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

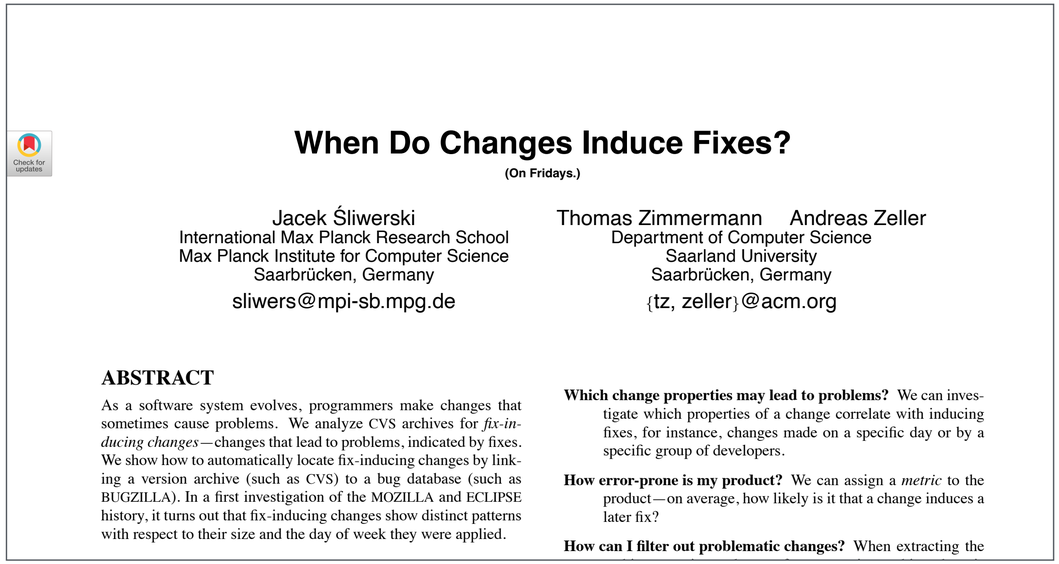
About me

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

COUNT_SCRATCHULA

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



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

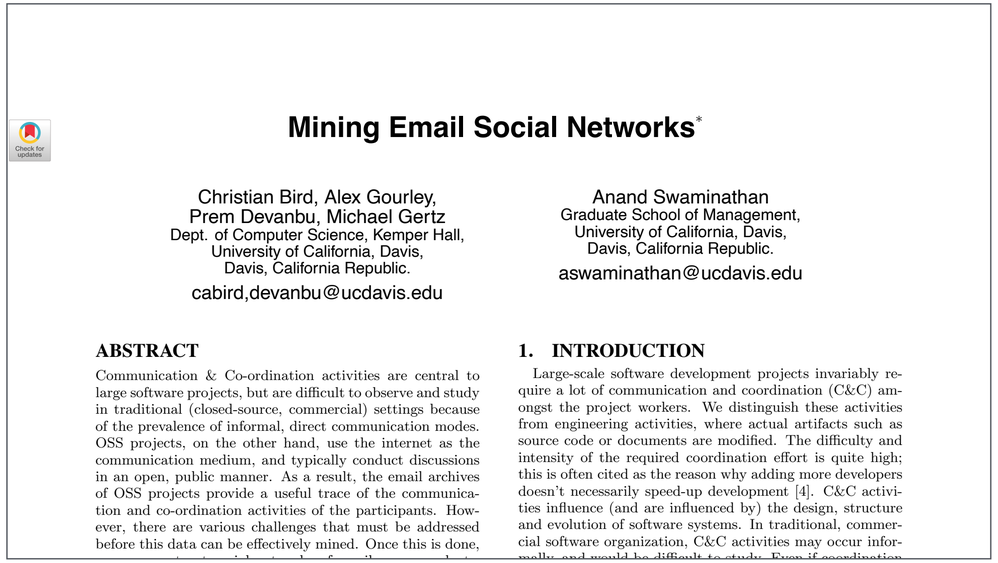
ABSTRACT

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



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 infor-

About me

- Raj Reddy Associ ms
- Societal Comput
- Socio-Technical R
- Reading since M l 2012


When Do Networks*

Jacek Śliwerski
International Max Planck Resea
Max Planck Institute for Comput
Saarbrücken, German
sliwers@mpi-sb.mpg

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

ABSTRACT

As a software system evolves, programmers mak
sometimes cause problems. We analyze CVS arch
ducing changes—changes that lead to problems, ind
We show how to automatically locate fix-inducing c
ing a version archive (such as CVS) to a bug dat
BUGZILLA). In a first investigation of the MOZILL
history, it turns out that fix-inducing changes show c
with respect to their size and the day of week they w

TRODUCTION

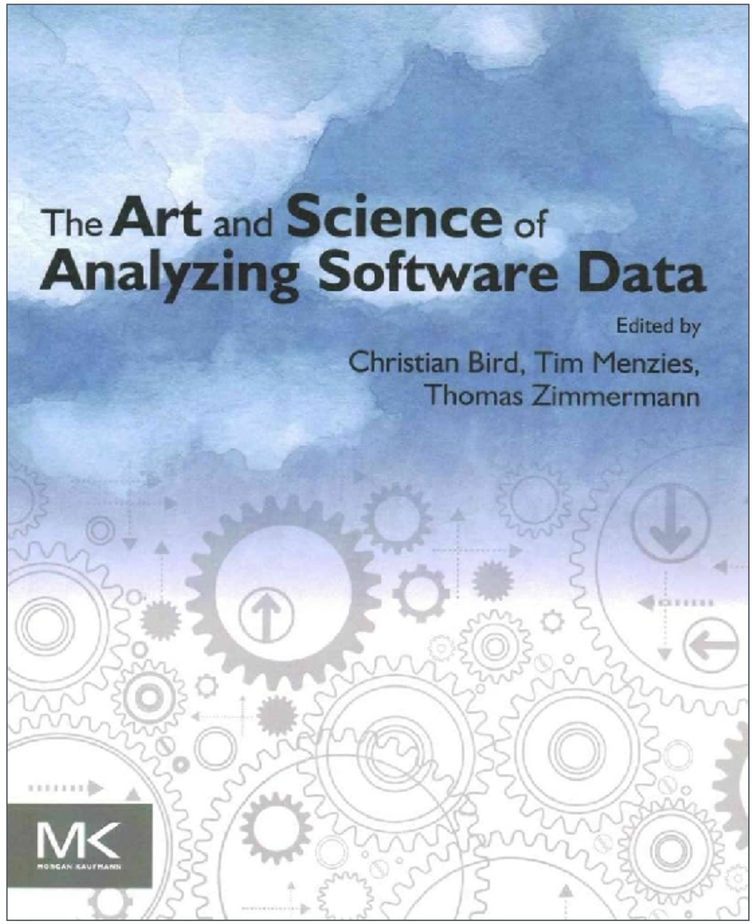
scale software development projects invariably re-
ot of communication and coordination (C&C) am-
project workers. We distinguish these activities
ineering activities, where actual artifacts such as
ode or documents are modified. The difficulty and
of the required coordination effort is quite high;
en cited as the reason why adding more developers
necessarily speed-up development [4]. C&C activi-
ence (and are influenced by) the design, structure
ution of software systems. In traditional, commer-
vare organization, C&C activities may occur infor-

There is no such thing as MSR!

MSR = Data Science + Software Engineering

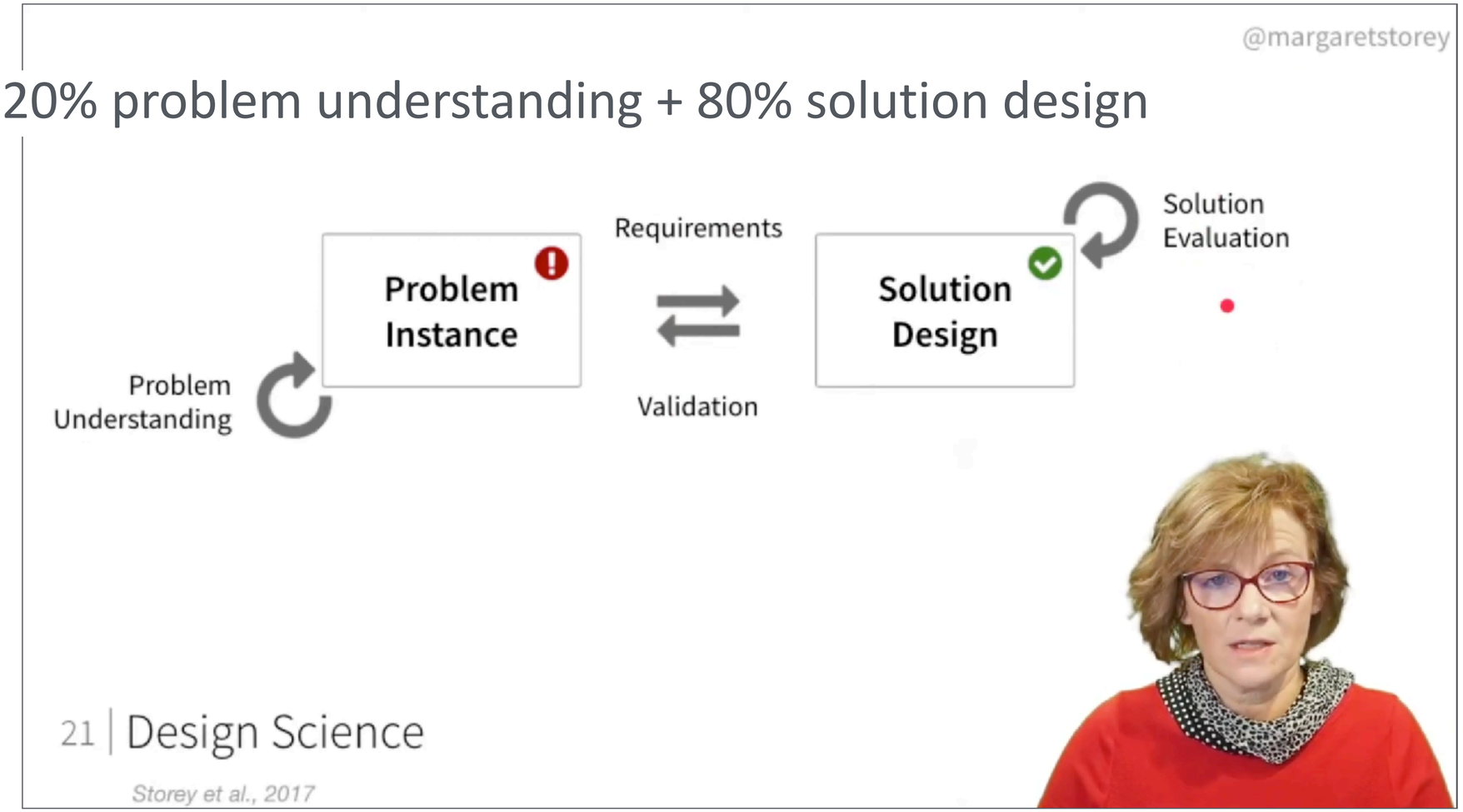
Table 1.3 Mapping Information Needs (Left) to Automatic Technique (Right)

Information Need	Description	Insight	Relevant Techniques
Summarization	Search for important or unusual factors to 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



