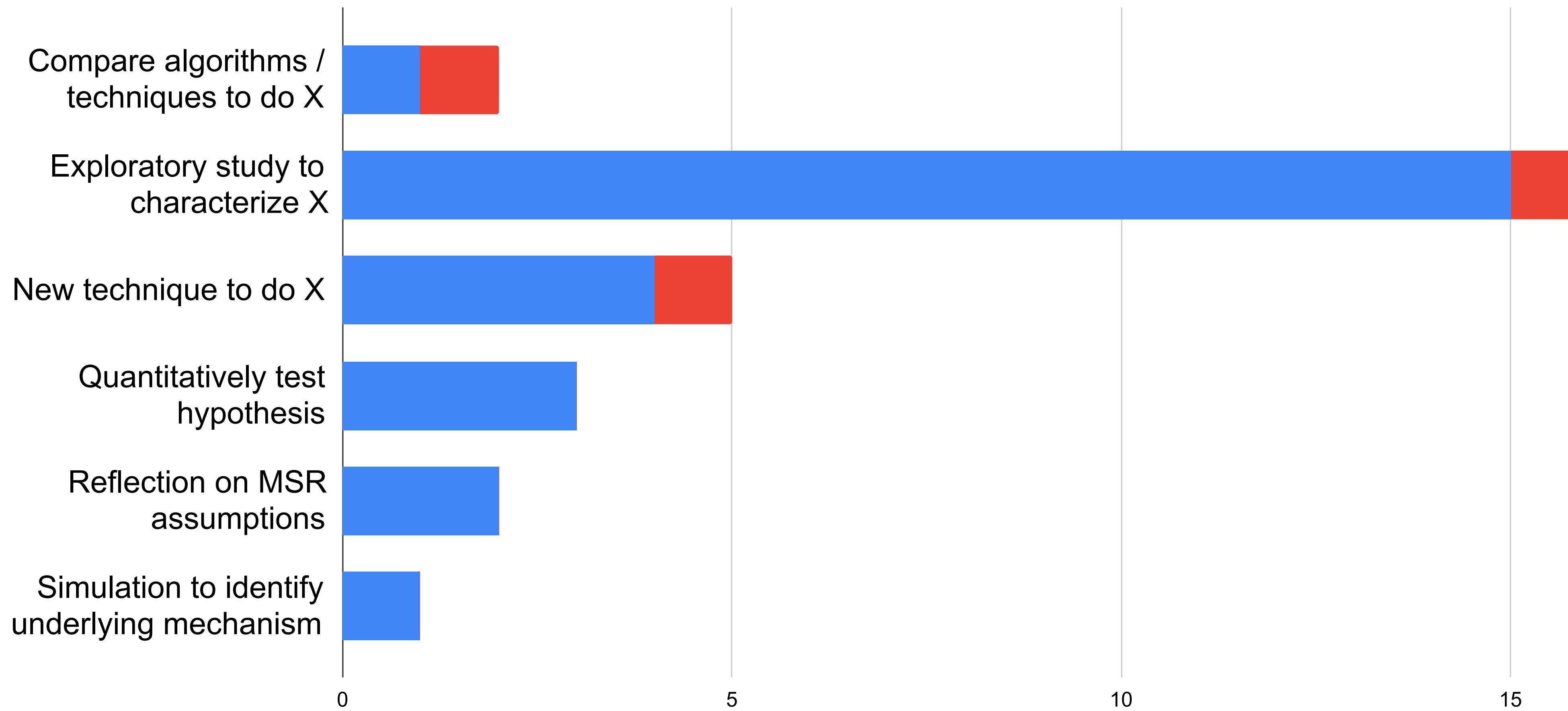
Home Program Mining Challenge Data Showcase Registration Venue Hall Of Fame Organization

MSR '24
21st INTERNATIONAL CONFERENCE ON
MINING SOFTWARE REPOSITORIES
April 15-16, Lisbon, Portugal

Attending Program Tracks Organization Search Series

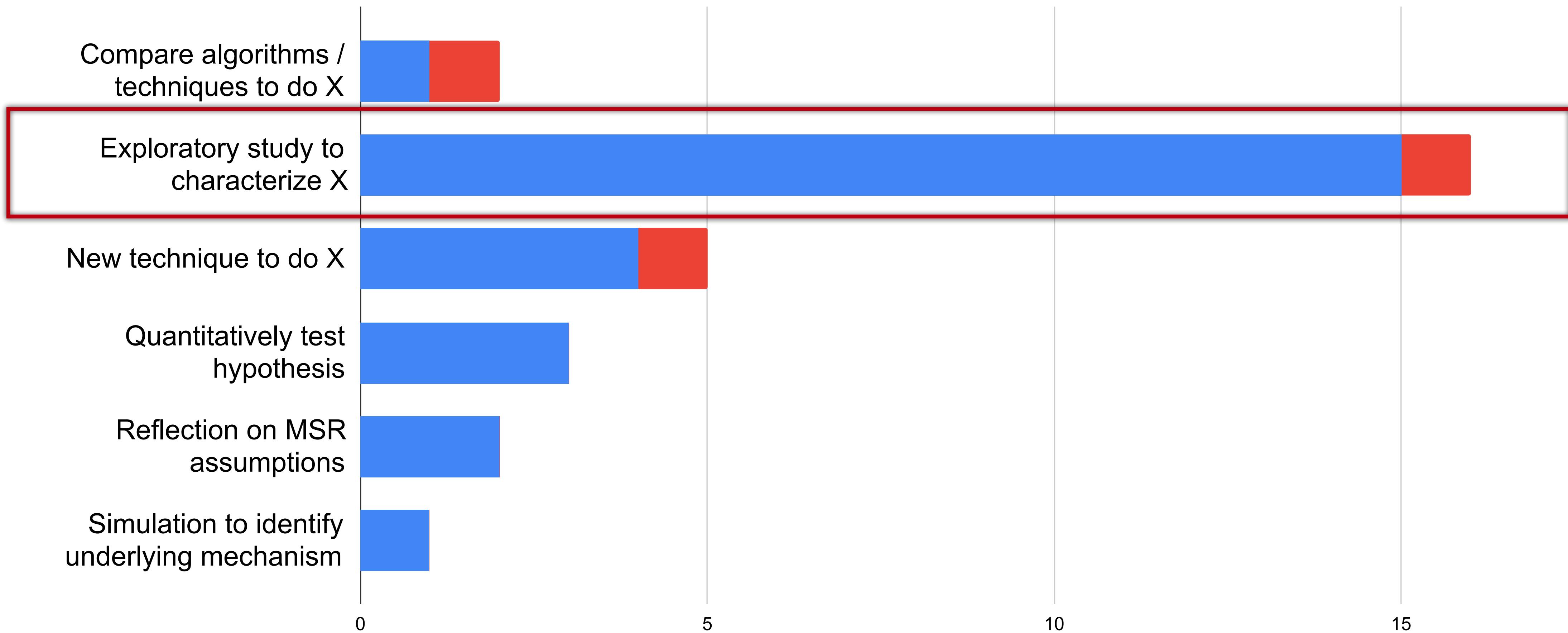
MSR 2015 (Florence): 29 full papers

Types of 10-page papers at MSR 2015 (Florence)



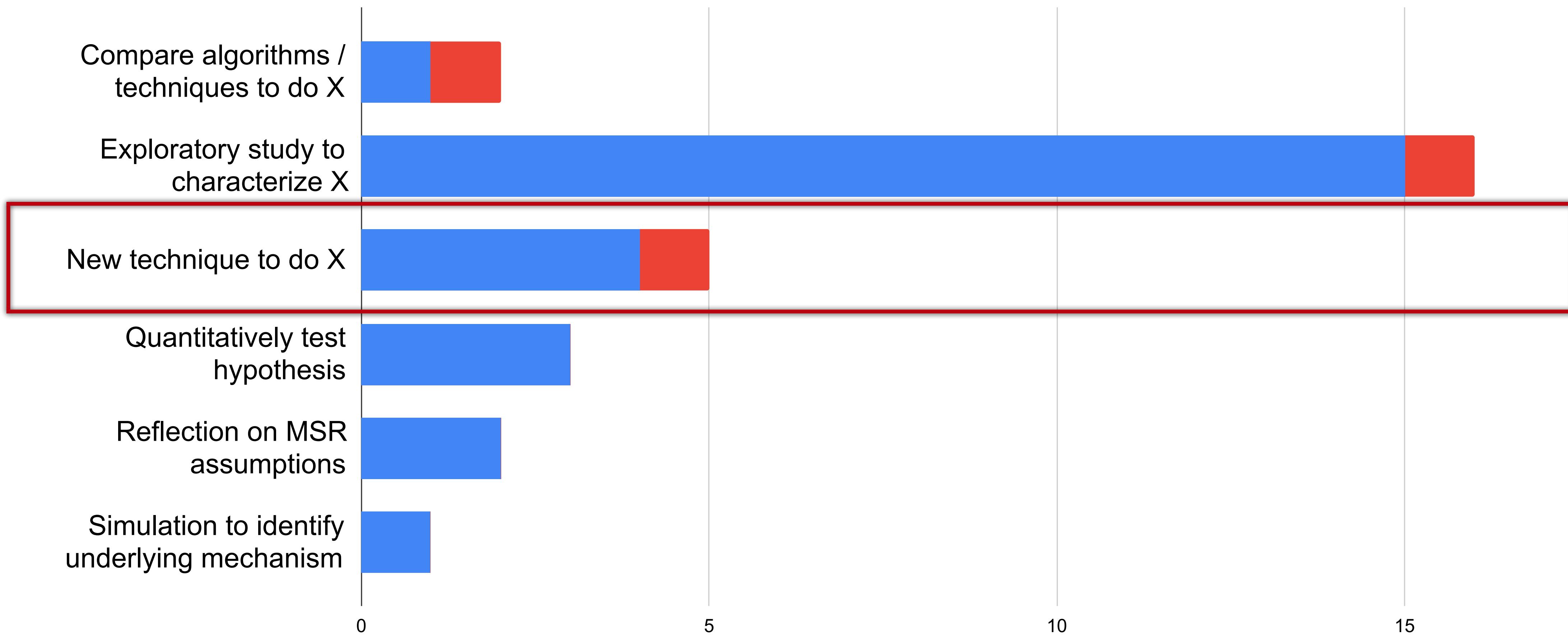
MSR 2015 (Florence): 29 full papers

Types of 10-page papers at MSR 2015 (Florence)



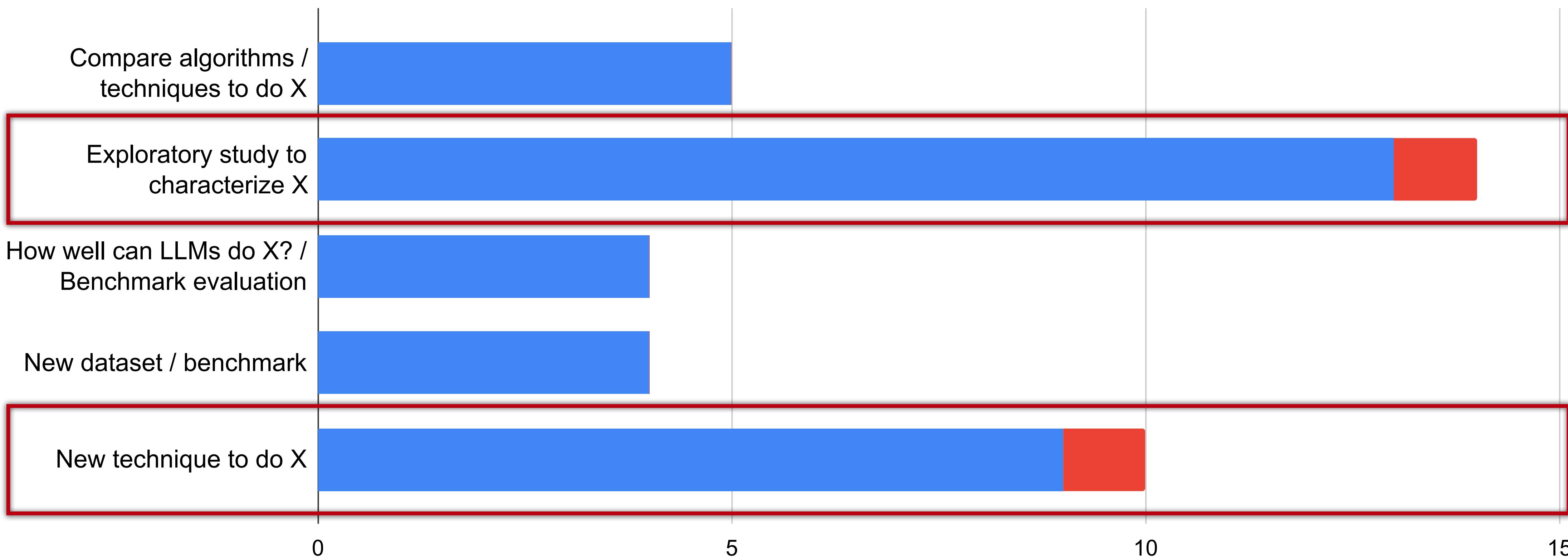
MSR 2015 (Florence): 29 full papers

Types of 10-page papers at MSR 2015 (Florence)



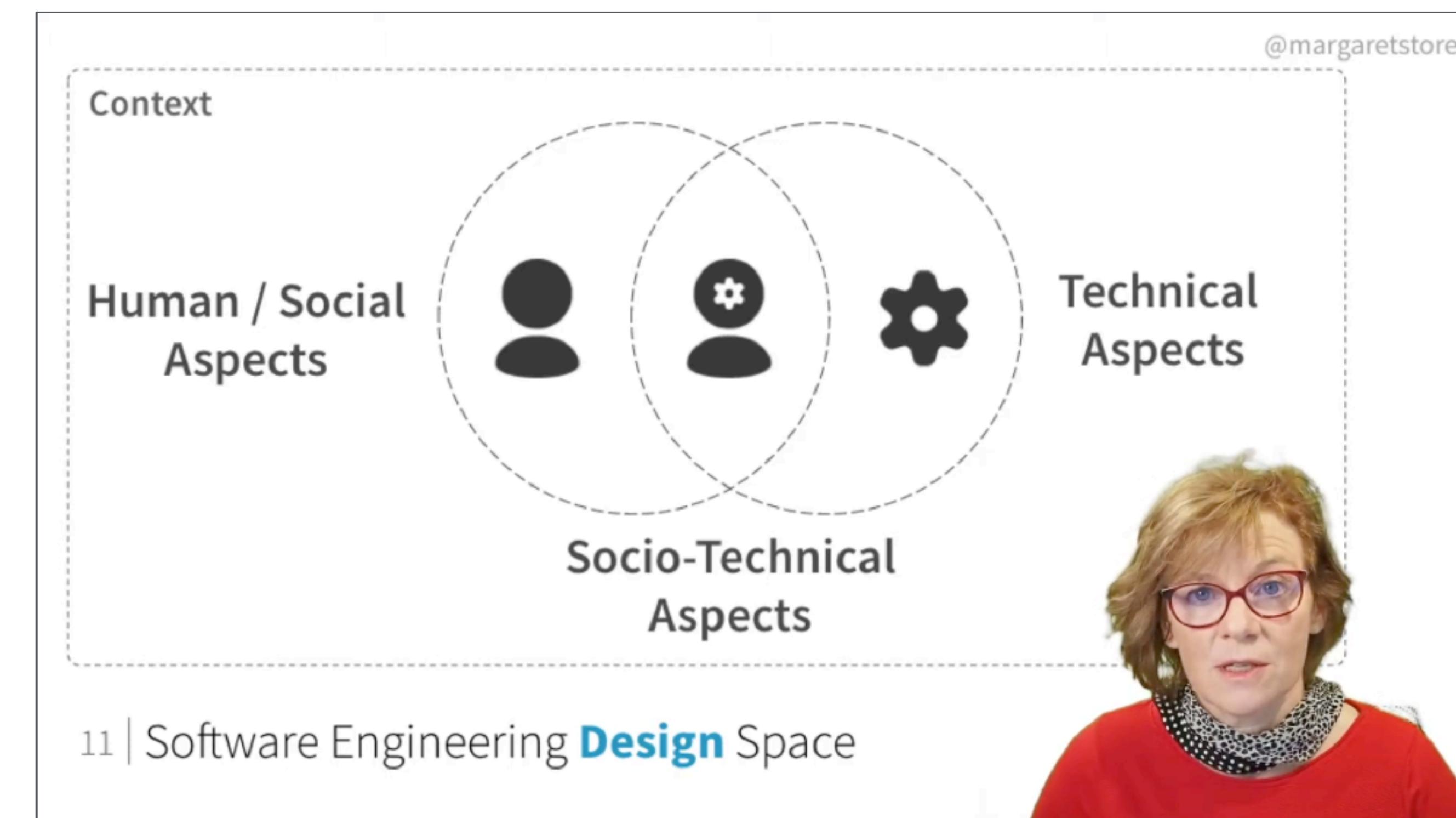
MSR 2024 (Lisbon): 37 full papers

Types of 10-page papers at MSR 2024 (Lisbon)



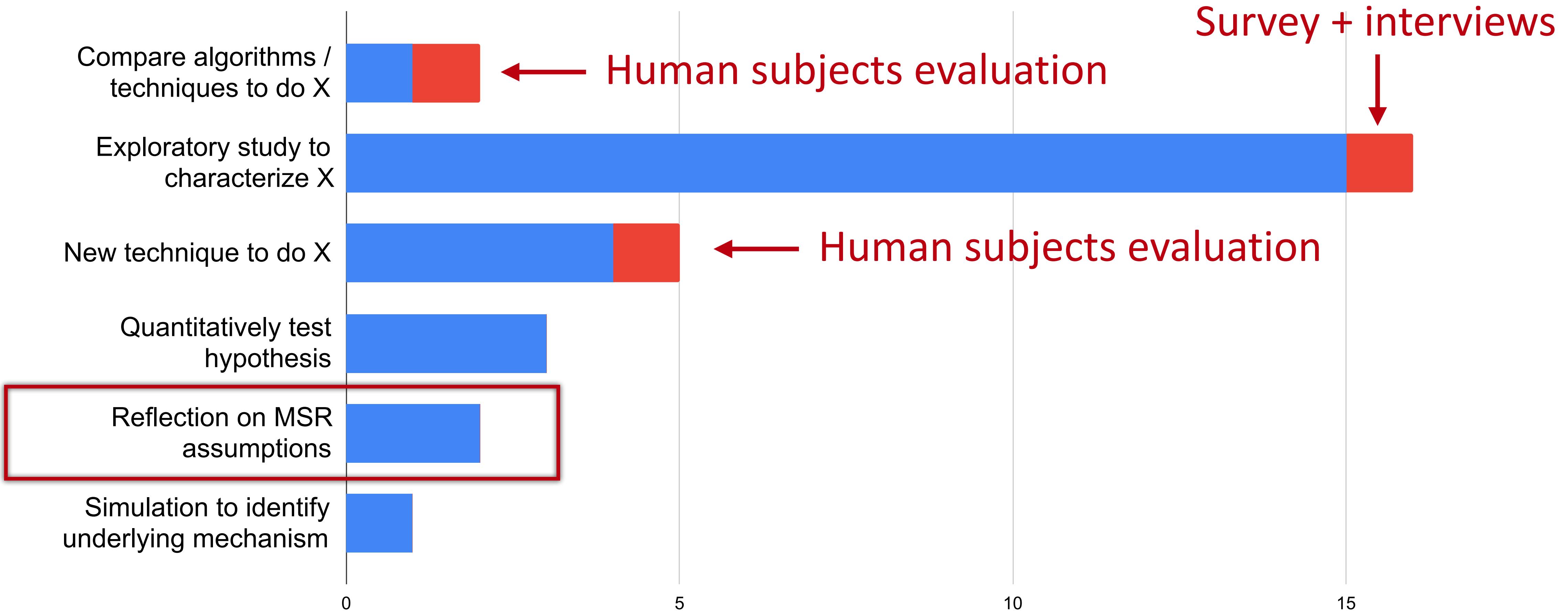
Lots written about high-quality, impactful SE research already

- Storey, Ernst, et al. “The who, what, how of software engineering research: a socio-technical framework.” EMSE 2020.
 - Argument to increase impact by increasing the emphasis on humans



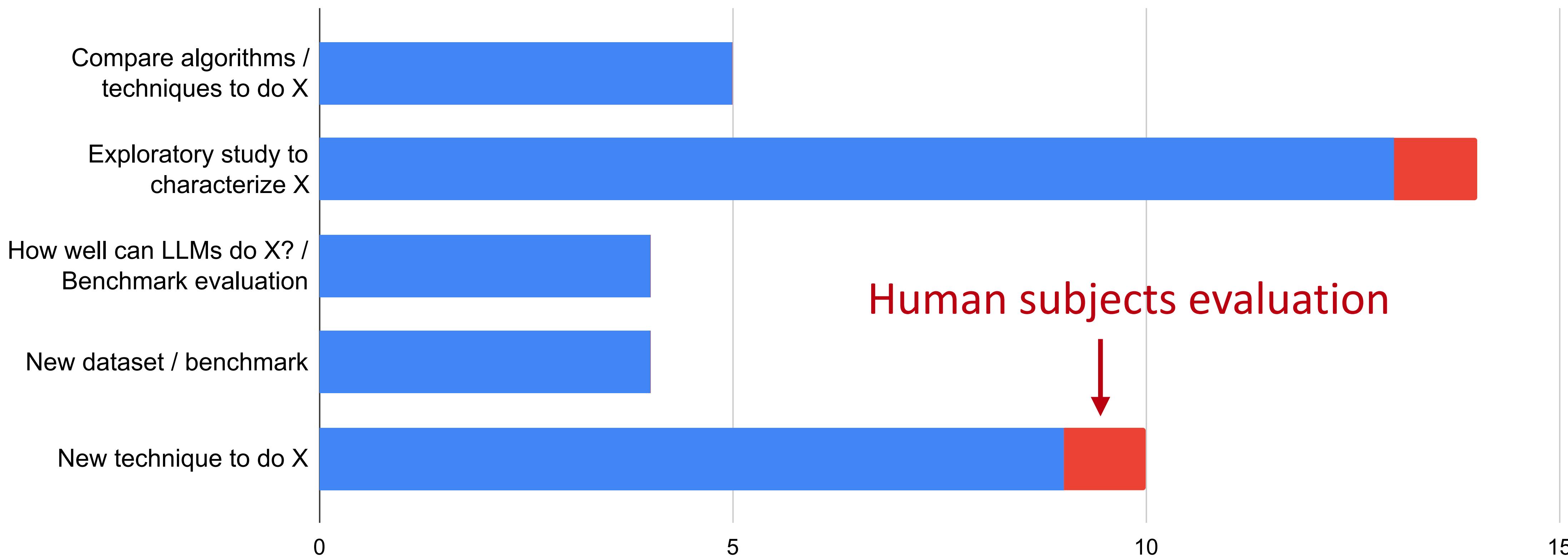
MSR 2015 (Florence): 29 full papers

Types of 10-page papers at MSR 2015 (Florence)



MSR 2024 (Lisbon): 37 full papers

Types of 10-page papers at MSR 2024 (Lisbon)



Lots written about high-quality, impactful SE research already

- Mary Shaw, “Writing Good Software Engineering Research Papers.” 2003
 - “Why should the reader believe your result?”
 - “What concrete evidence shows that your result satisfies your claim?”(Among many others)
- Laurie Williams & colleagues, “Writing Good Software Engineering Research Papers: Revisited.” 2017



Still, perception that MSR research is shallow ...

- In big data some patterns and associations are always visible
- Data doesn't mean insights
- “So what?”
- Etc

We have known a solution for over 20 years



University of Toronto

Department of Computer Science

You Gotta Have A Theory

Steve Easterbrook
sme@cs.toronto.edu
www.cs.toronto.edu/~sme

© 2004-5 Steve Easterbrook. This presentation is available free for non-commercial use with attribution under a creative commons license.

1

- Easterbrook. FSE 2006 Doctoral Symposium

The same argument reappears from time to time

Science of Computer Programming 101 (2015) 79–98

 ELSEVIER

Contents lists available at [ScienceDirect](#)

Science of Computer Programming

www.elsevier.com/locate/scico



Theory-oriented software engineering 

Klaas-Jan Stol*, Brian Fitzgerald

Lero – The Irish Software Engineering Research Centre, University of Limerick, Ireland

 CrossMark

@SCP 2015

The same argument reappears from time to time

Science of Computer Programming 101 (2015) 79–98

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scpro

Check for updates

ELSEVIER

Theory-oriented software engineering 

Klaas-Jan Stol*, Brian Fitzgerald

Lero – The Irish Software Engineering Research Centre, University of Limerick, Ireland

@SCP 201

**Building a Socio-Technical Theory of Coordination:
Why and How (Outstanding Research Award)**

James Herbsleb
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
+1 412 268 8933
jdh@cs.cmu.edu

ABSTRACT
Research aimed at understanding and addressing coordination breakdowns experienced in global software development (GSD) projects at Lucent Technologies took a path from open-ended qualitative exploratory studies to quantitative studies with a tight focus on a key problem – delay – and its causes. Rather than being directly associated with delay, multi-site work items involved more people than comparable same-site work items, and the number of people was a powerful predictor of delay. To counteract this we developed and deployed tools and practices to or innovate. When people organize in a habitual, consistent way, for example, in collocated teams, it is easy to overlook day-to-day coordination mechanisms that are simply taken for granted. It is easy to see the importance of things such as meetings of various flavors, processes, methods, and architectural separation, which have long been studied. Other, subtler mechanisms such as informal communication, practices, habits, and shared mental models are often only made visible by their absence.

Very interesting – and often disturbing – things happen when an

@FSE 2016

The same argument reappears from time to time

The collage consists of three vertically stacked screenshots:

- Dagstuhl Seminar 22231: Theories of Programming (Jun 06 – Jun 10, 2022)**: A group photo of seminar participants standing on the steps of a building with arched windows. The image is licensed under Creative Commons License CC BY-NC-ND.
- Science of Computer Programming 101 (2015) 79–98**: A screenshot of a journal page. It features a tree logo, a ScienceDirect link, and a thumbnail of the journal cover.
- Building a Socio-Technical Theory of Coordination: Why and How (Outstanding Research Award)**: A screenshot of a research paper abstract. It includes the title, author information (James Herbsleb, Carnegie Mellon University), contact details (+1 412 268 8933, jdh@cs.cmu.edu), an abstract section, and a concluding paragraph about innovation and coordination mechanisms.

@Dagstuhl 2022

@FSE 2016

We already have lots of CS-related theories

- **Statistical theory** enables proper hypothesis testing and confidence intervals
- **Information theory** guides efficient data encoding and compression
- **Linear algebra** and **calculus** form the backbone of most machine learning models
- **Optimization theory** guides efficient model training approaches
- Etc.

But not enough good theories about SE processes and stakeholder behavior

- A theory is a set of **propositions** that are logically related, expressing the **relation(s)** among several different **constructs** and **propositions**.
- Theories are the building blocks of scientific knowledge.

But not enough good theories about SE processes and stakeholder behavior

- A theory is a set of **propositions** that are logically related, expressing the **relation(s)** among several different **constructs** and **propositions**.
- Theories are the building blocks of scientific knowledge.

The slide features a light gray background with a white rectangular content area. In the top left corner of the content area is a small icon with a blue circle, a red square, and a white triangle, with the text 'Check for updates' below it. The title 'When Do Changes Induce Fixes?' is centered in large black font. Below the title, in smaller black font, is '(On Fridays.)'. To the right of the title, the text '@MSR 2005' is written in red. A horizontal red arrow points from the title area towards the bottom right. At the bottom left, there is a block of text for Jacek Śliwerski, and at the bottom right, there is a block of text for Thomas Zimmermann and Andreas Zeller. Both blocks include their names, titles, institutions, and email addresses.

When Do Changes Induce Fixes?
(On Fridays.) 