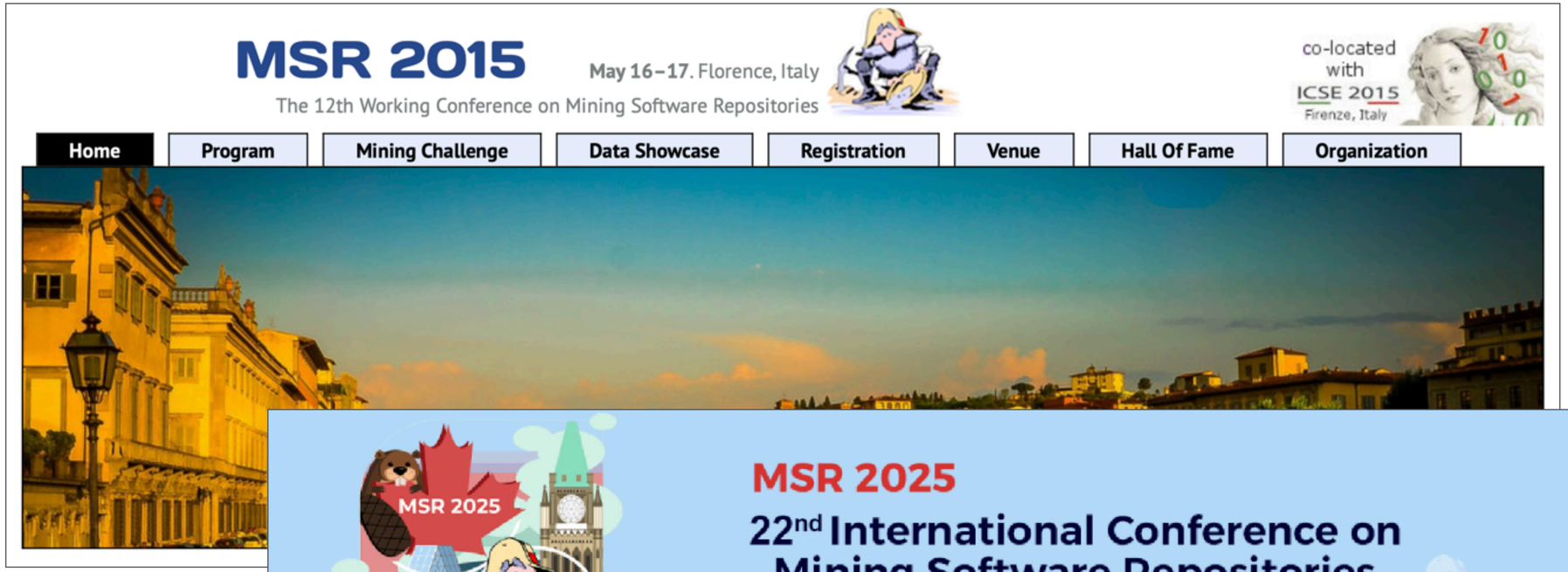
- Storey, Ernst, et al. “The who, what, how of software engineering research: a socio-technical framework.” EMSE 2020. Talk: https://youtu.be/fs2XhM5-zXI

Where is MSR now?

How have we changed in the last 10 years?

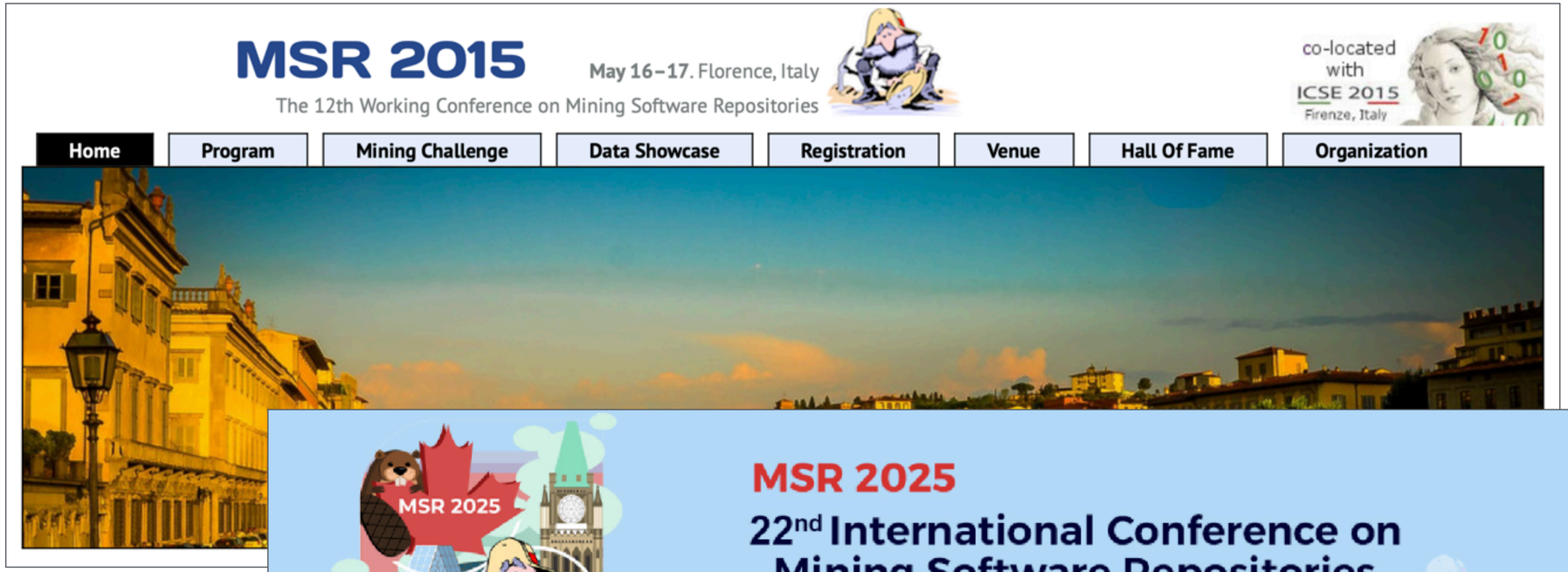
How to increase the impact of our work?

MSR 2015 (Florence) vs MSR 2025 (Ottawa)