@MSR 2005

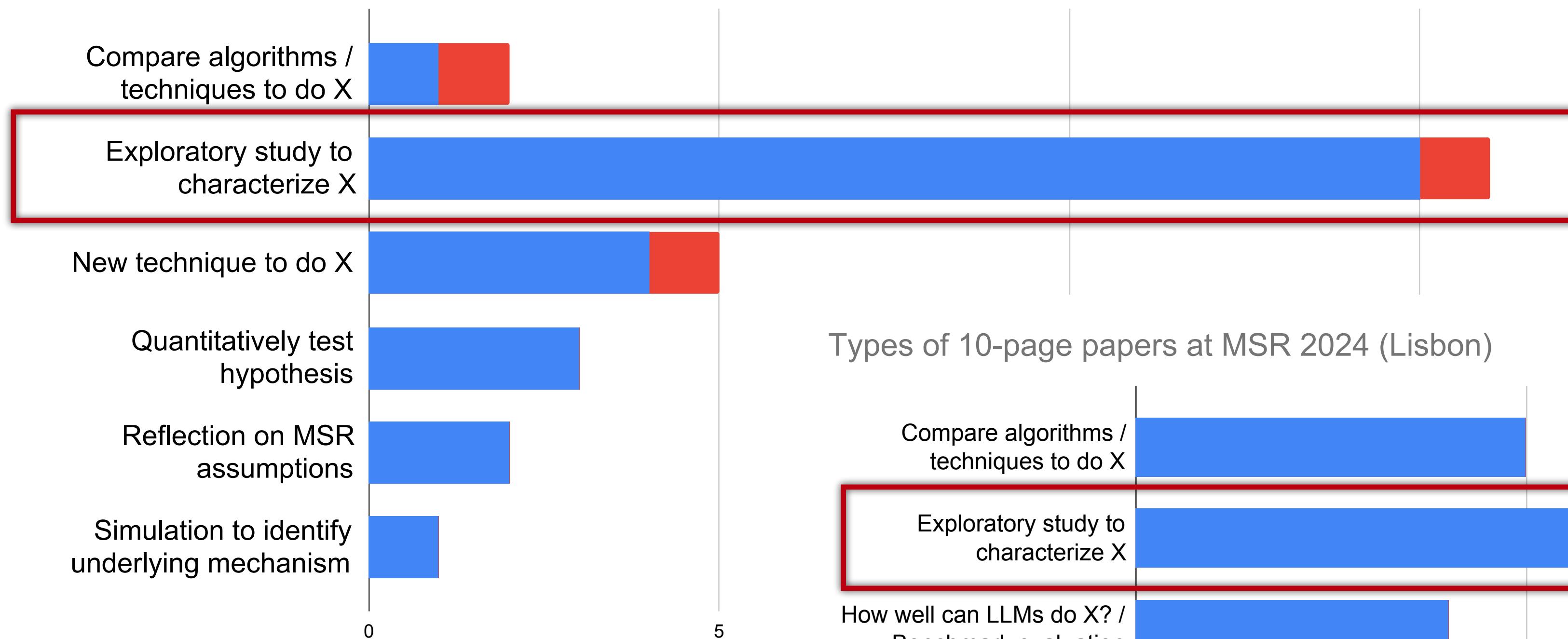

Check for updates

Jacek Śliwerski
International Max Planck Research School
Max Planck Institute for Computer Science
Saarbrücken, Germany
sliwers@mpi-sb.mpg.de

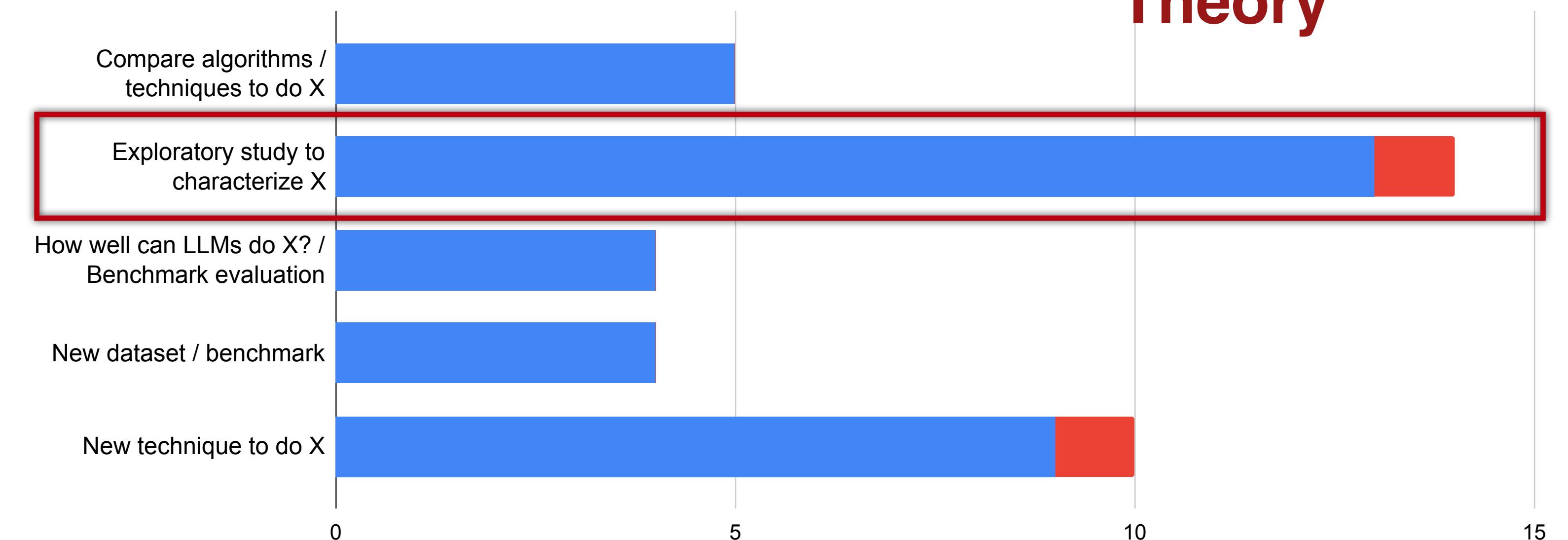
Thomas Zimmermann Andreas Zeller
Department of Computer Science
Saarland University
Saarbrücken, Germany
{tz, zeller}@acm.org

“What are you talking about? We have tons of theories”

Types of 10-page papers at MSR 2015 (Florence)



Types of 10-page papers at MSR 2024 (Lisbon)



Data collection



Interpretation of findings



Propositions / Hypotheses



Theory

SBOMs are a kind of theory

Concern: Evolving of SBOM "Definition"

The slide displays four versions of the 'Framing Software Component Transparency: Establishing a Common Software Bill of Materials (SBOM)' document, each with a different date and a corresponding image below it.

- 2019/11:** The image shows a grid of rectangles representing software components.
- 2021/7:** The image shows a document cover with the NTIA logo.
- 2021/10:** The image shows a document cover with a background of many small, overlapping rectangles.
- 2024/9:** The image shows a document cover with a background of many small, overlapping rectangles.

Source: https://www.ntia.gov/files/ntia/publications/framingbom_20191117.pdf

Source: https://www.ntia.gov/files/ntia/publications/ntia_sbom_minimum_elements_report.pdf

Source: https://www.ntia.gov/files/ntia/publications/ntia_sbom_framing_2nd_edition_20211021.pdf

Source: <https://www.cisa.gov/sites/default/files/2024-10/SBOM%20Software%20Component%20Transparency%202024.pdf>

License: CC BY-SA 4.0

Attribute	SPDX	CycloneDX	SWID
Author Name	C-33: CycloneDX	SPDX-1.2.1: Author	SWID-1.0: Author
Timestamp	C-33: CycloneDX	SPDX-1.2.1: Timestamp	SWID-1.0: Timestamp
Supplier Name	C-33: CycloneDX	SPDX-1.2.1: SupplierName	SWID-1.0: SupplierName
Component Name	C-33: CycloneDX	SPDX-1.2.1: ComponentName	SWID-1.0: ComponentName
Version of the Component	C-33: CycloneDX	SPDX-1.2.1: Version	SWID-1.0: Version
Other Unique Identifiers	C-33: CycloneDX	SPDX-1.2.1: OtherIdentifiers	SWID-1.0: OtherIdentifiers
Dependency Relationship	C-33: CycloneDX	SPDX-1.2.1: Relationship	SWID-1.0: Relationship
Author of KBOM Data	C-33: CycloneDX	SPDX-1.2.1: Author	SWID-1.0: Author
Timestamp	C-33: CycloneDX	SPDX-1.2.1: Timestamp	SWID-1.0: Timestamp
Component Hash	C-33: CycloneDX	SPDX-1.2.1: Hash	SWID-1.0: Hash
Unique Identifier	C-33: CycloneDX	SPDX-1.2.1: Identifier	SWID-1.0: Identifier
Relationship	C-33: CycloneDX	SPDX-1.2.1: Relationship	SWID-1.0: Relationship

Table 1: Mapping NIST SBOM component information to existing formats

But not enough good theories about SE processes and stakeholder behavior

- A theory is a set of **propositions** that are logically related, expressing the **relation(s)** among several different **constructs** and **propositions**.
- Theories are the building blocks of scientific knowledge.
- A theory that describes a phenomenon is a valid theory.
- A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.

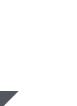
Theory



Hypotheses



Data collection



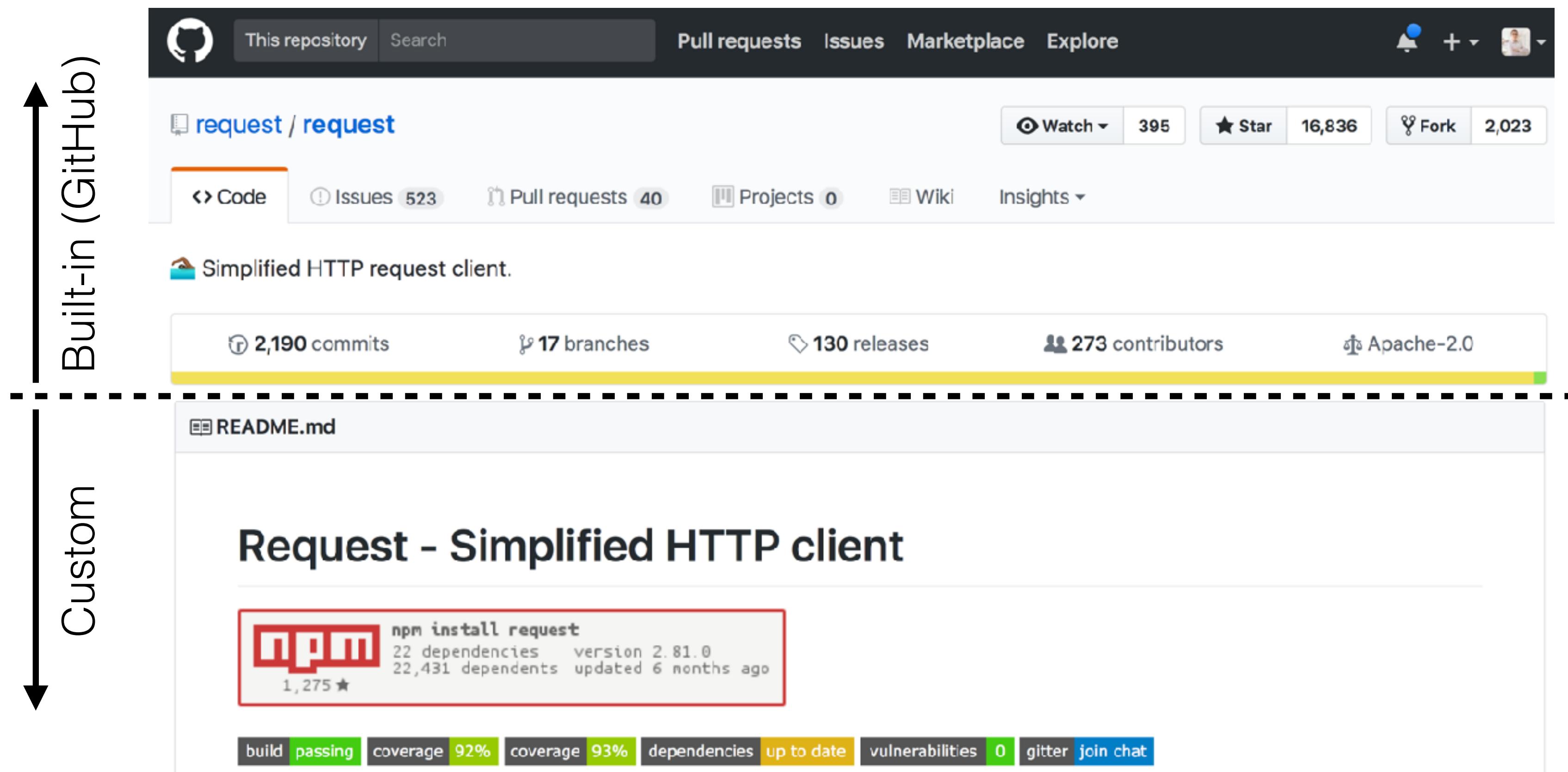
Interpretation of findings



Validation / Refinement

Example: Signaling theory (Spence, 1973)

People use the visible cues on the platform as **signals**, to make risk inferences about unobservable traits of other users or projects.

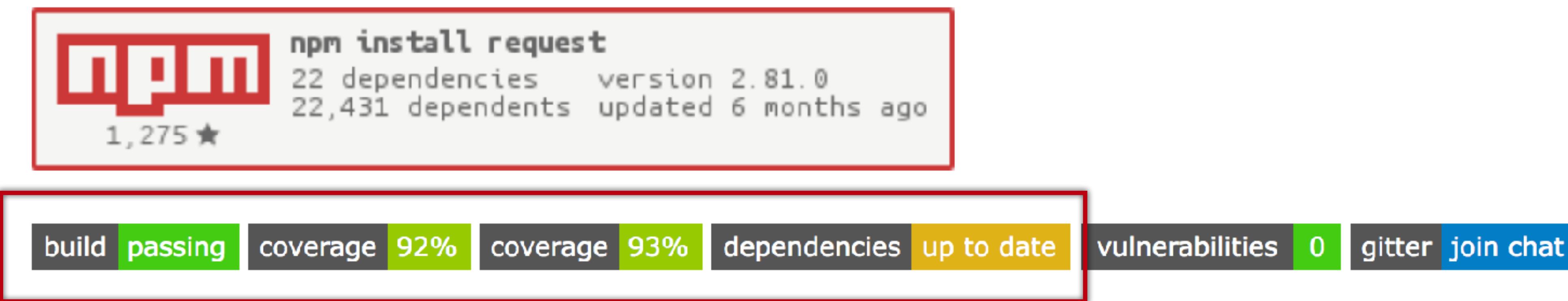


- Trockman, Zhou, Kästner, & Vasilescu. Adding sparkle to social coding: An empirical study of repository badges in the npm ecosystem. ICSE 2018

Example: Signaling theory (Spence, 1973)

People use the visible cues on the platform as **signals**, to make risk inferences about unobservable traits of other users or projects.

Request - Simplified HTTP client

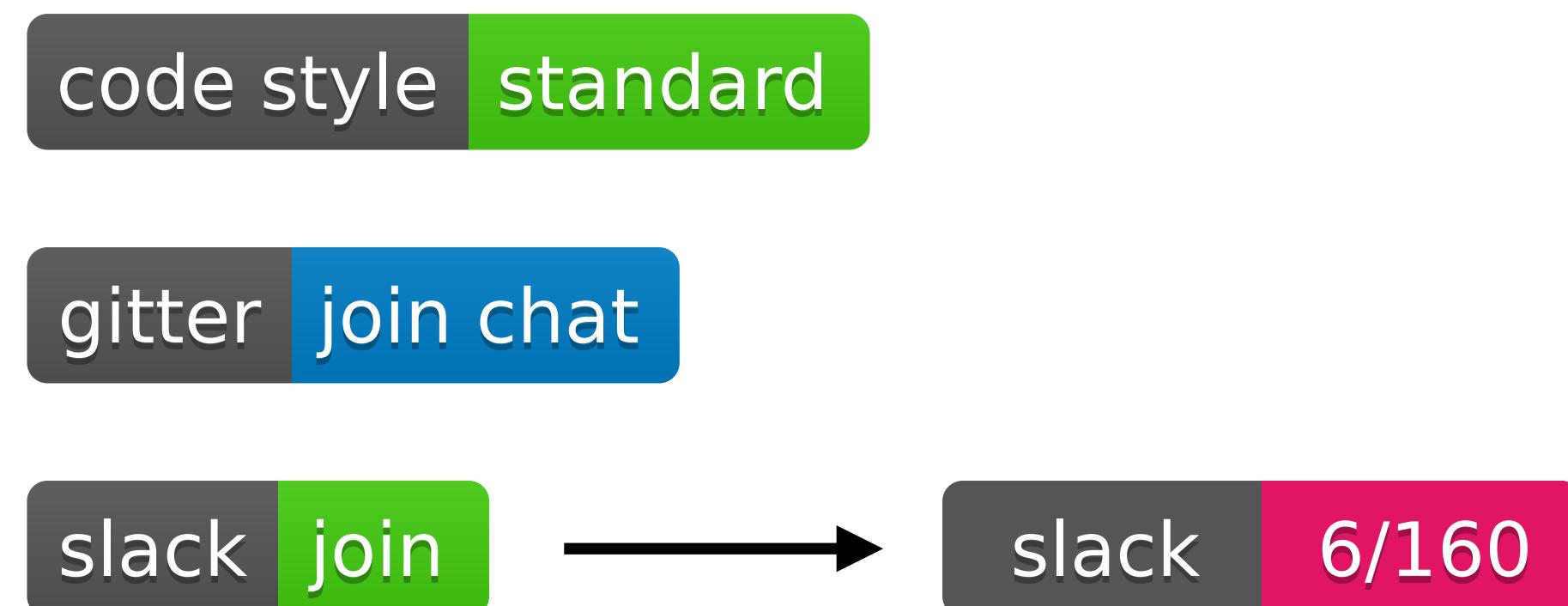


- Trockman, Zhou, Kästner, & Vasilescu. Adding sparkle to social coding: An empirical study of repository badges in the npm ecosystem. ICSE 2018

Example: Signaling theory (Spence, 1973)

“Assessment” vs “conventional” signals: the cost of producing the signal should result in the two types of badges having differential effects.

Harder to fake “assessment” badges provide more reliable signals.



- Trockman, Zhou, Kästner, & Vasilescu. Adding sparkle to social coding: An empirical study of repository badges in the npm ecosystem. ICSE 2018

But not enough good theories about SE processes and stakeholder behavior

- A theory is a set of **propositions** that are logically related, expressing the **relation(s)** among several different **constructs** and **propositions**.
 - Theories are the building blocks of scientific knowledge.
 - A theory that describes a phenomenon is a valid theory.
 - A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.
 - We don't have enough of these!
-
- ```
graph TD; Theory[Theory] --> Hypotheses[Hypotheses]; Hypotheses --> DataCollection[Data collection]; DataCollection --> Interpretation[Interpretation of findings]; Interpretation --> Validation[Validation / Refinement]
```

Theory



Hypotheses



Data collection



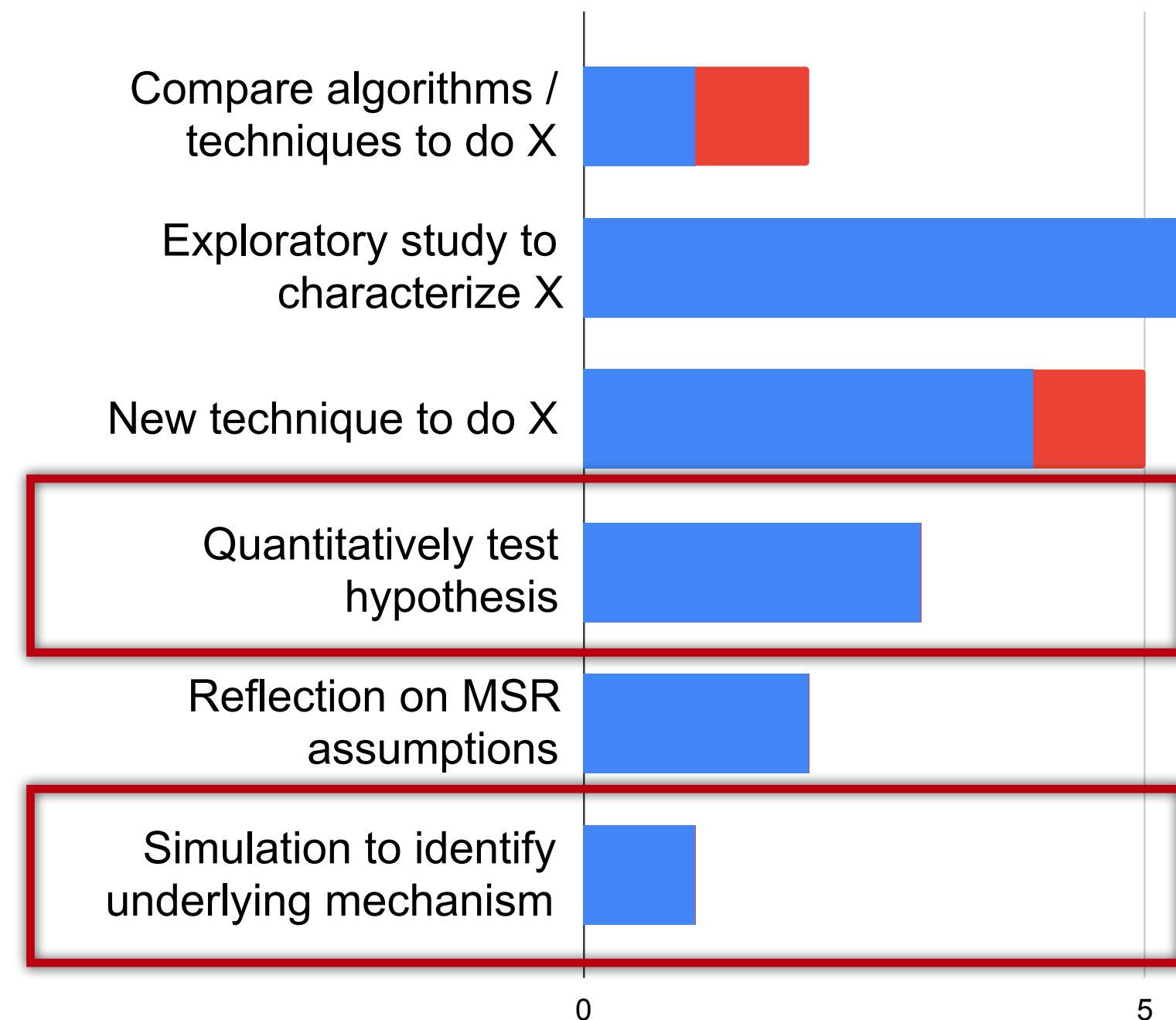
Interpretation of findings



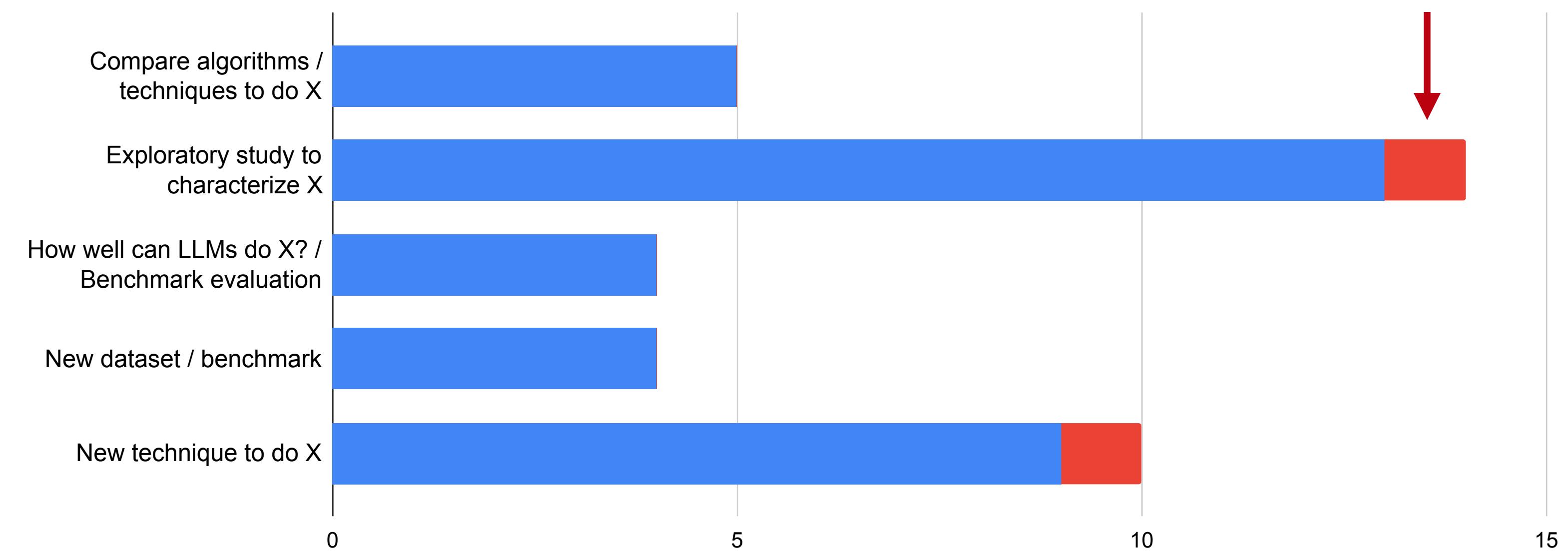
Validation / Refinement

# We have very little deductive use of theories

Types of 10-page papers at MSR 2015 (Florence)



Types of 10-page papers at MSR 2024 (Lisbon)



Tests specific hypothesis



# Is there space for this kind of theory in MSR?

Herbsleb is skeptical:

- “The universal principle of interdisciplinary contempt”
- “Intellectual worth is evaluated on a single dimension from math to BS”
- “Is that really computer science?”



**Building a Socio-Technical Theory of Coordination:  
Why and How (Outstanding Research Award)**

James Herbsleb  
Carnegie Mellon University  
5000 Forbes Avenue  
Pittsburgh, PA 15213  
+1 412 268 8933  
jdh@cs.cmu.edu

**ABSTRACT**  
Research aimed at understanding and addressing coordination breakdowns experienced in global software development (GSD) projects at Lucent Technologies took a path from open-ended qualitative exploratory studies to quantitative studies with a tight

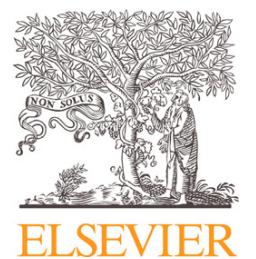
or innovate. When people organize in a habitual, consistent way, for example, in collocated teams, it is easy to overlook day-to-day coordination mechanisms that are simply taken for granted. It is easy to see the importance of things such as meetings of various flavors, processes, methods, and architectural separation, which

So are Menzies & Shepperd:

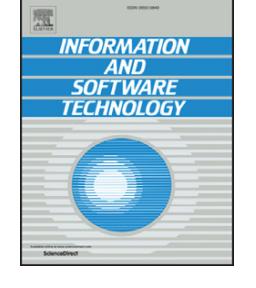
- “Data analytics studies are almost always theory light because they’re inductive in their approach.”



Information and Software Technology 112 (2019) 35–47



Contents lists available at ScienceDirect  
**Information and Software Technology**  
journal homepage: [www.elsevier.com/locate/infsof](http://www.elsevier.com/locate/infsof)



“Bad smells” in software analytics papers

Tim Menzies <sup>a,\*</sup>, Martin Shepperd <sup>b</sup>

<sup>a</sup> Dept. of Computer Science North Carolina State University, USA  
<sup>b</sup> Brunel Software Engineering Lab (BSEL) Dept. of Computer Science Brunel University London UB8 3PH, UK



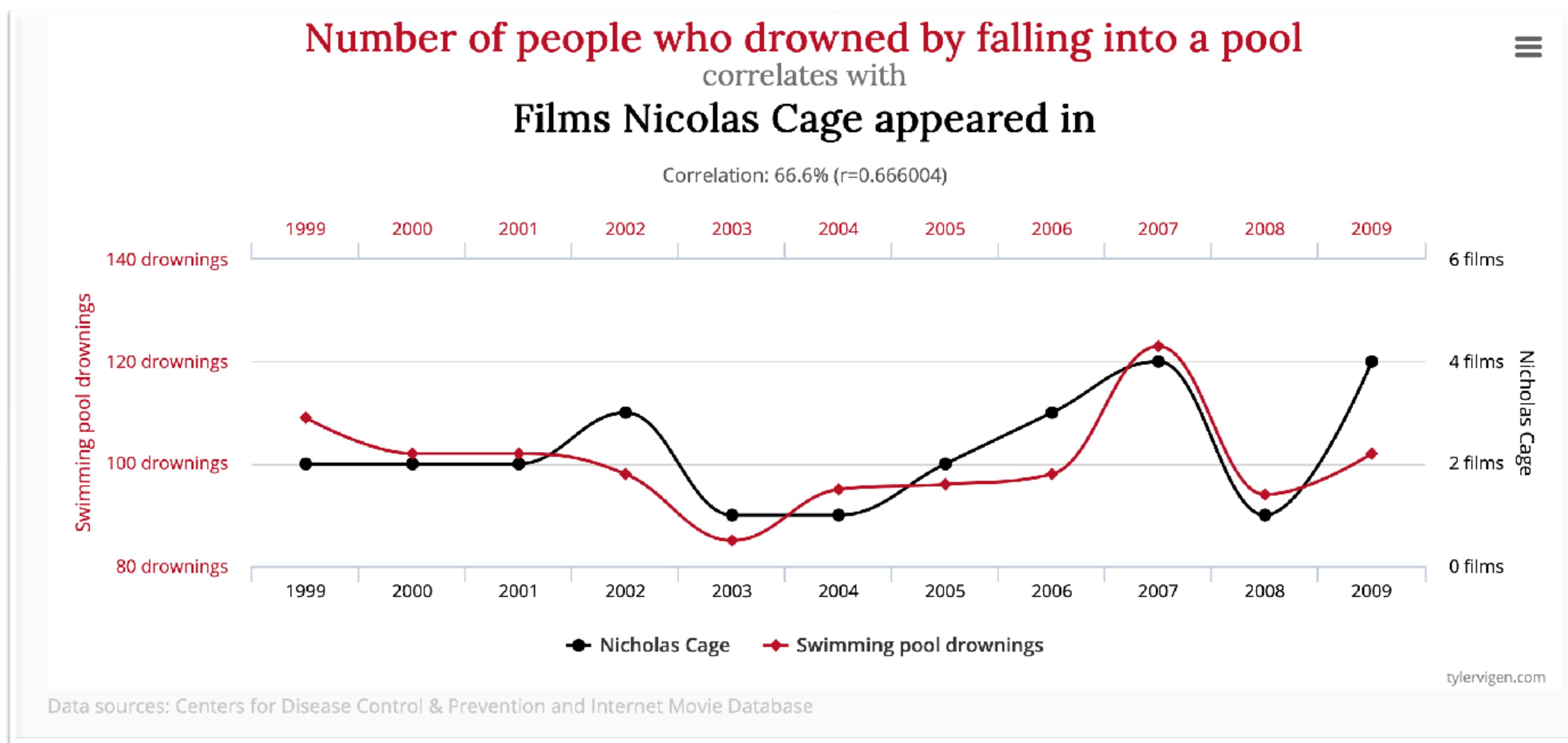
# Now what?