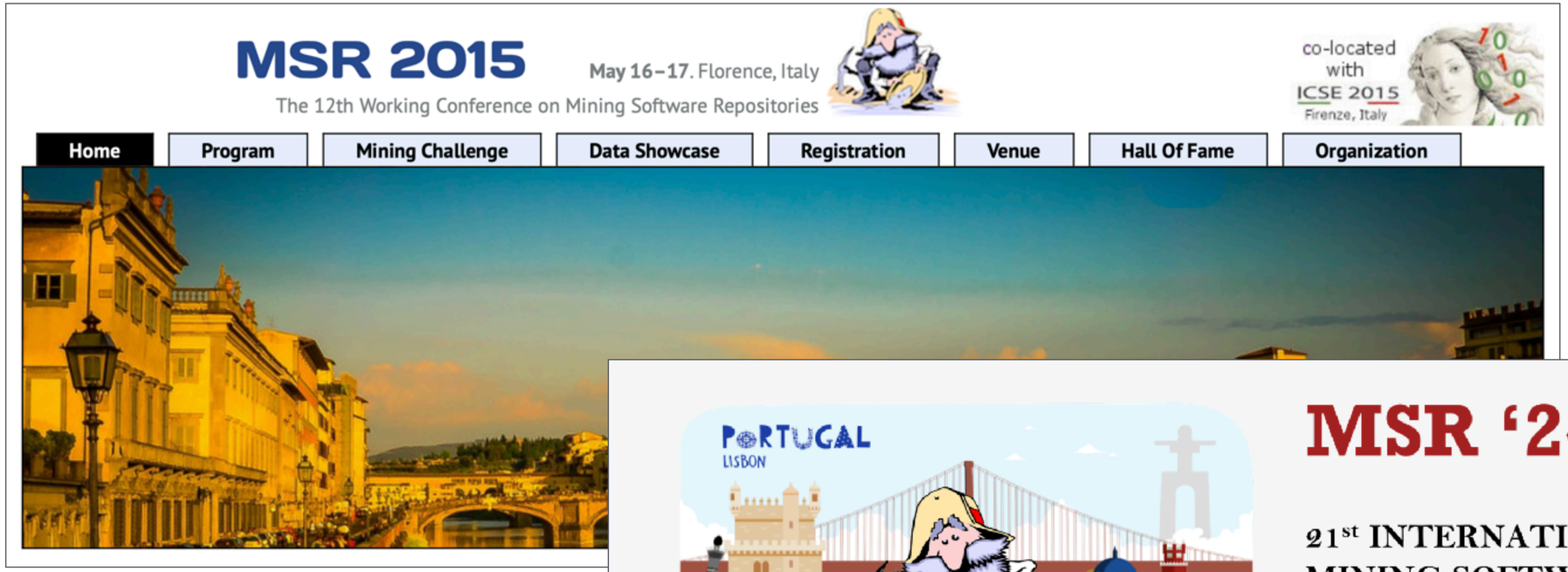


MSR 2015 (Florence) vs ~~MSR 2025 (Ottawa)~~

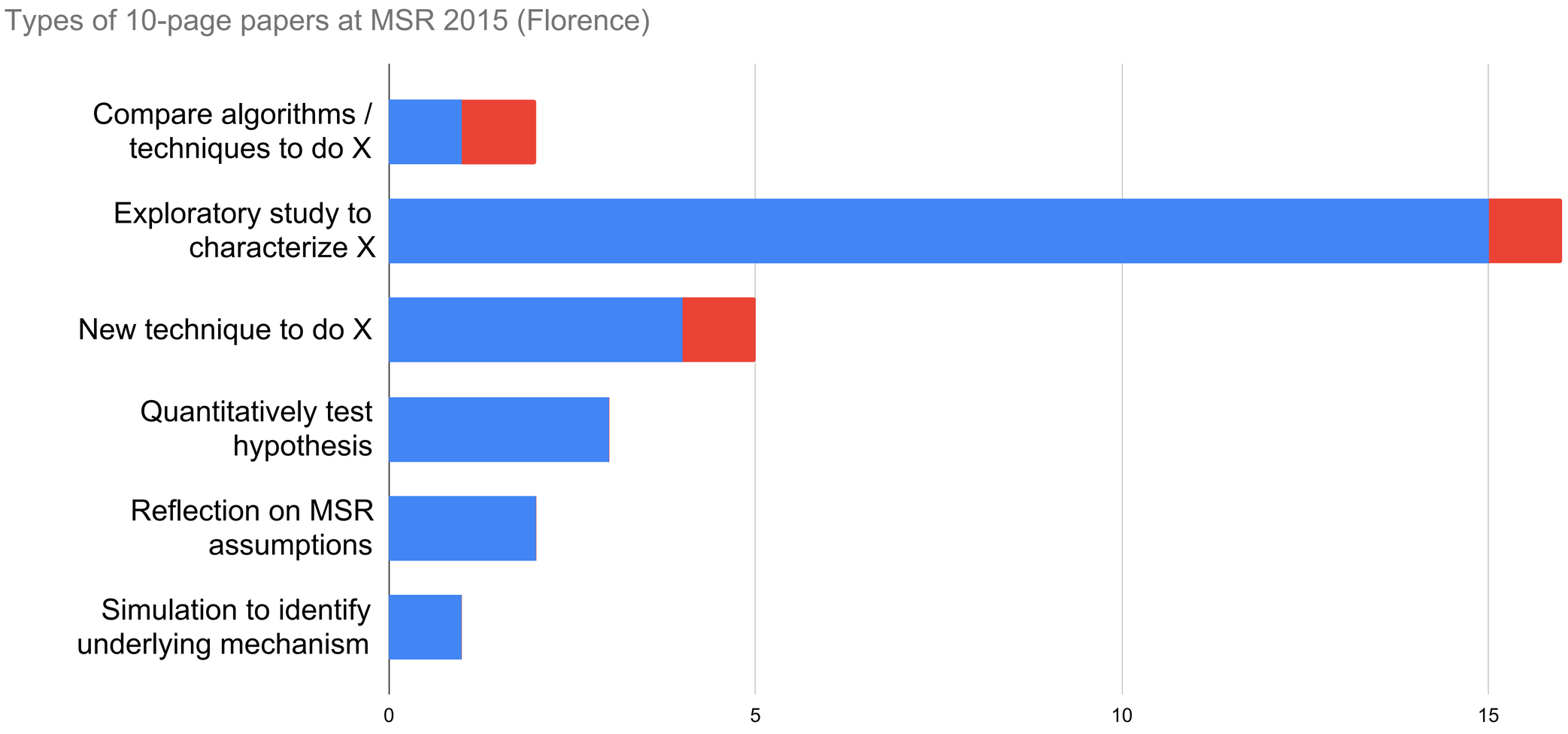
No proceedings, no preprints ¯_(ツ)_/¯



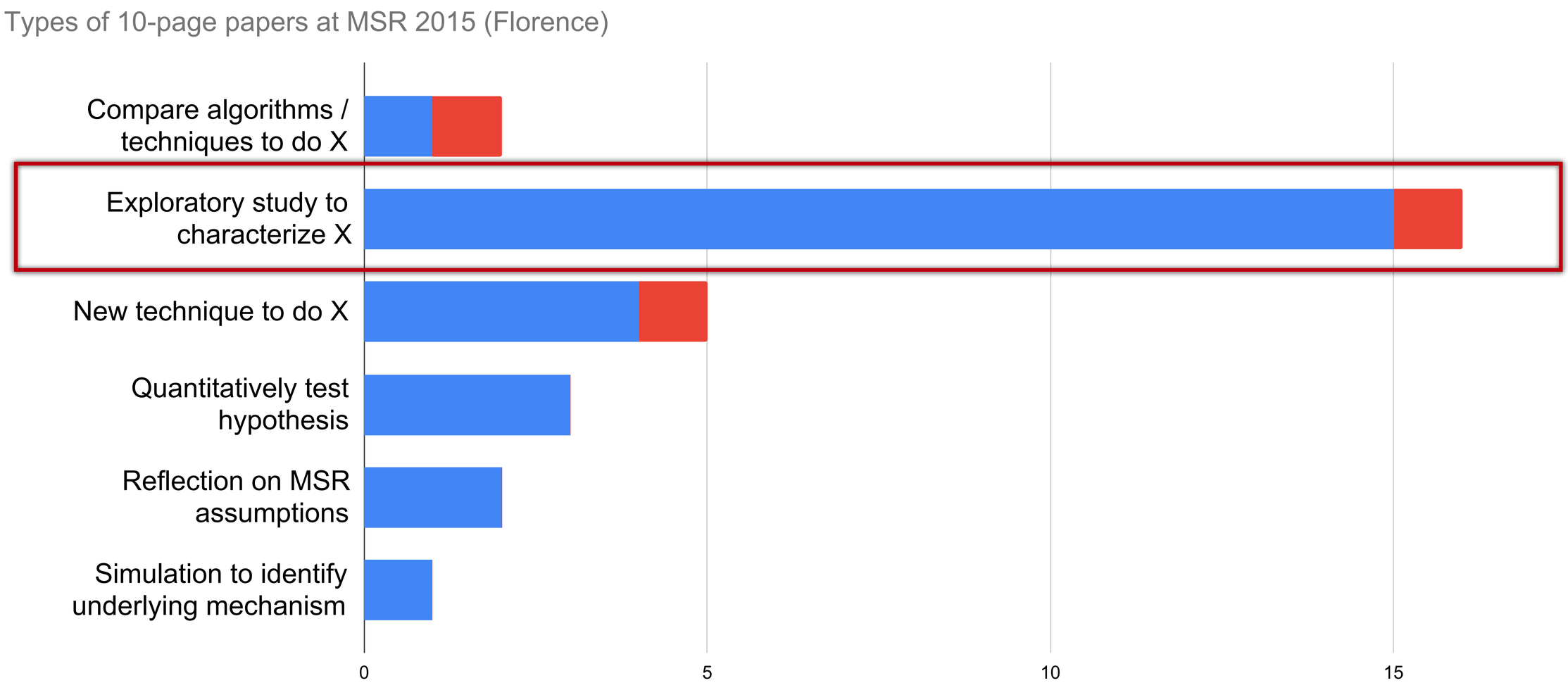
MSR 2015 (Florence) vs MSR 2024 (Lisbon)