# Proposal: Let's establish more causal relationships

- A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.
  - Causal relationships allow for stronger predictions
  - Bonus points if we validate the mechanism

# Proposal: Let's establish more causal relationships

- A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.



# Ingredients for establishing a causal relationship?

Three properties must hold to establish a causal relationship between X and Y.

$X \rightarrow Y$  when:

- 
- 
-

# Ingredients for establishing a causal relationship?

Three properties must hold to establish a causal relationship between X and Y.

$X \rightarrow Y$  when:

- X precedes Y
- X and Y are correlated
- We can exclude plausible alternative explanations for Y other than X

# Proposal: Let's establish more causal relationships

- A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.
  - Causal relationships allow for stronger predictions
  - Bonus points if we validate the mechanism
- There are lots of techniques for **causal inference from observational data**.
  - We are up to date on AI tech but 20 years behind on research methods?
- MSR was always about methods
  - The name itself is a method!

# Example: Do tweets cause GitHub stars?

The image consists of two vertically stacked screenshots. The top screenshot is a Twitter post from Max Woolf (@minimaxir). It features a profile picture of Max, his name, and handle, followed by a blue Twitter bird icon. The tweet content is:  
I just released my new Python package: simpleaichat, an open-source tool for working with ChatGPT/GPT-4 with minimal code yet max flexibility!  
I built simpleaichat out of sheer frustration with LangChain and aim to make it the easiest way to make AI apps.

The bottom screenshot is a GitHub project page for `minimaxir/simpleaichat`. It shows the repository name, a profile picture of Max, a brief description: "Python package for easily interfacing with chat apps, with robust features and minimal code complexity.", and statistics: 3 contributors, 1 used by, 549 stars, and 22 forks. A GitHub logo is also present. Below the repository details, there's a snippet of the GitHub URL and a truncated description of the repository. At the bottom of the GitHub page, the timestamp "5:24 PM · Jun 8, 2023" and a reply/share section with "Read 18 replies" are visible.

- Fang, Lamba, Herbsleb, & Vasilescu. “This is damn slick!” Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

# Example: Do tweets cause GitHub stars?

Max Woolf (@minimaxir · Follow)  
I just released my new Python package: simpleaichat, an open-source tool for working with ChatGPT/GPT-4 with minimal code yet max flexibility!  
I built simpleaichat out of sheer frustration with LangChain and aim to make it the easiest way to make AI apps.

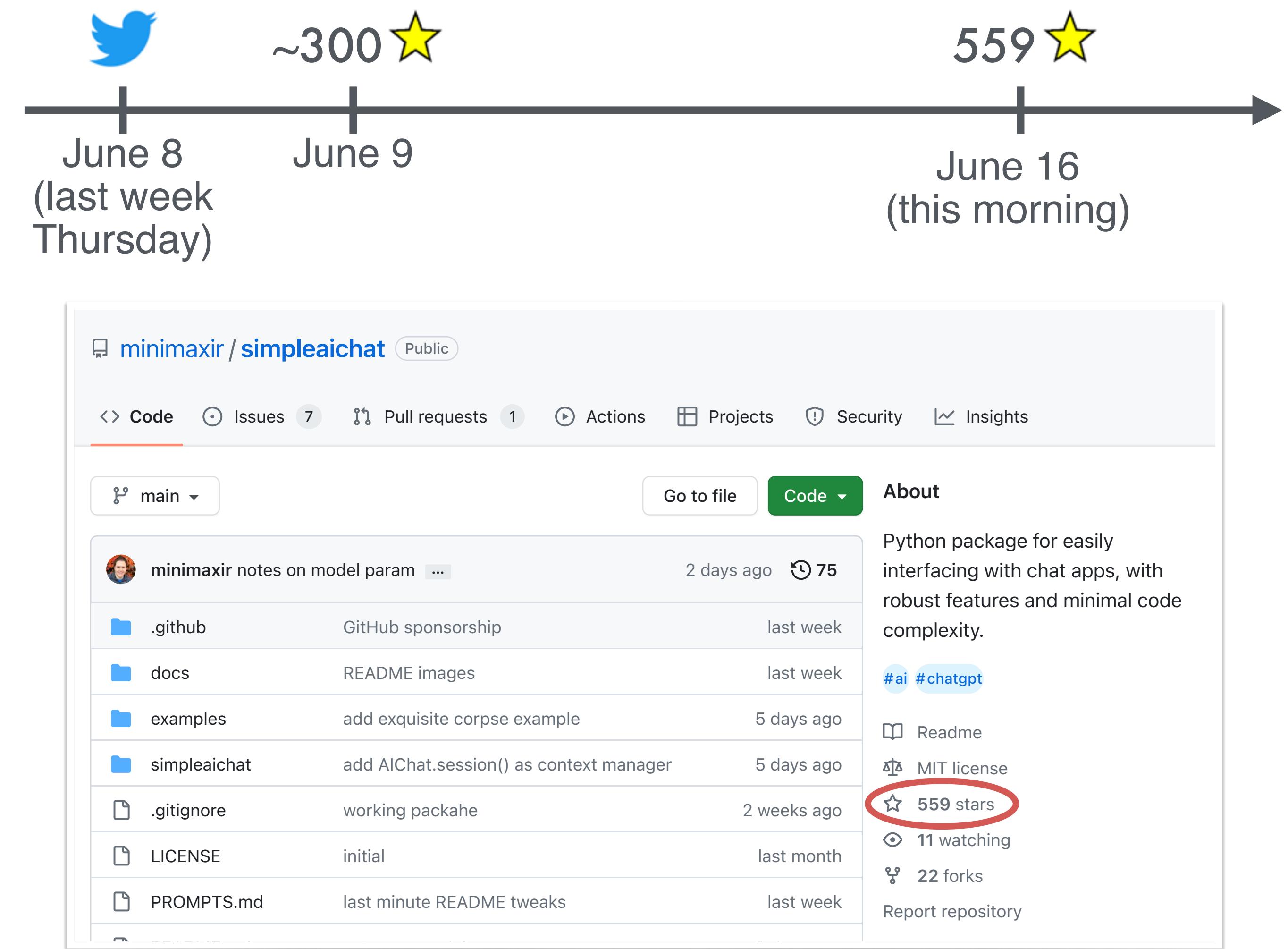
**minimaxir/simpleaichat**

Python package for easily interfacing with chat apps, with robust features and minimal code complexity.  
Contributors: 3 | Used by: 1 | Stars: 549 | Forks: 22

github.com  
GitHub - minimaxir/simpleaichat: Python package for easily interfacing with chat apps, with robust features and minimal code complexity. - GitHub - ...

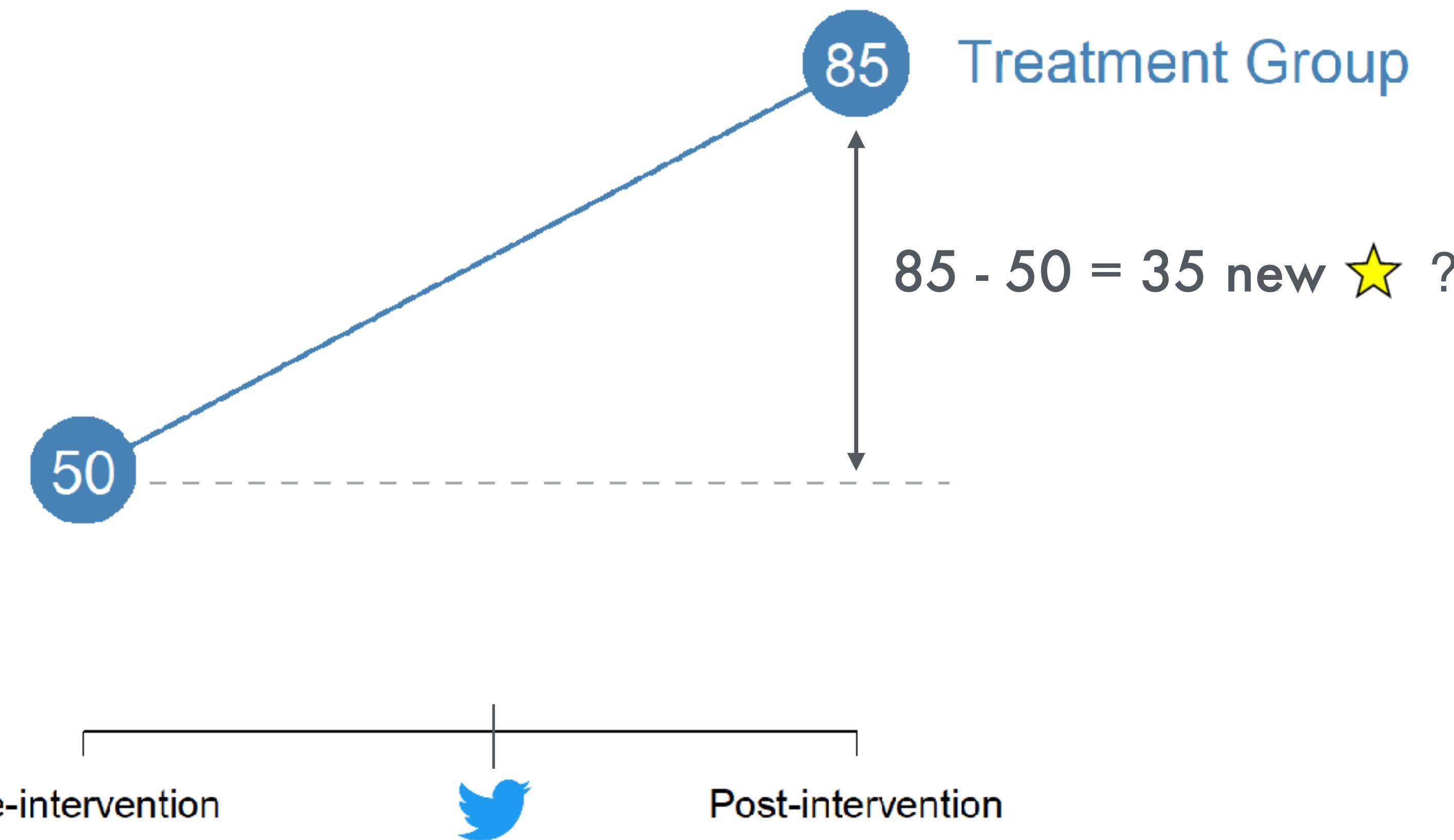
5:24 PM · Jun 8, 2023

737 likes | 1 reply | Share | Read 18 replies



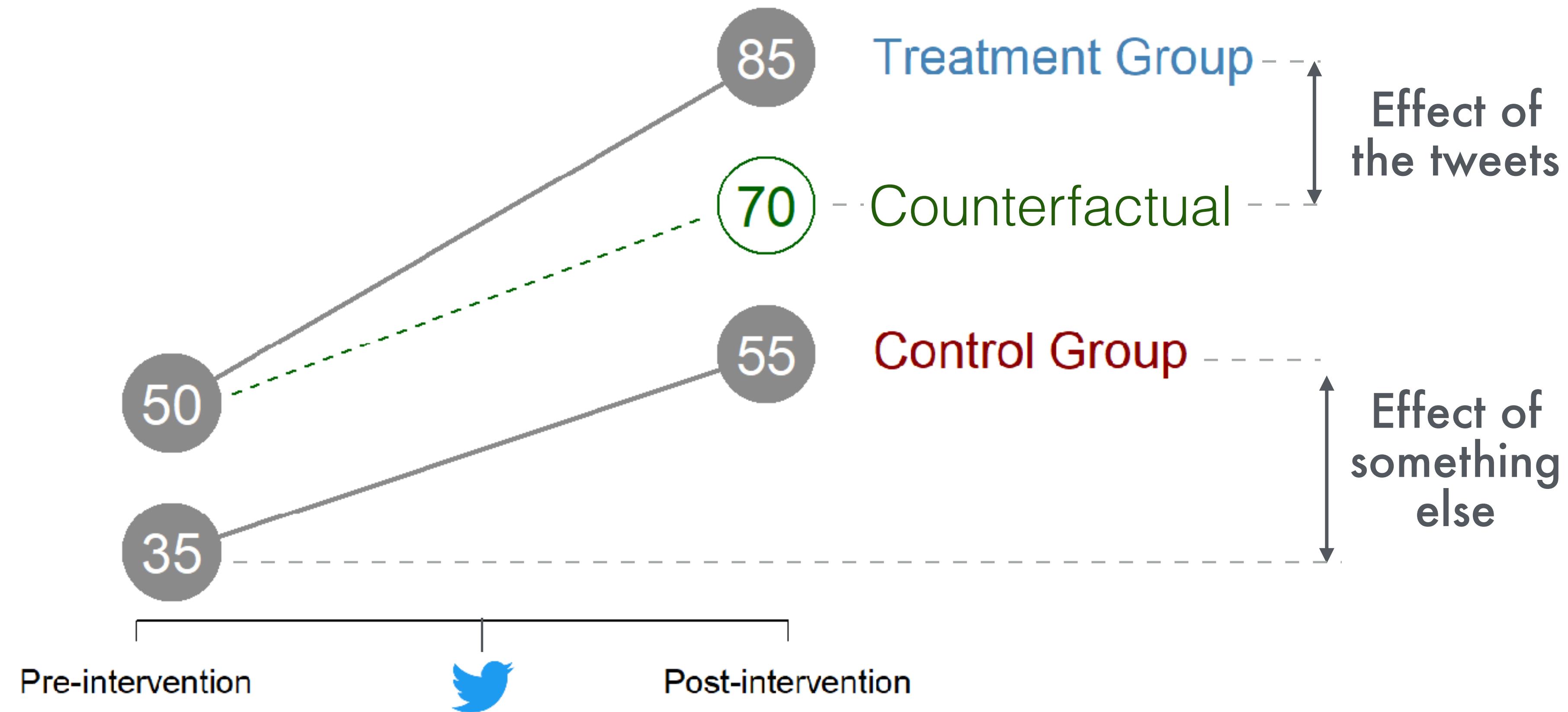
- Fang, Lamba, Herbsleb, & Vasilescu. "This is damn slick!" Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