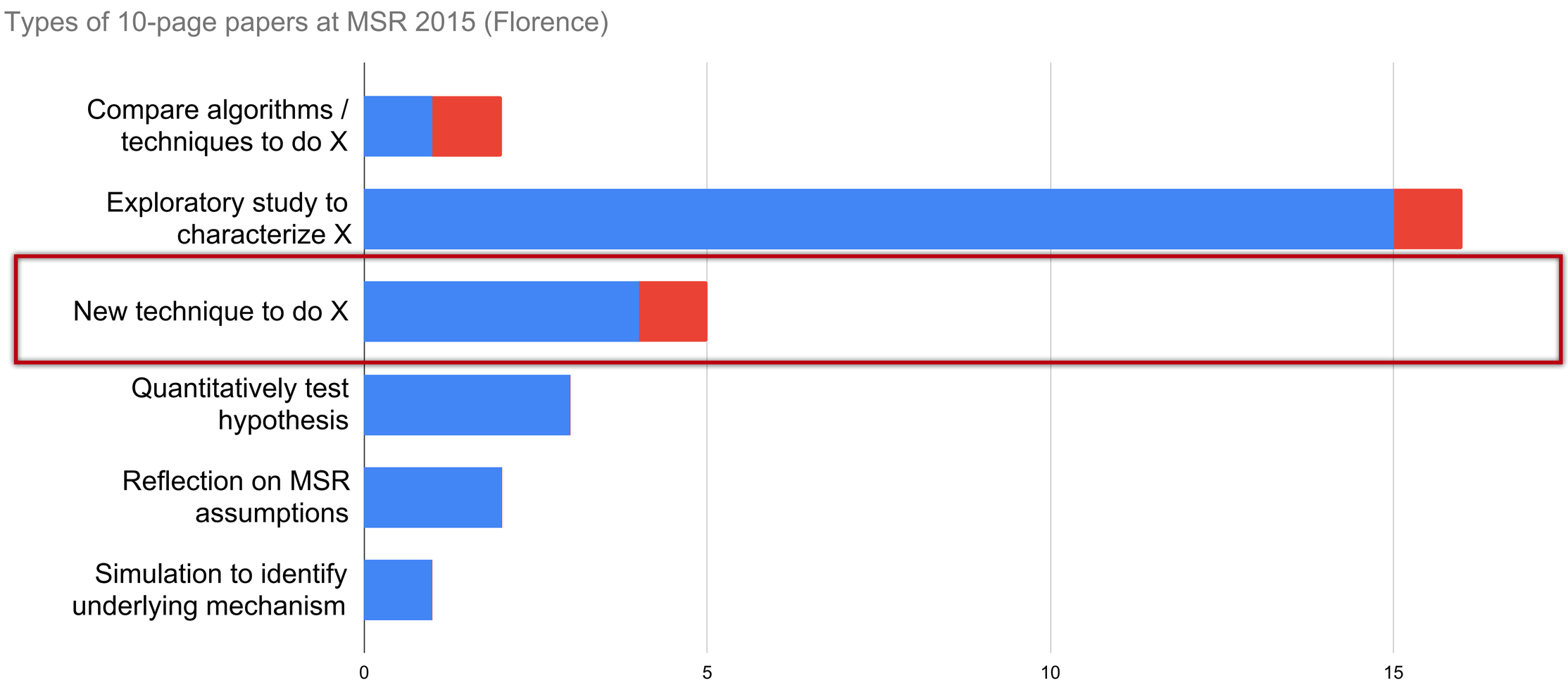
MSR 2015 (Florence): 29 full papers



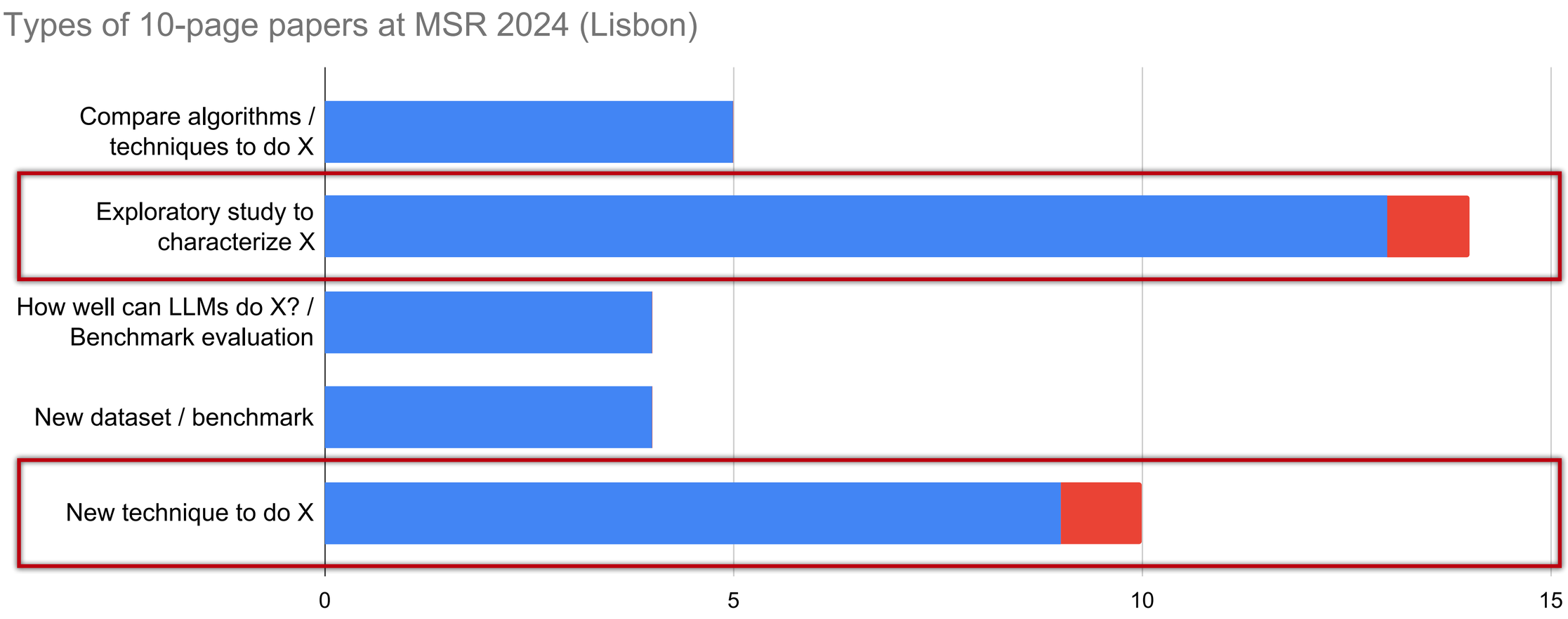
MSR 2015 (Florence): 29 full papers



MSR 2015 (Florence): 29 full papers

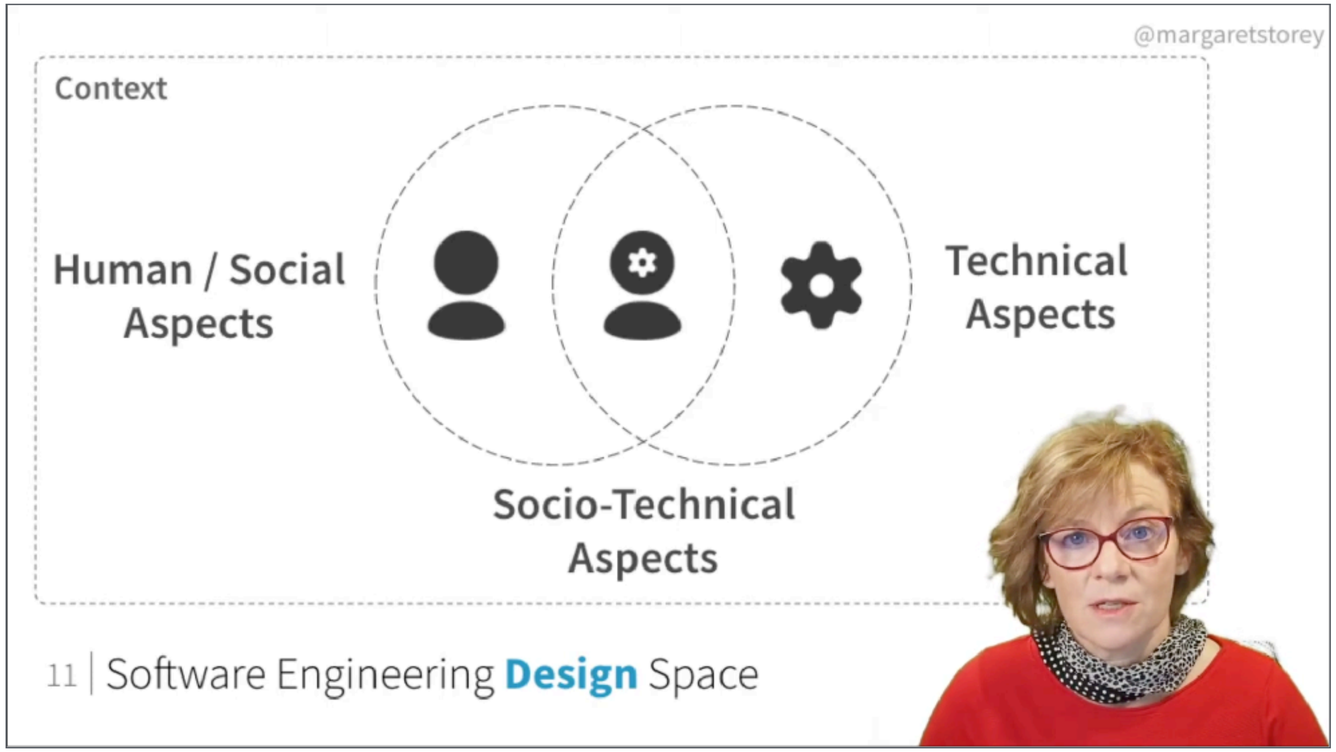


MSR 2024 (Lisbon): 37 full papers



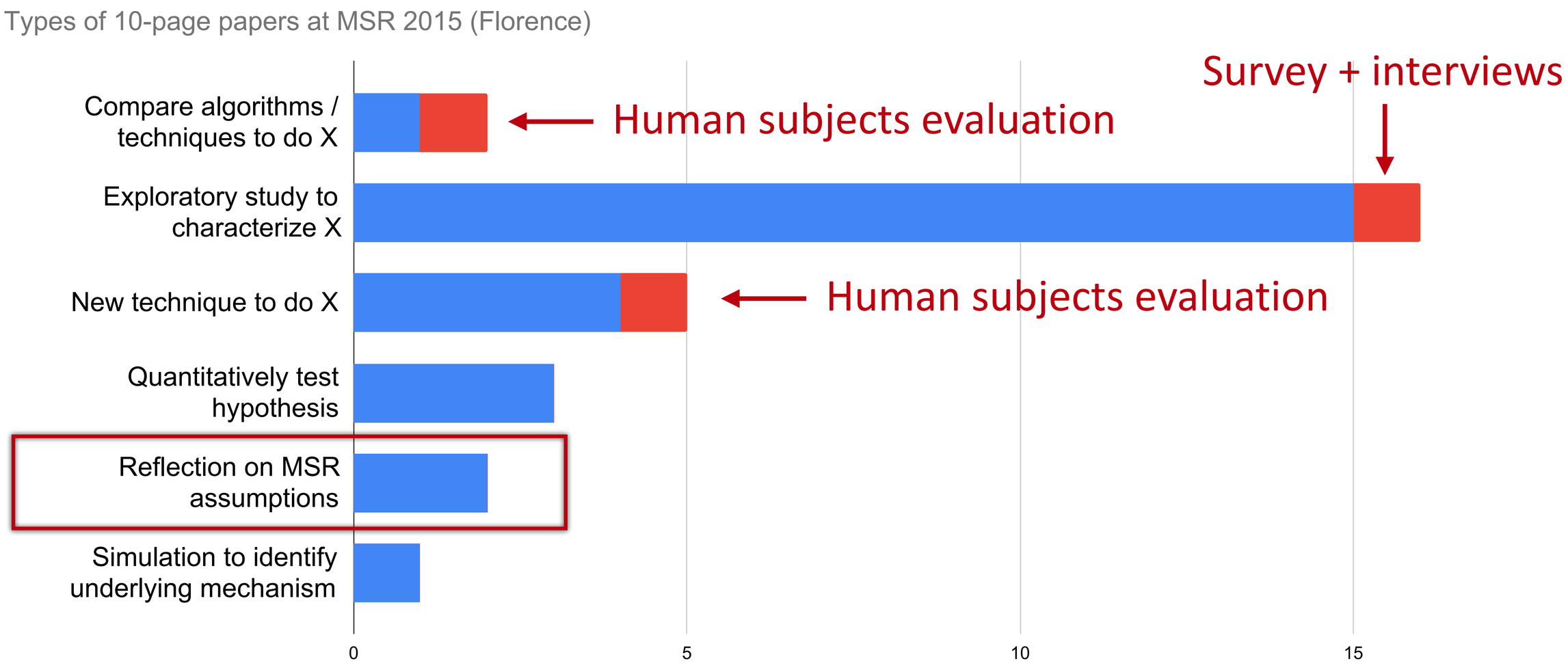
Lots written about high-quality, impactful SE research already

- 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

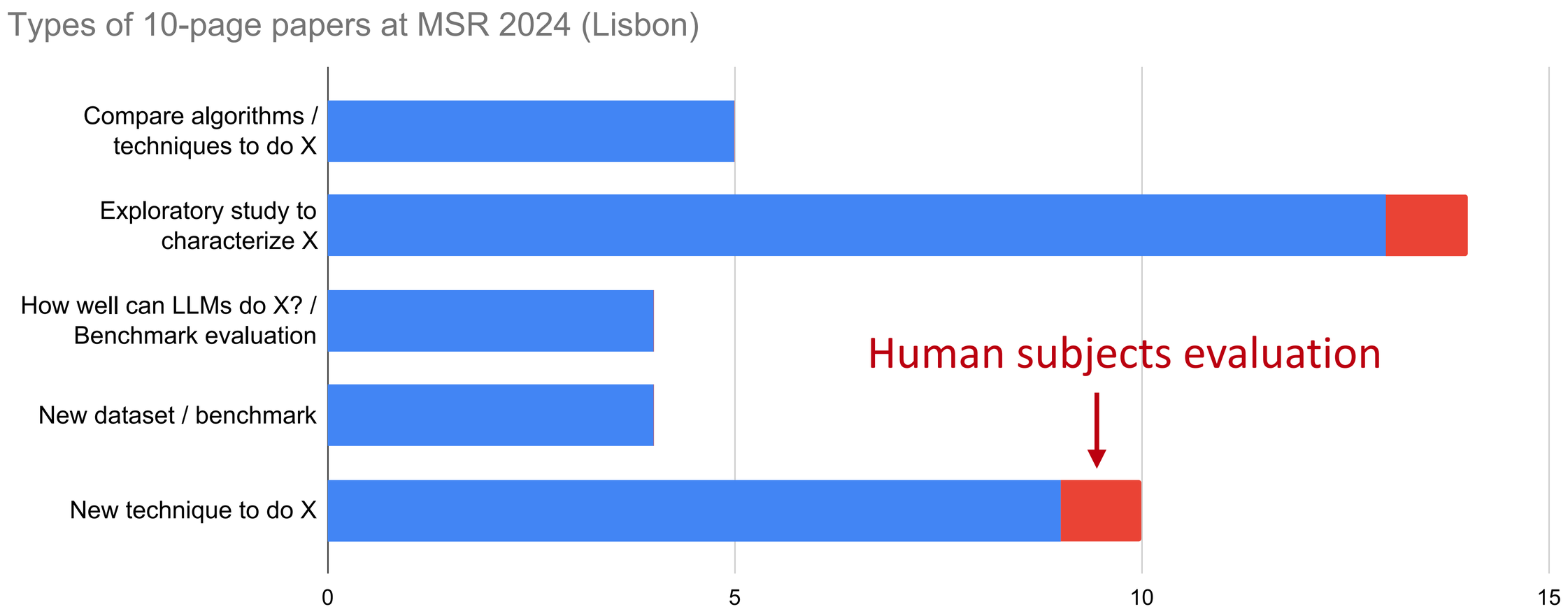


https://youtu.be/fs2XhM5-zXI

MSR 2015 (Florence): 29 full papers



MSR 2024 (Lisbon): 37 full papers



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

 1

- Easterbrook. FSE 2006 Doctoral Symposium

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/scico

Theory-oriented software engineering ☆

Klaas-Jan Stol *, Brian Fitzgerald

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

@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/s

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

Science of Computer Programming 101 (2015) 79–98

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/s

e engineering ☆

d

University of Limerick, Ireland

@SCP 201

Dagstuhl Seminar 22231

Theories of Programming

(Jun 06 – Jun 10, 2022)

@Dagstuhl 2022

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

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.

When Do Changes Induce Fixes?

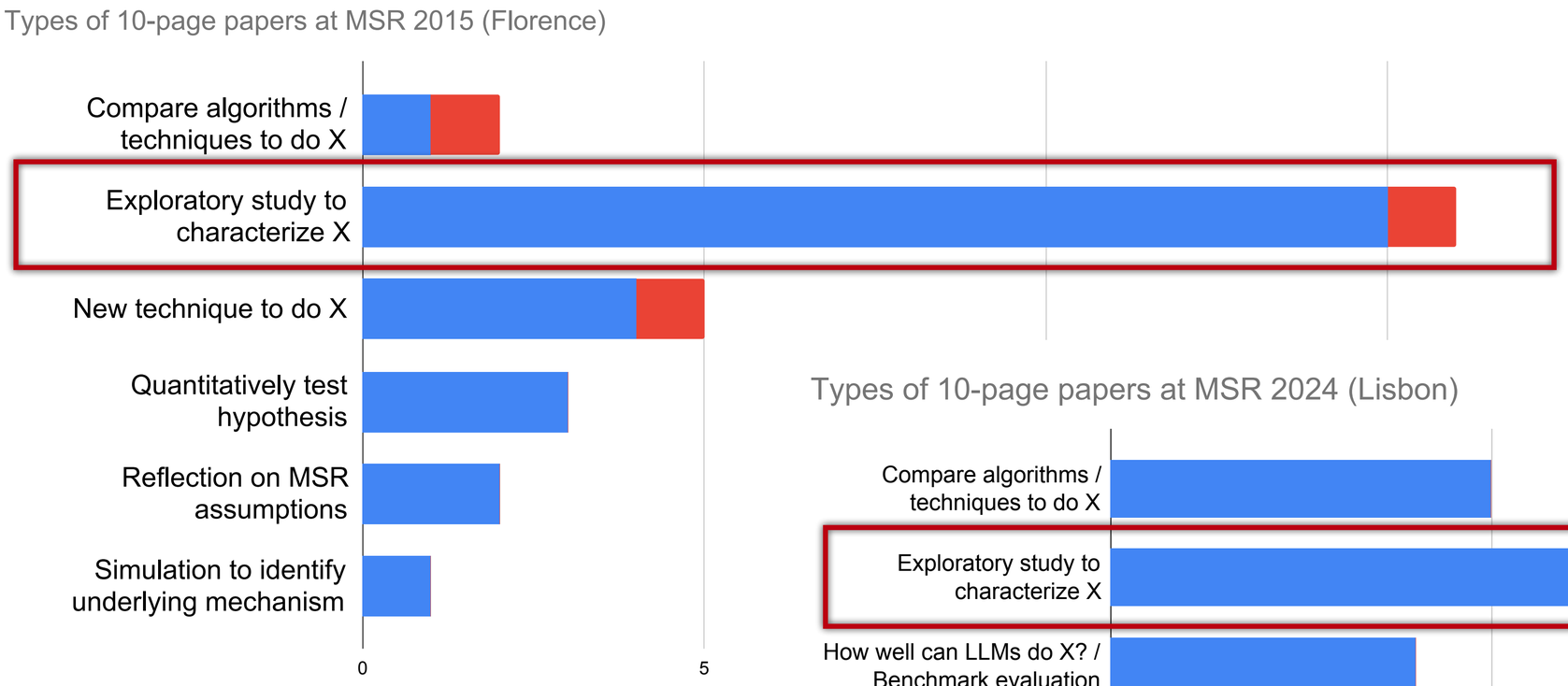
(On Fridays.)

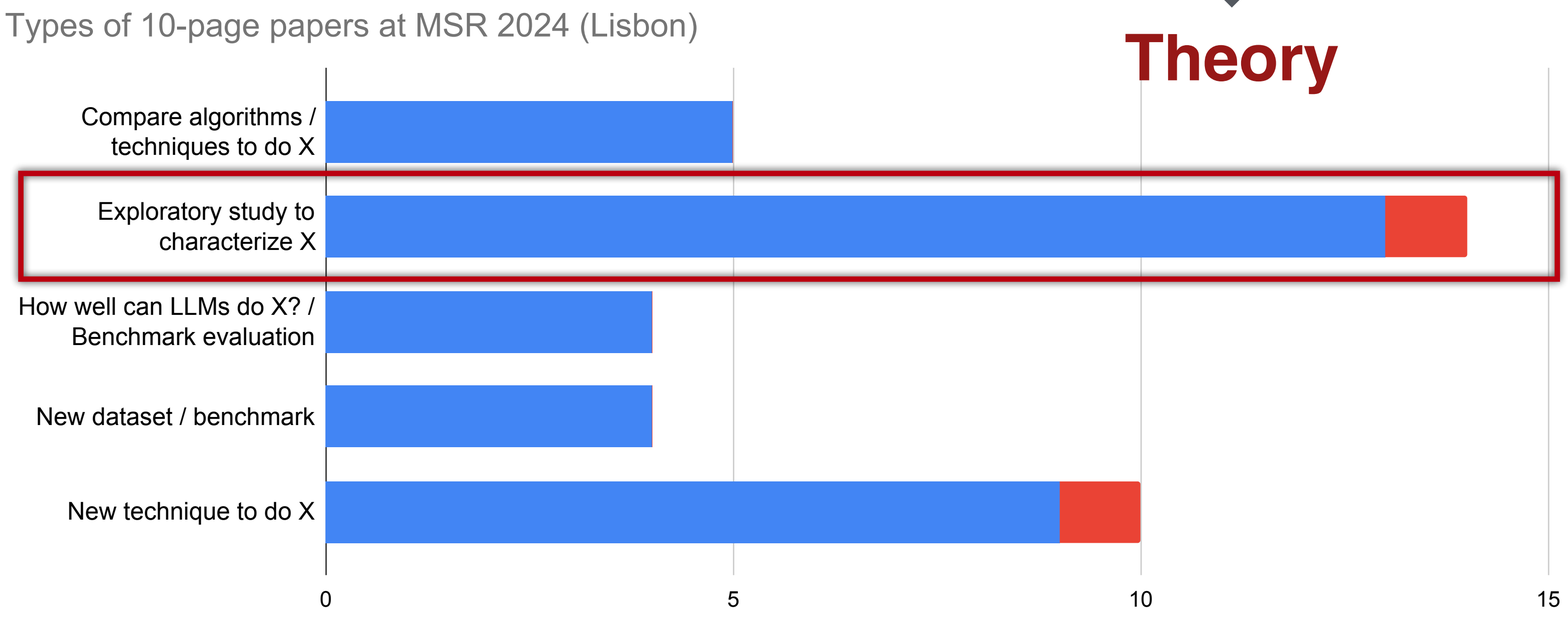
@MSR 2005

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”





Data collection

↓

Interpretation of findings

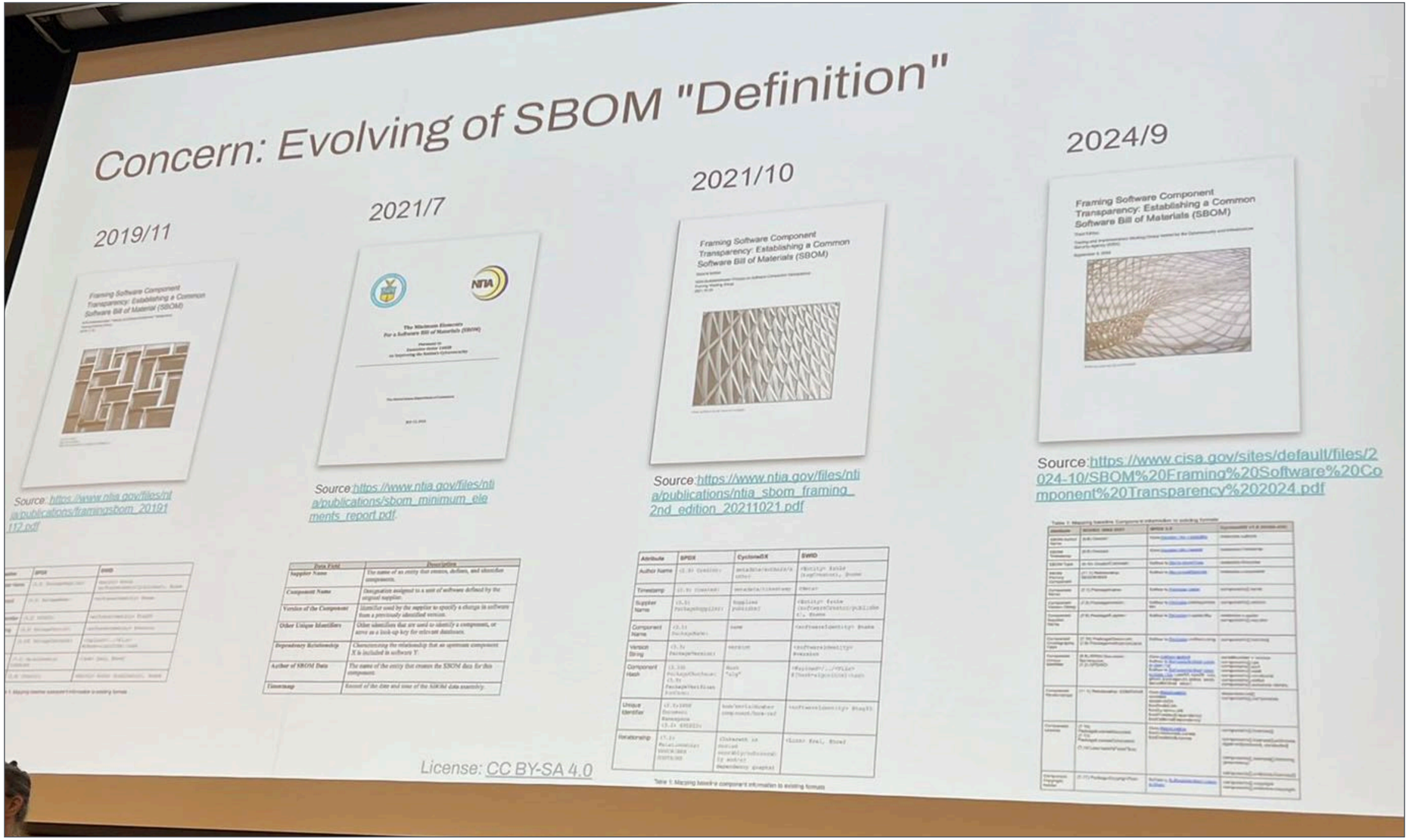
↓

Propositions / Hypotheses

↓

Theory

SBOMs are a kind of theory



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

↓

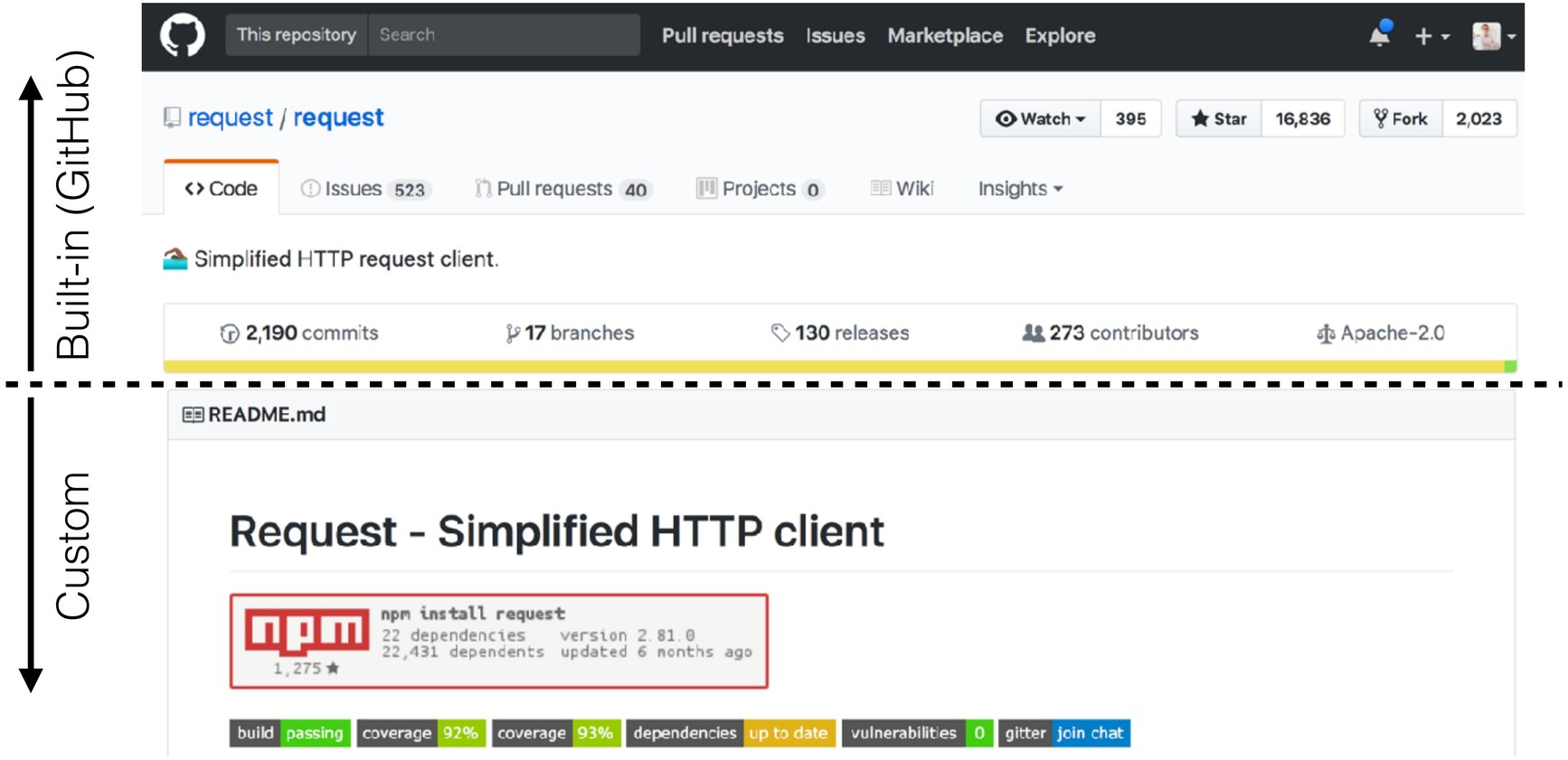
Interpretation of findings

↓

Validation / Refinement

Example: Signaling theory (Spence, 1973)