# Idea: Measure how much a group mean changes before and after an intervention



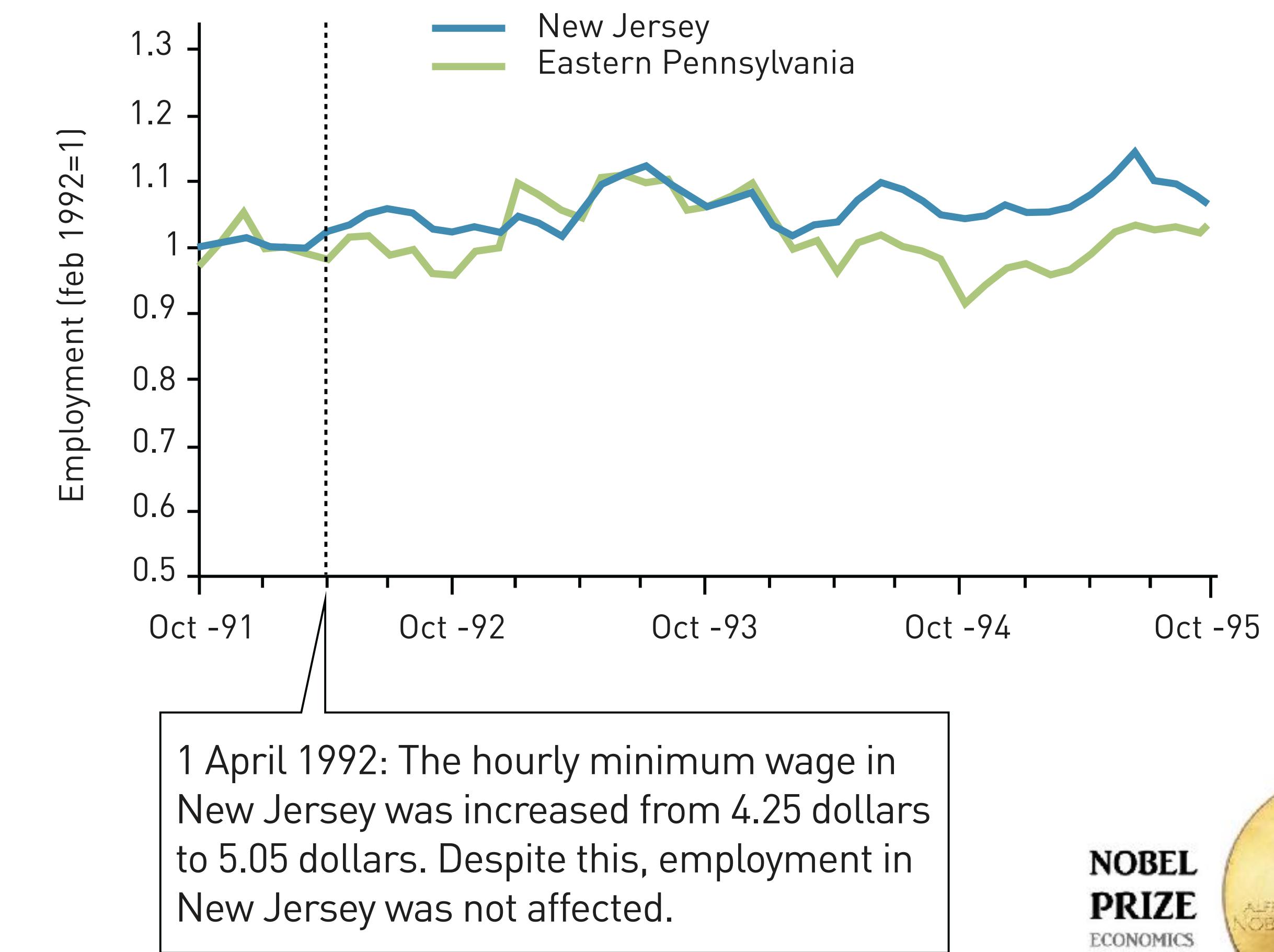
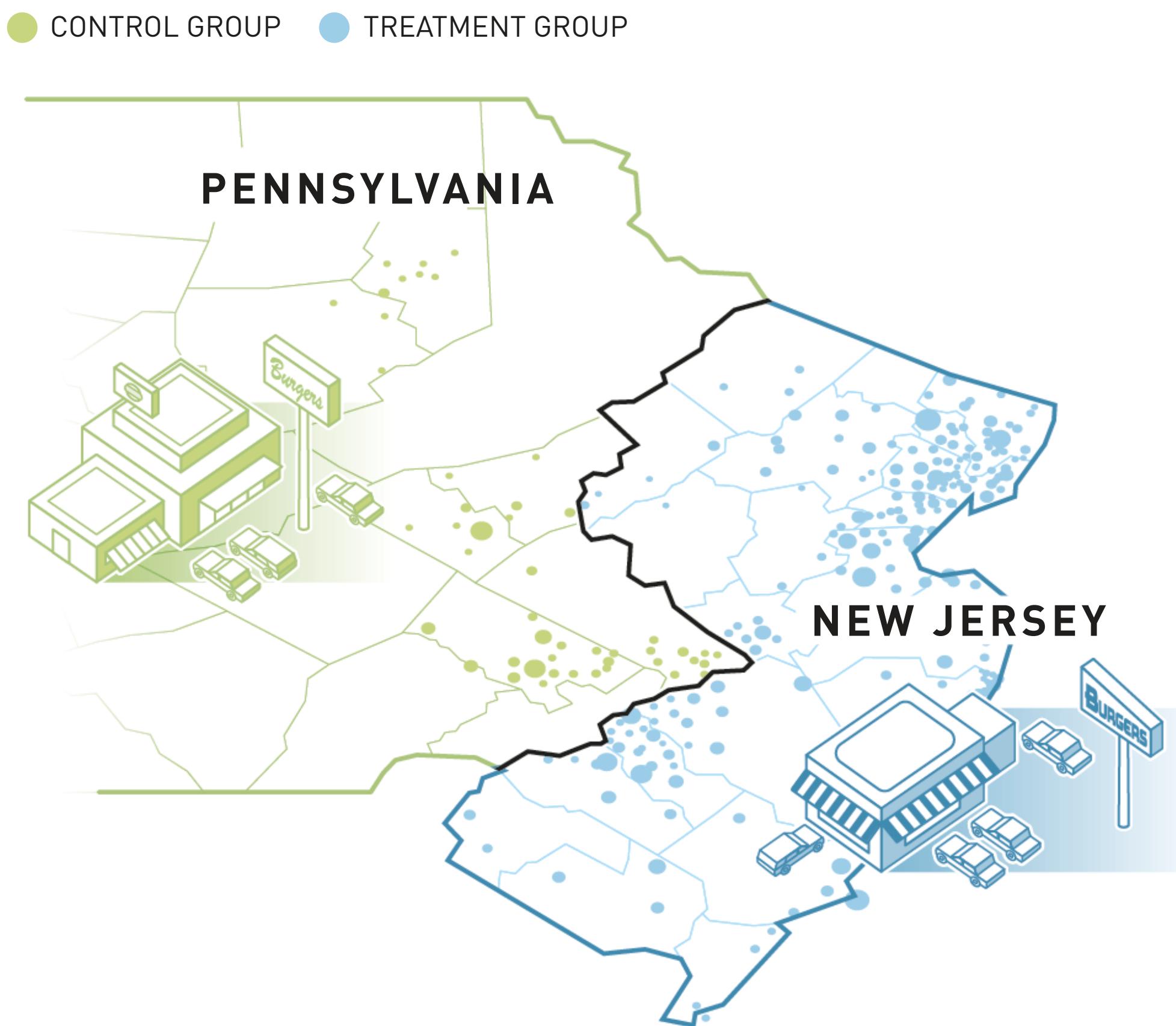
- Fang, Lamba, Herbsleb, & Vasilescu. "This is damn slick!" Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

# Better idea: Compare that change to the change in an appropriate control group



- Fang, Lamba, Herbsleb, & Vasilescu. "This is damn slick!" Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

# Card and Krueger (1993) natural experiment to study how increasing the minimum wage affects employment.



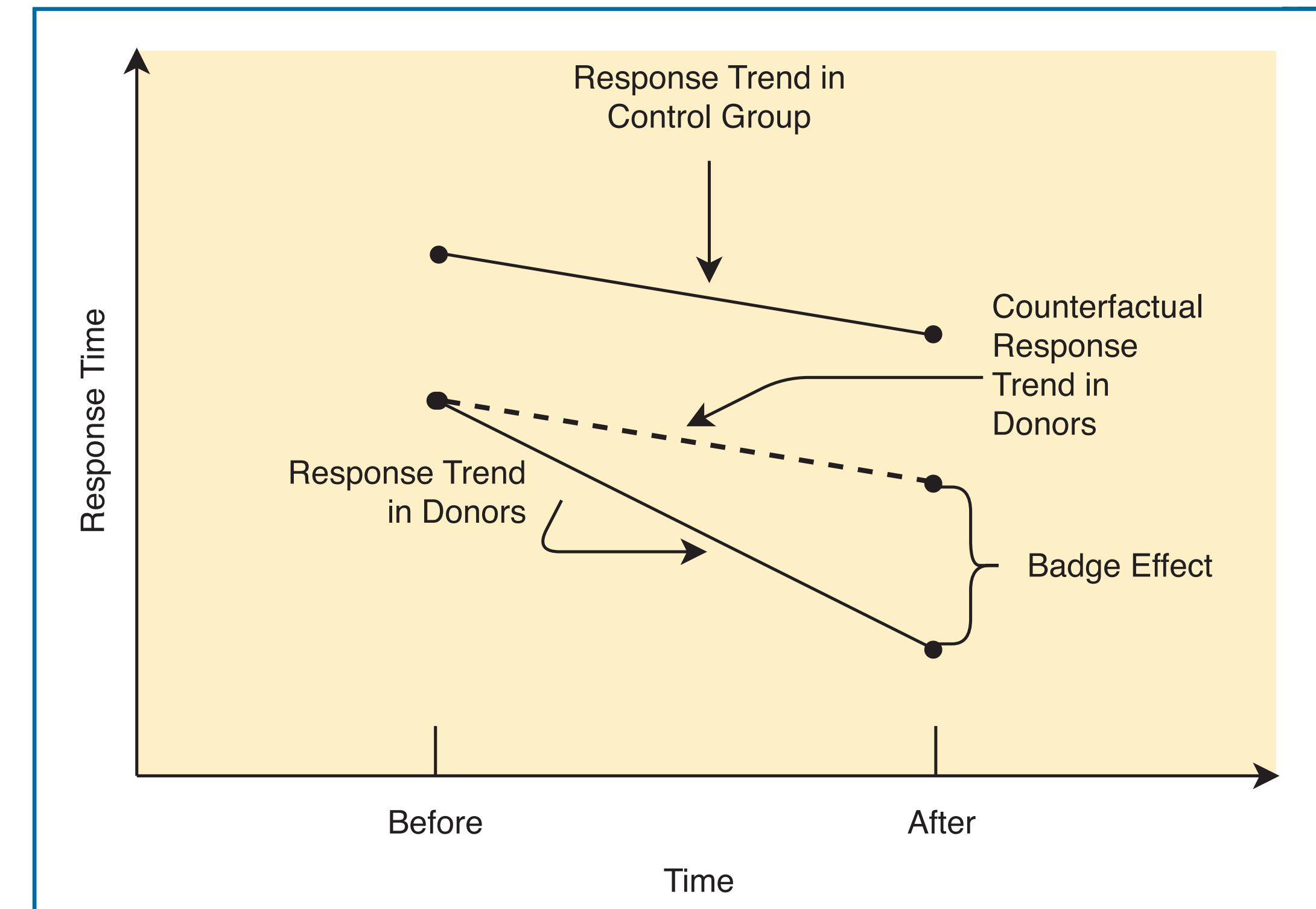
NOBEL  
PRIZE  
ECONOMICS



<https://www.nobelprize.org/uploads/2021/10/popular-economicsciencesprize2021-2.pdf>

# Another example: Donation badges decrease median bug report response times by ~2 h

Some Eclipse donors are recognized on Bugzilla with a “Friend of Eclipse” badge.