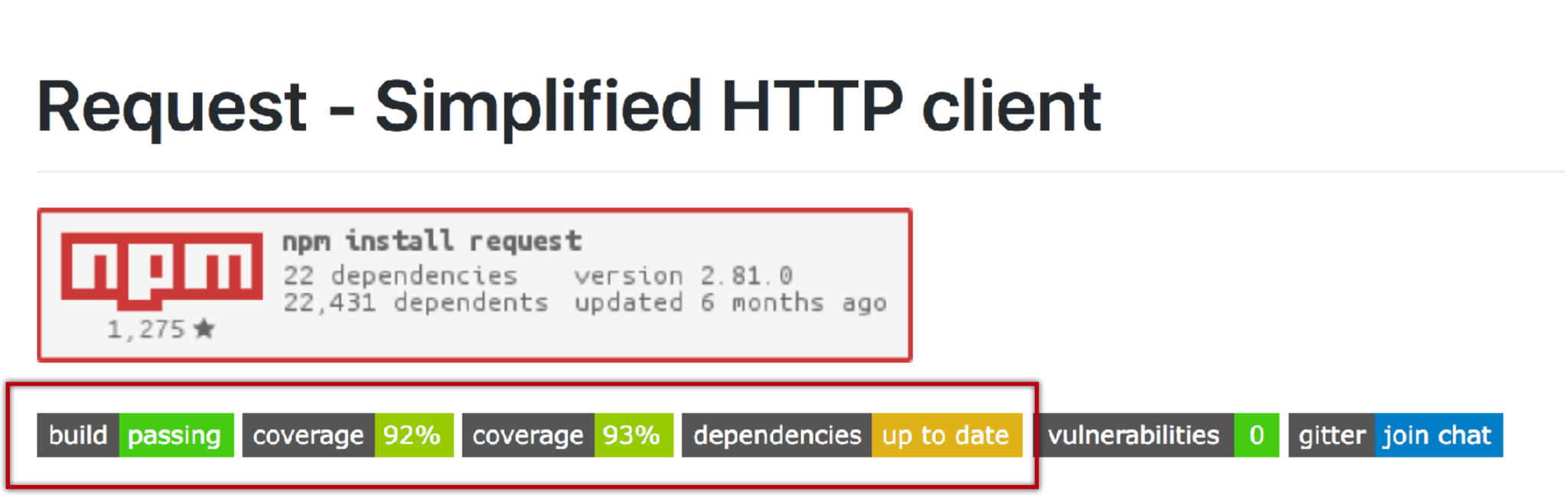
People use the visible cues on the platform as signals, to make rick inferences about unobservable traits of other users or projects.



- 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 rick inferences about unobservable traits of other users or projects.



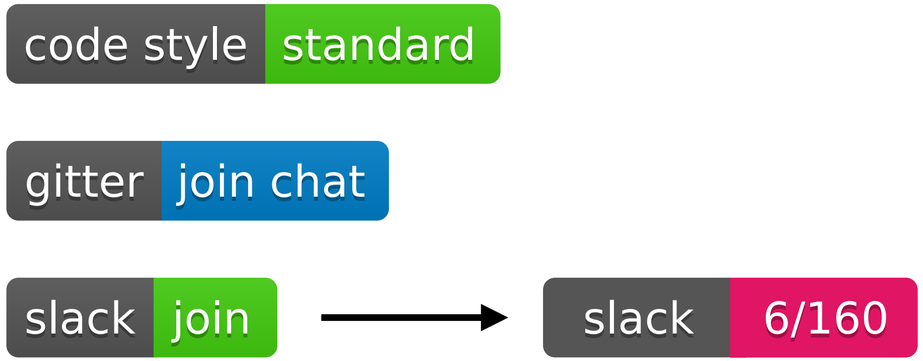
Signals of code quality

- 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!

Theory

↓

Hypotheses

↓

Data collection

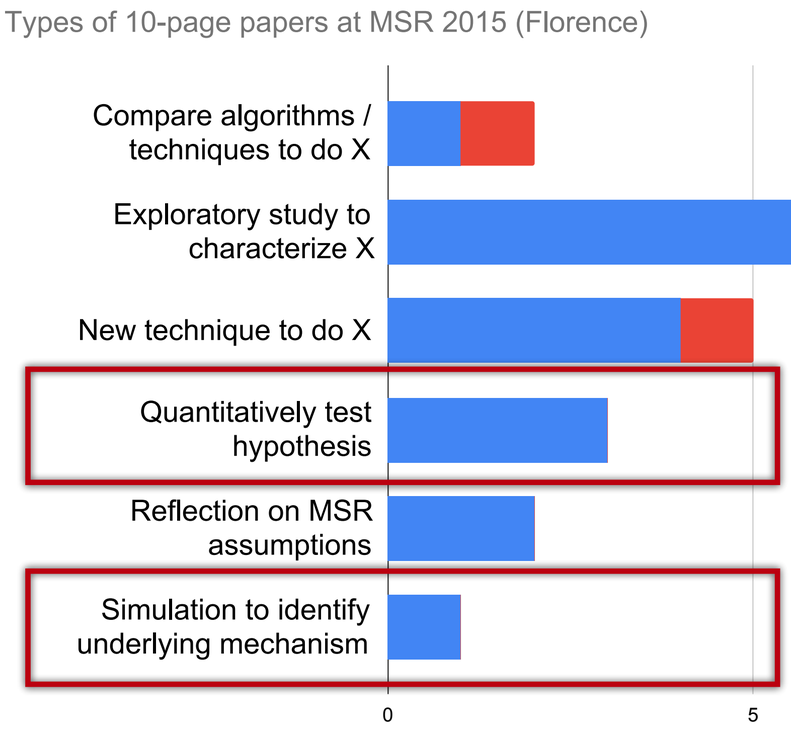
↓

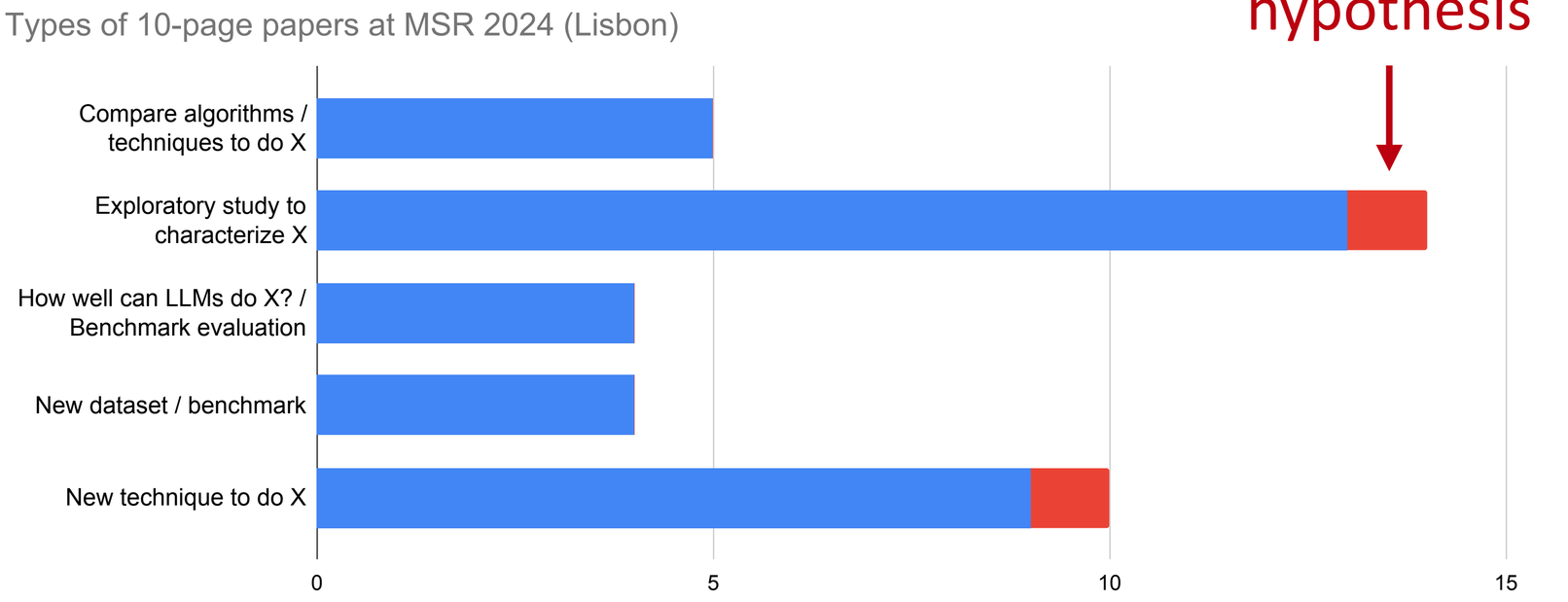
Interpretation of findings

↓

Validation / Refinement

We have very little deductive use of theories





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?”

So are Menzies & Shepperd:

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

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

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

Information and Software Technology 112 (2019) 35–47

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

“Bad smells” in software analytics papers

Tim Menzies[a,*], Martin Shepperd[b]

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

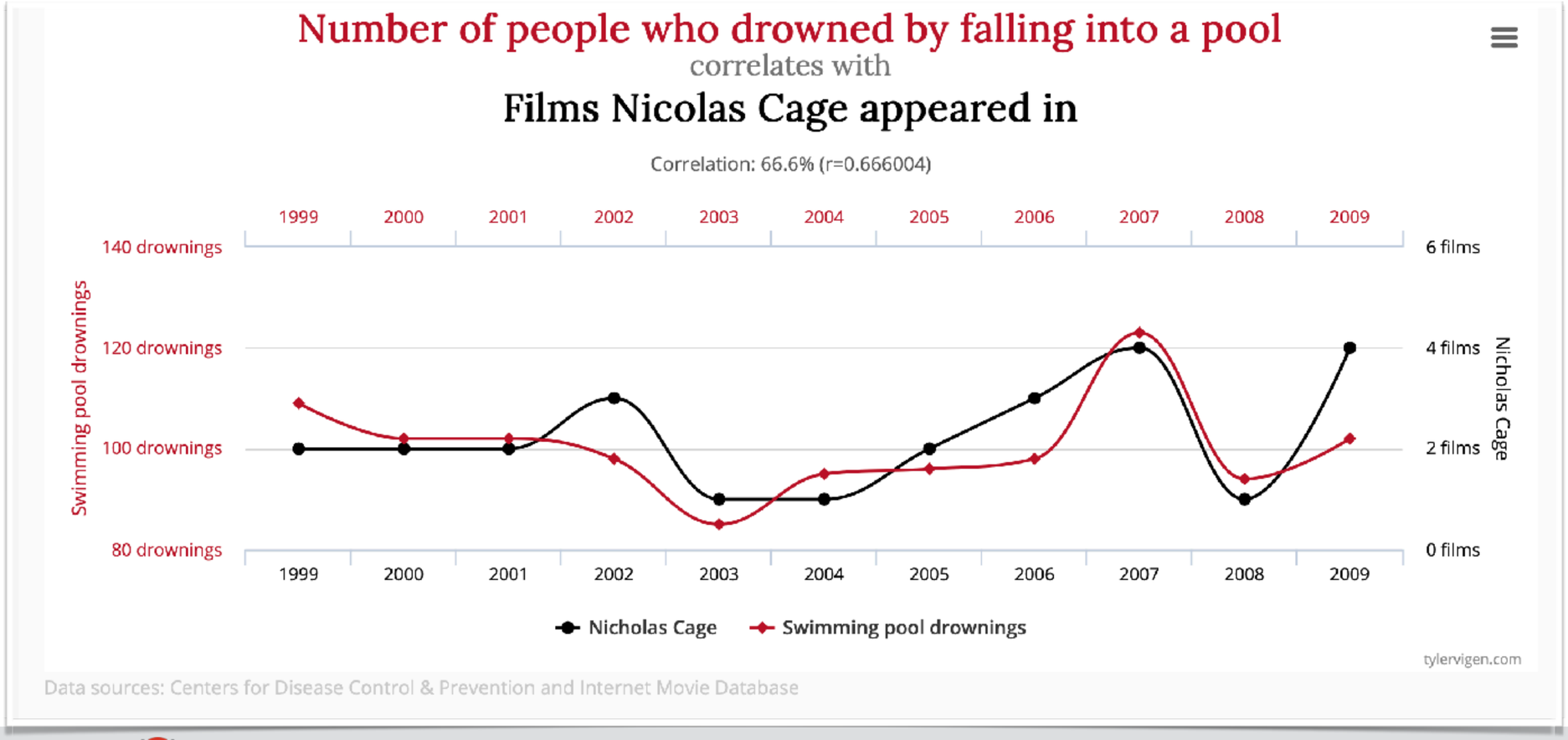
Now what?