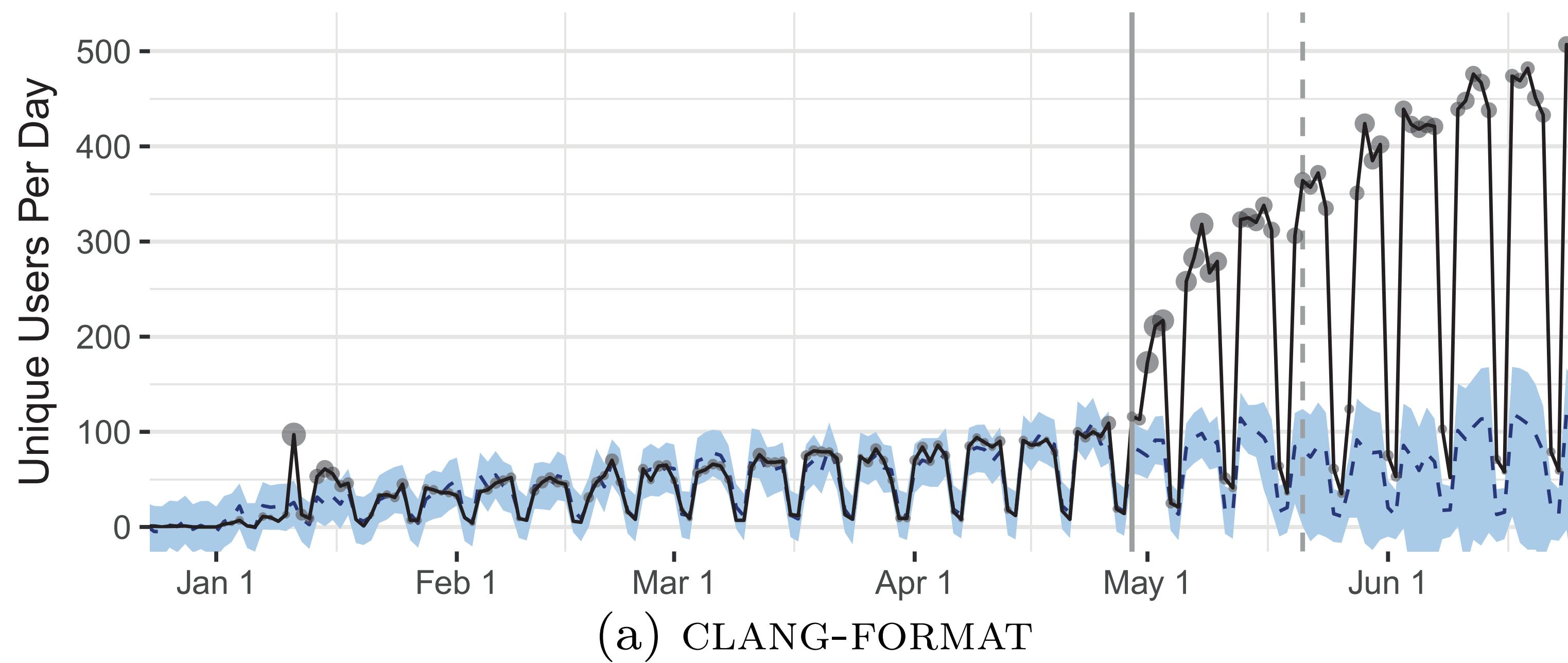


**FIGURE 1.** An example of the causal inference framework using a DID model showing response time before versus after the introduction of donation badges.

- Nakasai, Hata, & Matsumoto. Are donation badges appealing?: A case study of developer responses to Eclipse bug reports. IEEE Software 2018.

# Another example still: Advertising tools inside Google office toilets increases adoption

CausallImpact R package: Inferring causal impact using Bayesian structural time-series models



Episode 284  
April 30 2013

**Testing on the Toilet Presents... Healthy Code on the Commode**

**Automatic formatting for C++**  
by Daniel Jasper in Munich

Are you tired of hitting space and backspace more often than anything else while coding? Are you annoyed by fighting over parameter and comment alignment in code reviews?

Consistent formatting allows readers to quickly scan and interpret code, dedicating their attention to what the code does and how it works. Without this consistency, effort is wasted parsing the wide variety of personal styles code might follow. However, keeping your code formatting nice and shiny is not a good task for humans. Luckily, we now have clang-format, which can do this tedious task for you.

Clang-format produces both readable and Google style-compliant code:

```
$ cat file.cc
int a; // clang-format can ..
int bbb; // .. align trailing comments.
#define UNDERSTAND_MULTILINE_MACROS int cc; int d;
LOG(INFO) << "... align operators\n" << "... and many more things";
$ clang-format file.cc -style Google
int a; // clang-format can ..
int bbb; // .. align trailing comments.
#define UNDERSTAND_MULTILINE_MACROS
 int cc;
 int d;
LOG(INFO) << "... align operators\n"
 << "... and many more things";
```

Conveniently integrating with your editor, you can format the current statement or a selected region (available for vim, emacs and eclipse - [go/clang-format](#)). You can also reformat unified diffs, e.g. in a CitiC client, by:

```
$ g4 diff -du0 | /usr/lib/clang-format/clang-format-diff.py
```

In addition to making the editor-based code development faster and more fun, consistently using clang-format provides other advantages:

- Code reviewers don't even need to consider whether all your spaces are correct
- Source files become fully machine editable, e.g. for API maintenance

So, give it a try and see how much fun it is to just type everything into a single line and let clang-format do the rest. If you encounter clang-format messing up the formatting, e.g. producing style guide violations, please file a bug on [go/clang-format-bug](#).

**clang-format**  
Learn how to use clang-format in your workflow.  
<http://go/clang-format>  
Find out more: [go/CodeHealth](#)

**Scythe**  
Want to see your dead code and automatically get rid of it?  
<http://go/scythe>  
Read all TotTs online: <http://tott>

- Murphy-Hill, Smith, Sadowski, et al. Do developers discover new tools on the toilet?. ICSE 2019

# Where to start

r-causal.org

The screenshot shows the homepage of the r-causal.org website. The main title is "Causal Inference in R". Below it, there's a search bar and a sidebar with navigation links like "Preface", "The Design Phase", and "Estimating Causal Effects". The main content area has a section titled "Preface" with a list of 10 steps from causal to causal. It also includes a "PUBLISHED" section with authors' names and a date.

**Causal Inference in R**

AUTHORS  
Malcolm Barrett  
Lucy D'Agostino McGowan  
Travis Gerke

PUBLISHED  
April 11, 2025

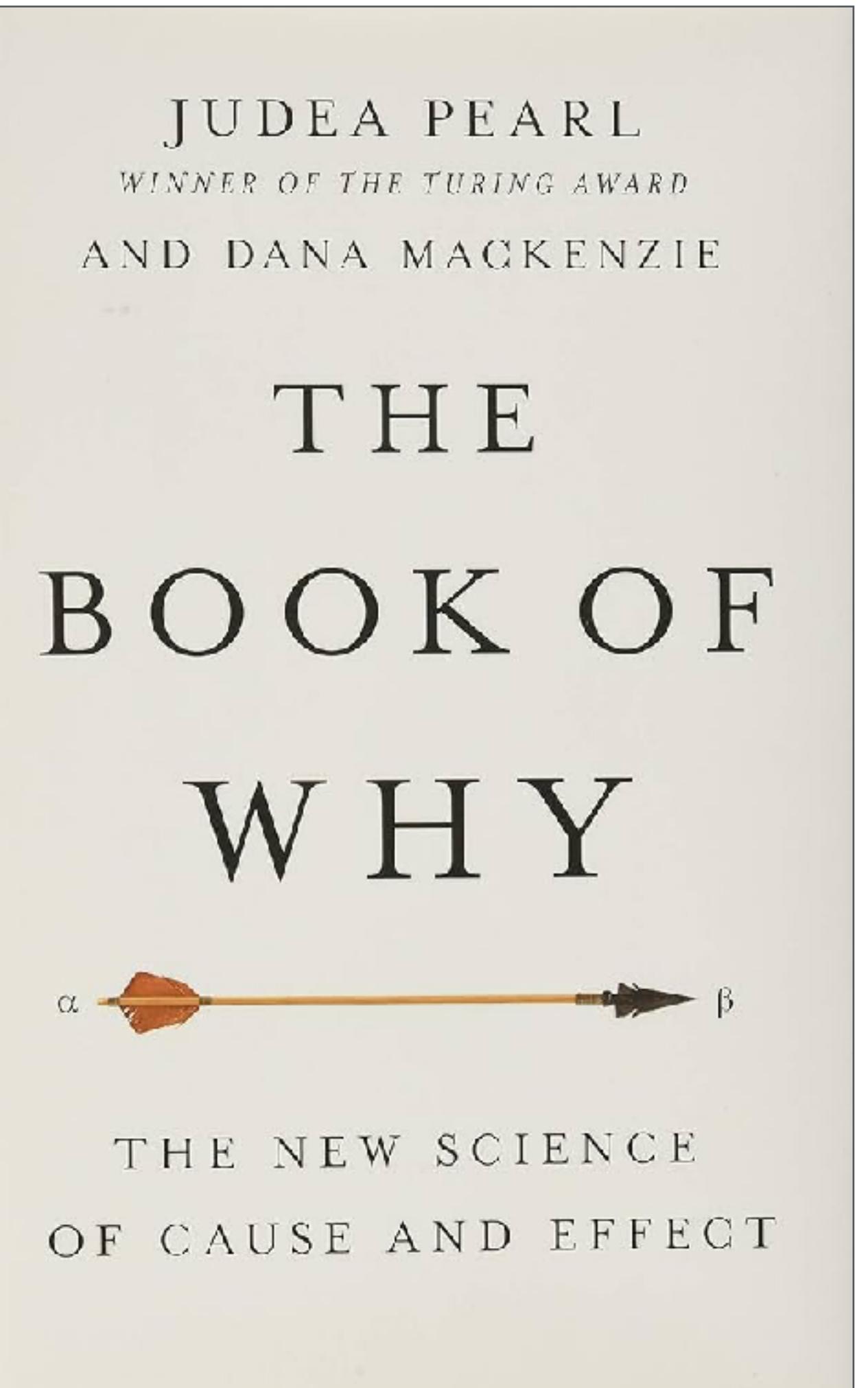
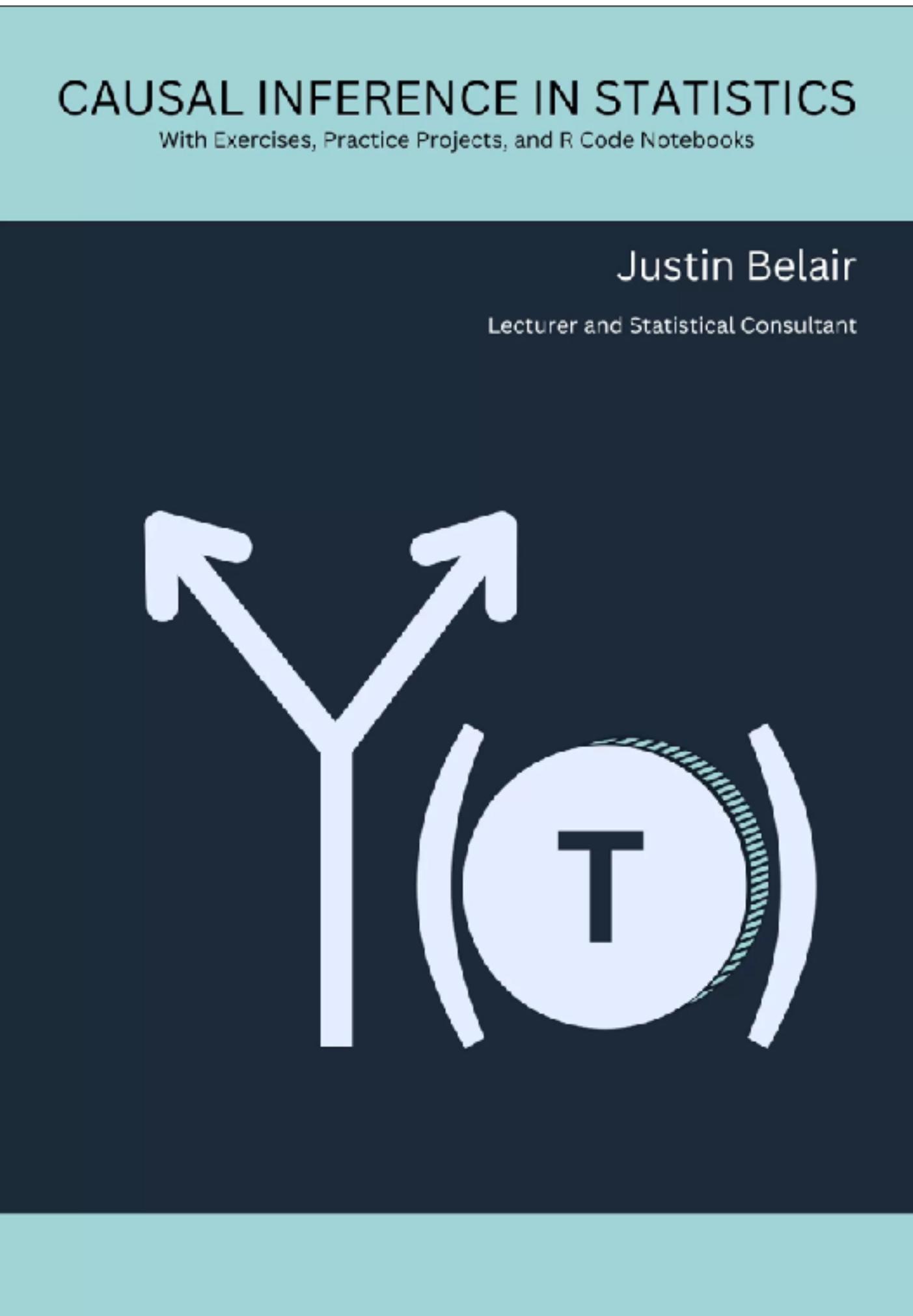
**Preface**

1 From causal to causal  
2 The whole game: mosquito nets and malaria  
3 Potential outcomes and counterfactuals  
4 Expressing causal questions as DAGs  
5 Causal inference is not (just) a statistical problem  
6 From question to answer: stratification and outcome models  
7 Preparing data to answer causal questions  
8 Propensity scores  
9 Evaluating your propensity score model  
10 Causal estimands

**The Design Phase**

1. Ask better causal questions.  
2. Understand the assumptions needed for causal inference  
3. Identify the target population for which you want to make inferences  
4. Fit causal models and check their problems  
5. Conduct sensitivity analyses where the techniques we use might be imperfect