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.



http://www.tylervigen.com/spurious-correlations

Ingredients for establishing a causal relationship?

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

X —> Y when:

-
-
-

Ingredients for establishing a causal relationship?

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

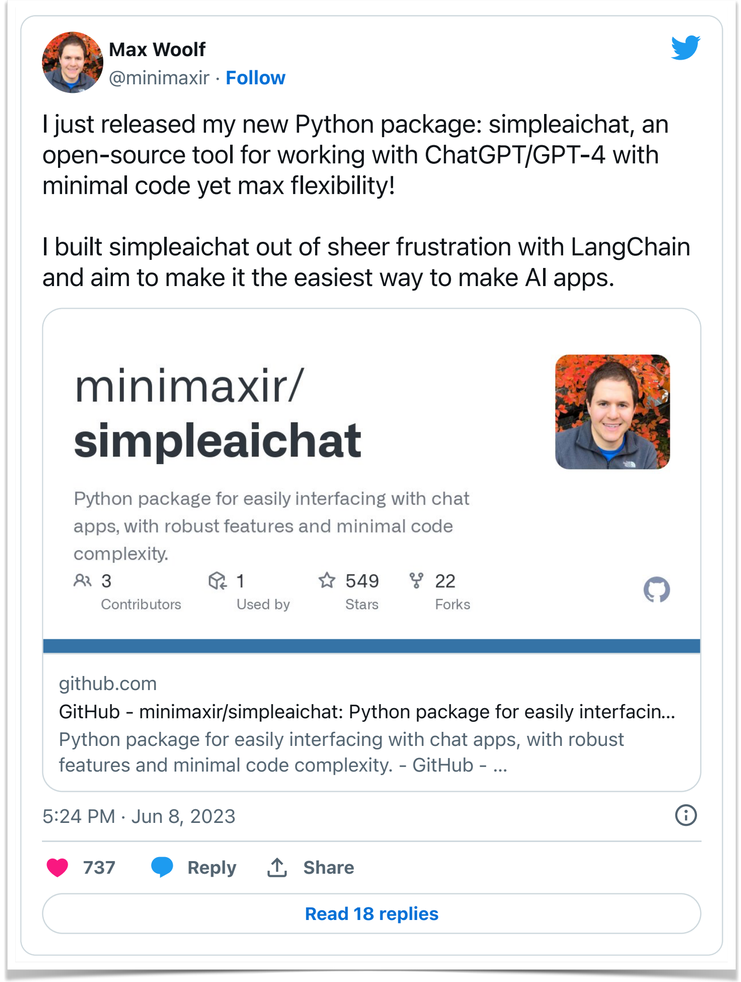
X —> Y when:

- 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

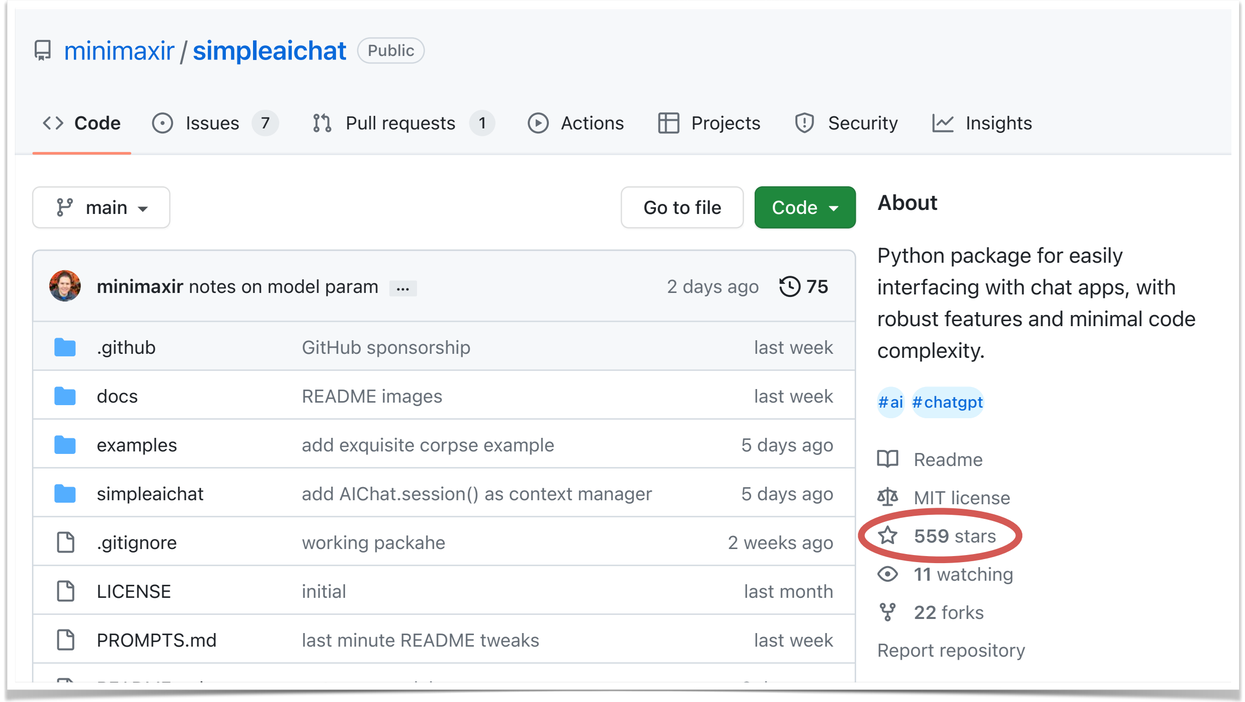
- A good theory both explains **how** and **why** certain phenomena occur, and allows **predictions** to be made.
  - Causal relationships allow for stronger predictions
  - Bonus points if we validate the mechanism
- There are lots of techniques for **causal inference from observational data**.
  - 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?



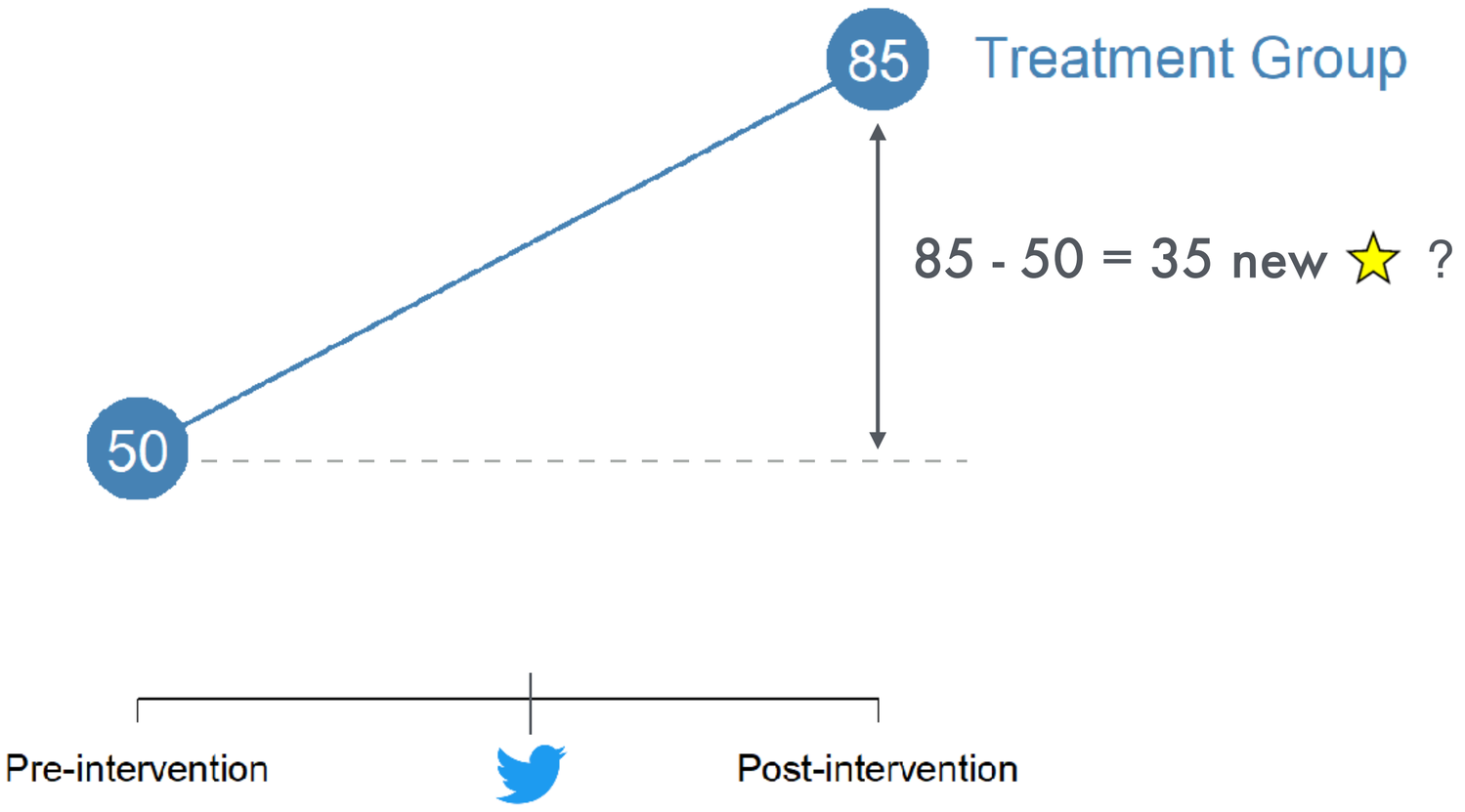
- Fang, Lamba, Herbsleb, & Vasilescu. “This is damn slick!” Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

Example: Do tweets cause GitHub stars?