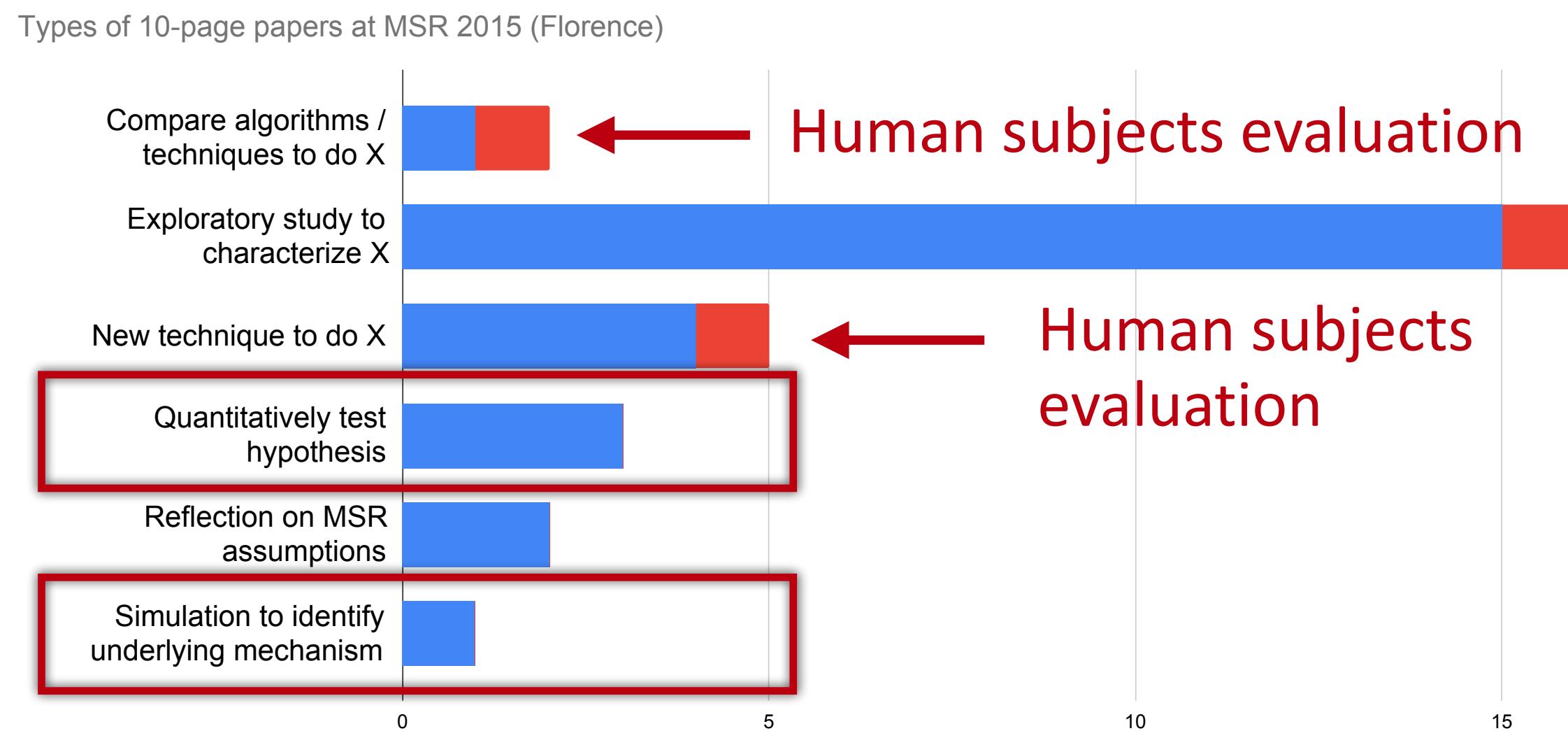
This book is for both academic researchers and data scientists. Although the questions may differ between these settings, many techniques are the same: causal inference is as helpful for asking questions about cancer as it is about clicks. We use a mix of examples from medicine, economics, tech, and other domains to demonstrate that you need a clear causal question and a willingness to be transparent about your assumptions.



# Summary: We are a methods conference, let's step up our methods game!

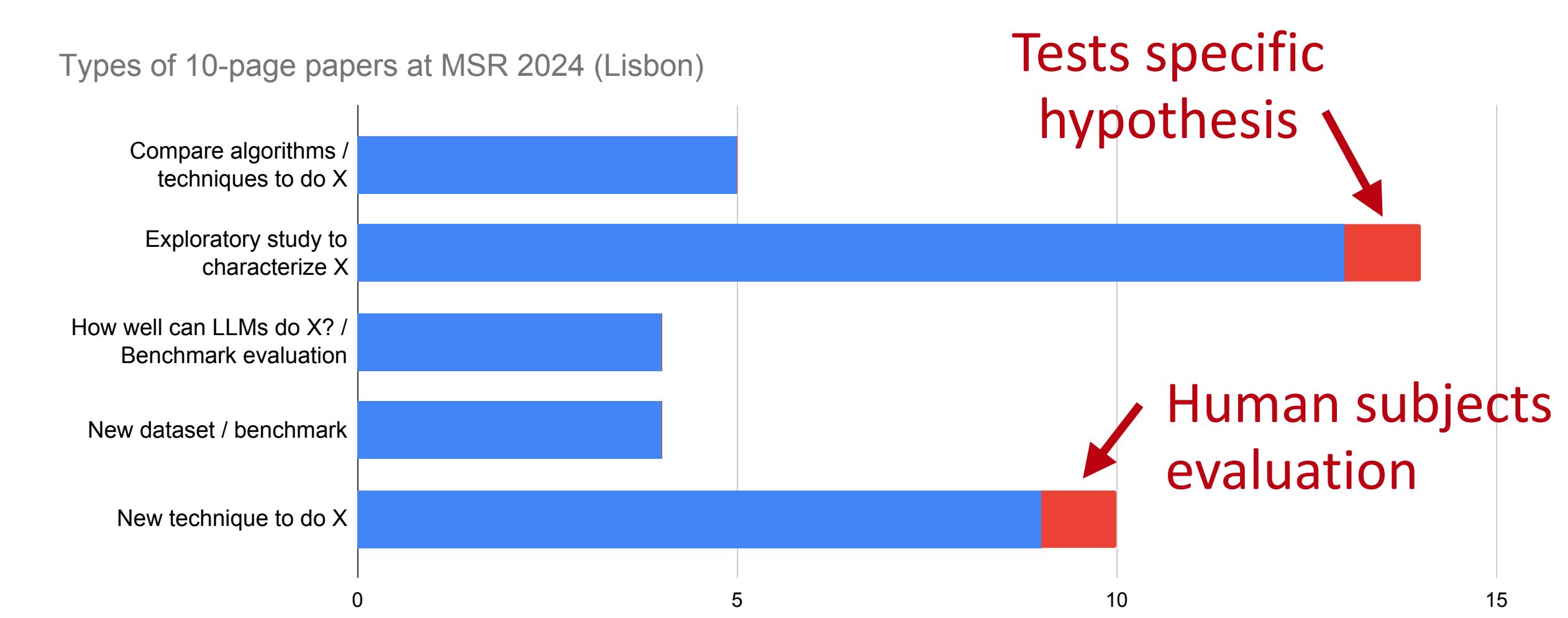
## Understanding the problem:

- Less descriptives, more understanding mechanisms and testing hypotheses
- Causal relationships are good theory fragments, and allow for predictions



## Designing solutions:

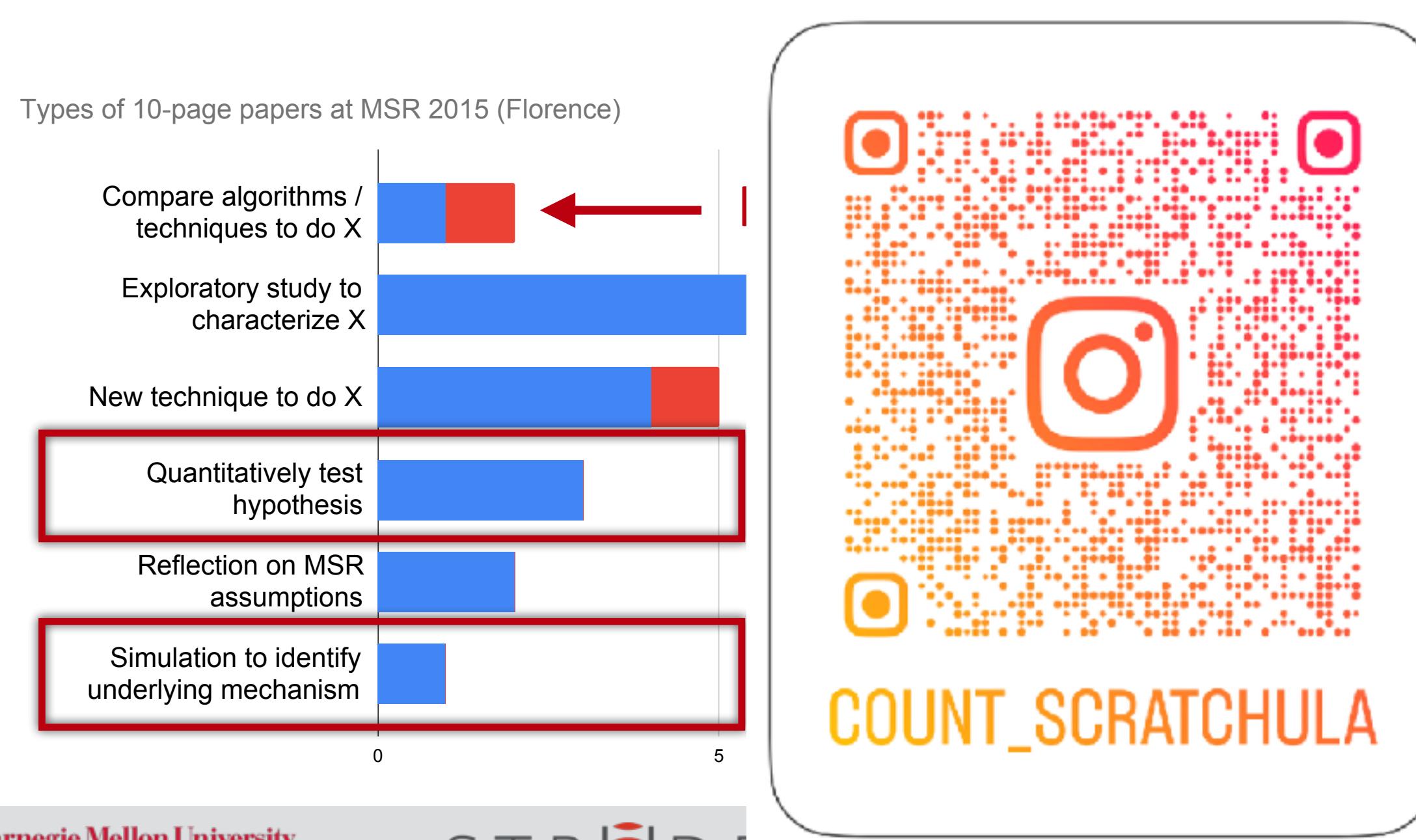
- Less benchmark evaluations, more human-centered methods
- More theory: why, how, when, for whom, and under what conditions does it work?



# Summary: We are a methods conference, let's step up our methods game!

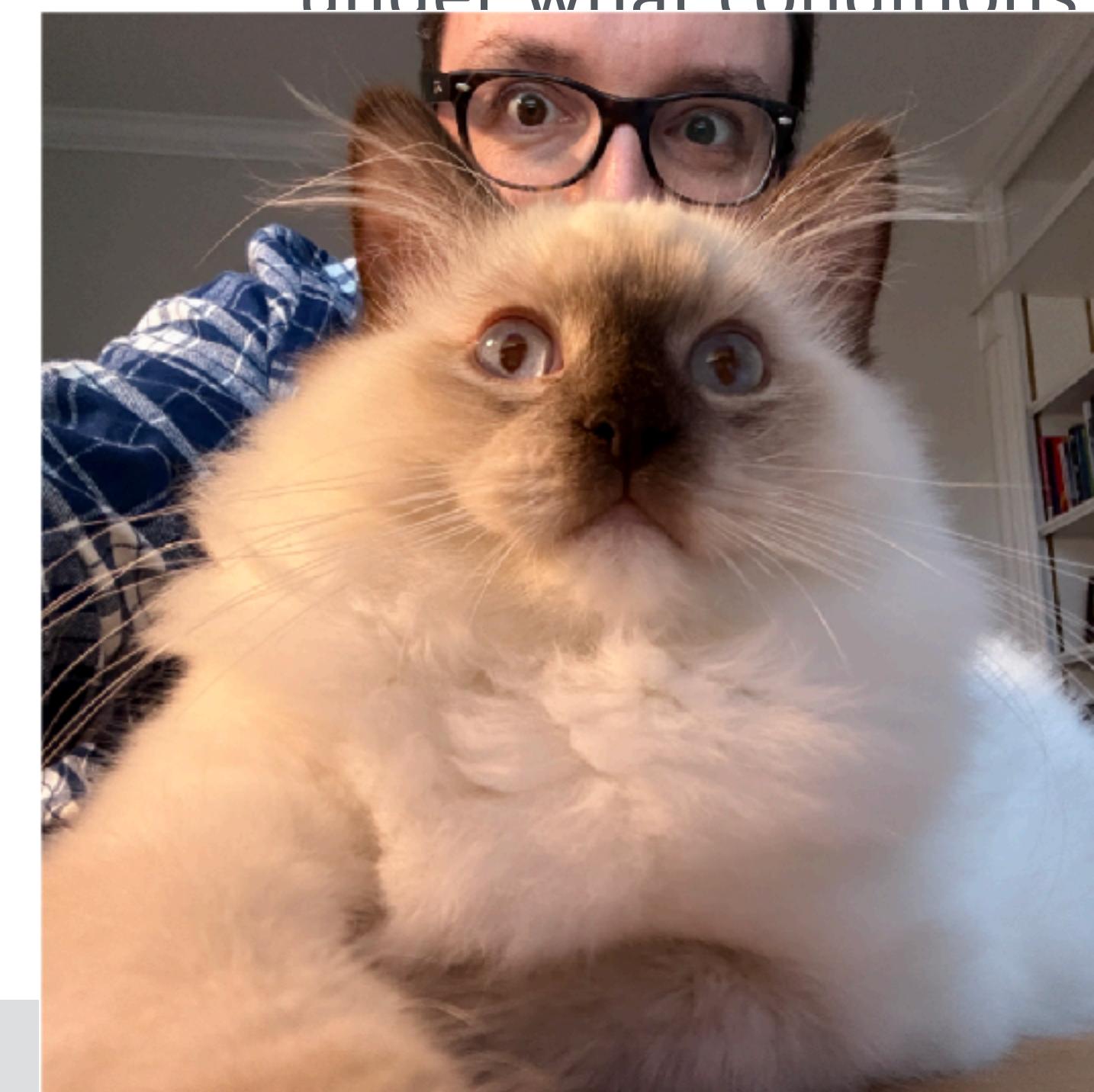
## Understanding the problem:

- Less descriptives, more understanding mechanisms and testing hypotheses
- Causal relationships are good theory fragments, and allow for predictions



## Designing solutions:

- Less benchmark evaluations, more human-centered methods
- More theory: why, how, when, for whom, and under what conditions does it work?



Tests specific hypothesis

Human subjects evaluation