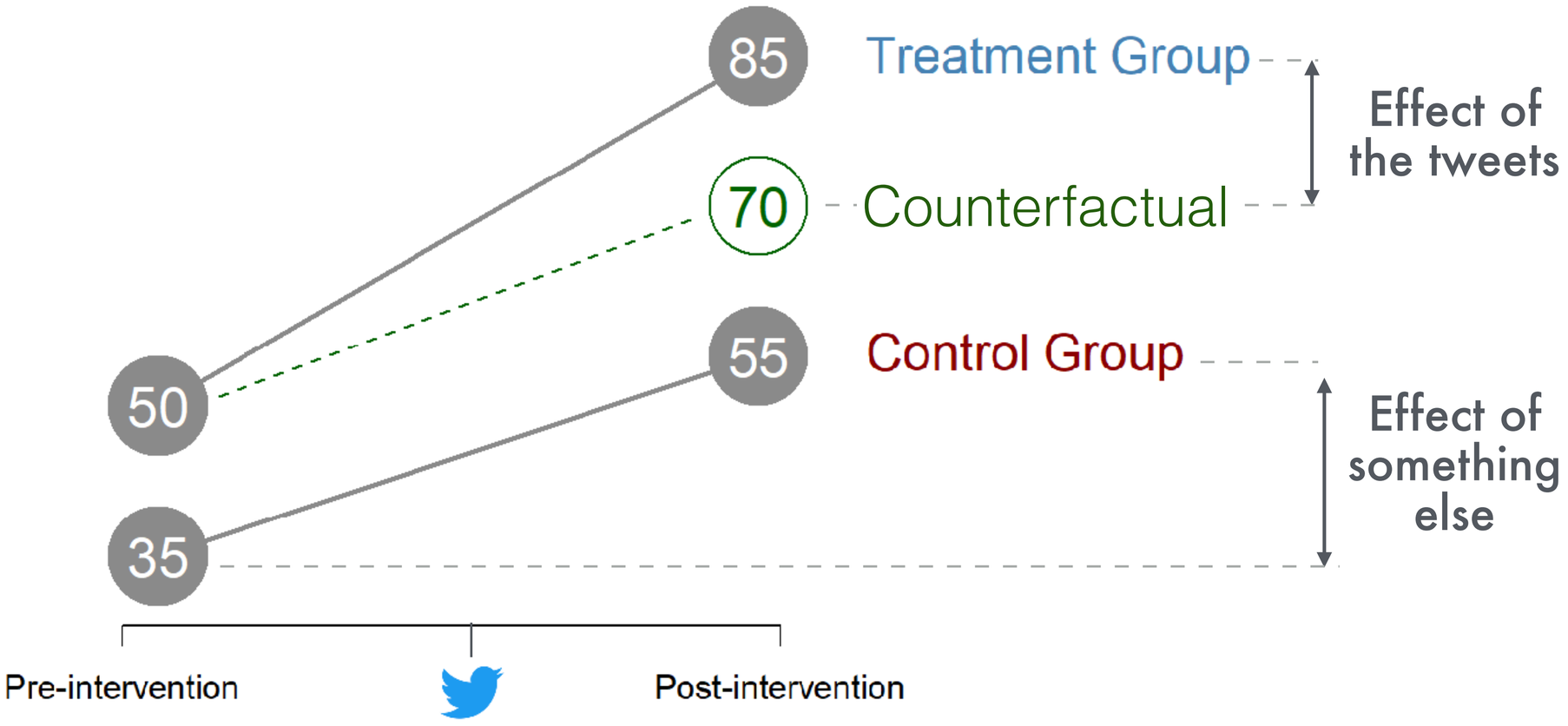
- Fang, Lamba, Herbsleb, & Vasilescu. “This is damn slick!” Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

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



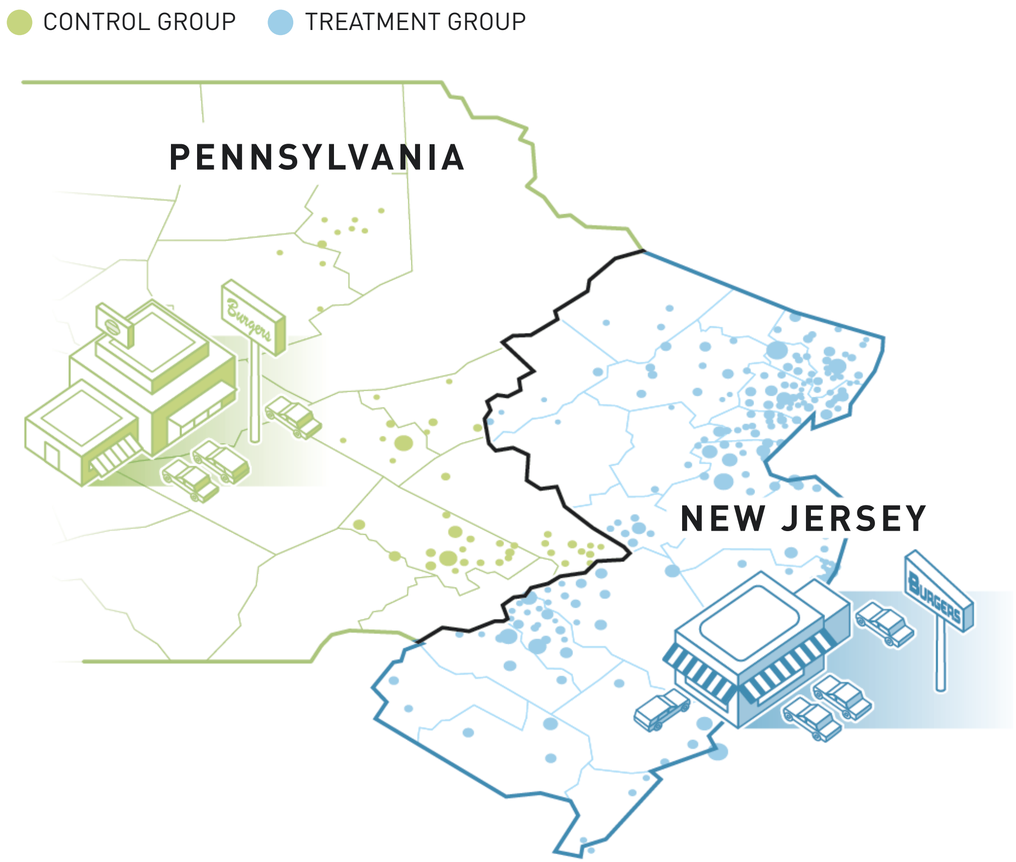
- Fang, Lamba, Herbsleb, & Vasilescu. “This is damn slick!” Estimating the impact of tweets on open source project popularity and new contributors. ICSE 2022

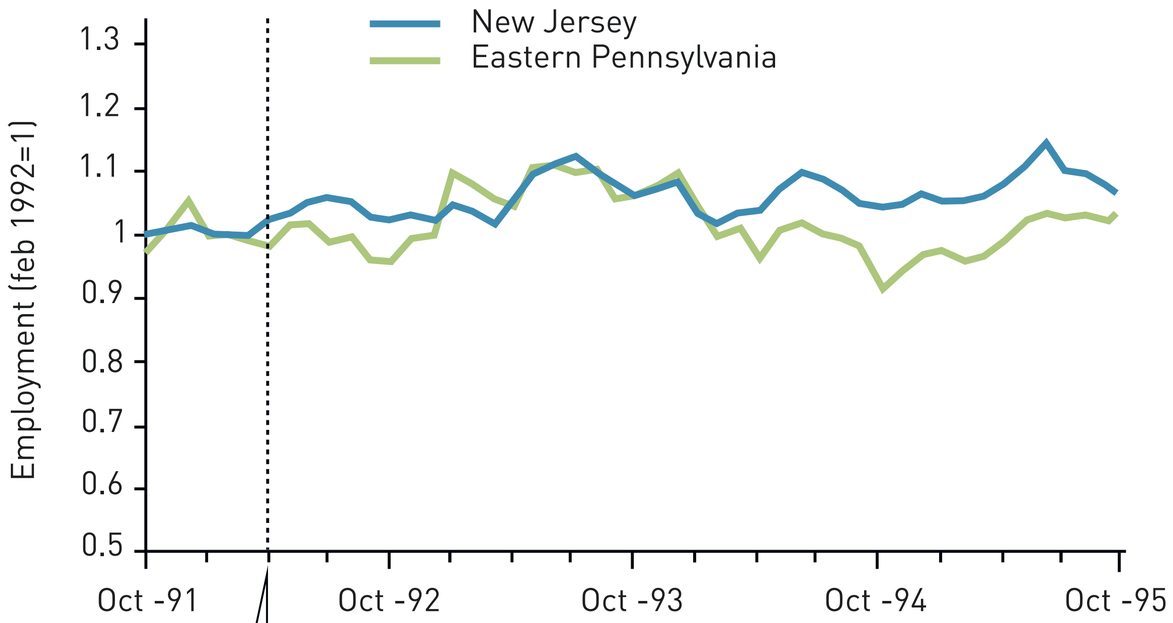
Better idea: Compare that change to the change in an appropriate control group



- 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.





1 April 1992: The hourly minimum wage in New Jersey was increased from 4.25 dollars to 5.05 dollars. Despite this, employment in New Jersey was not affected.

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.

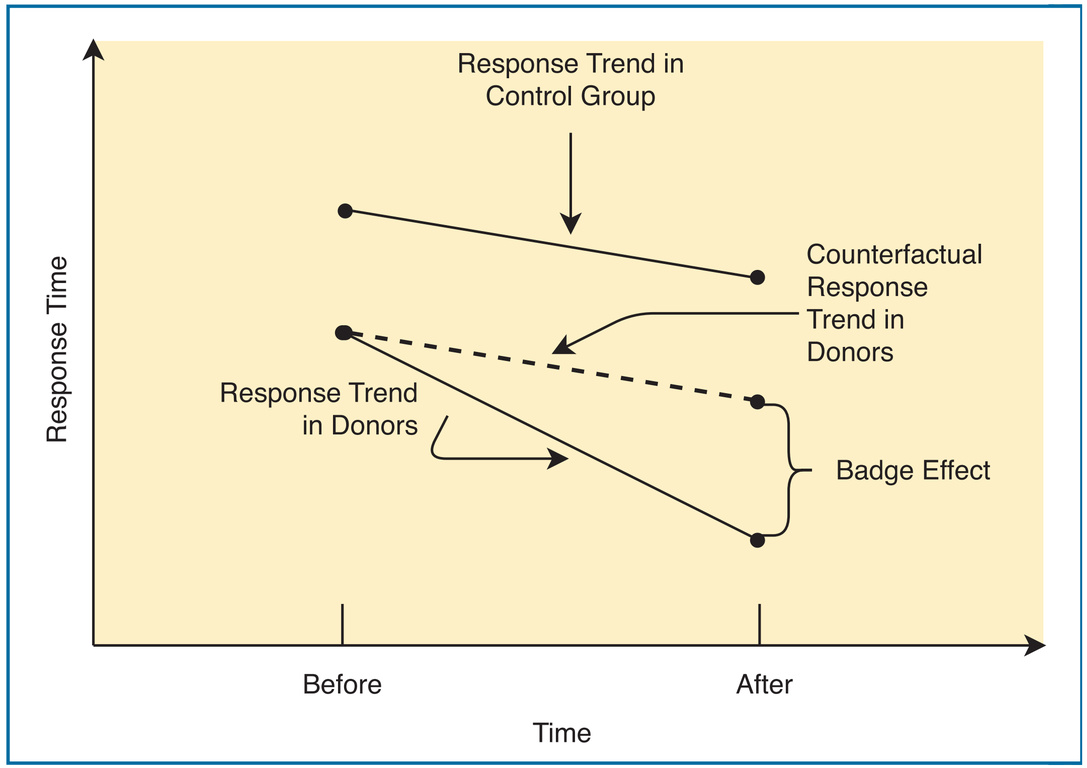
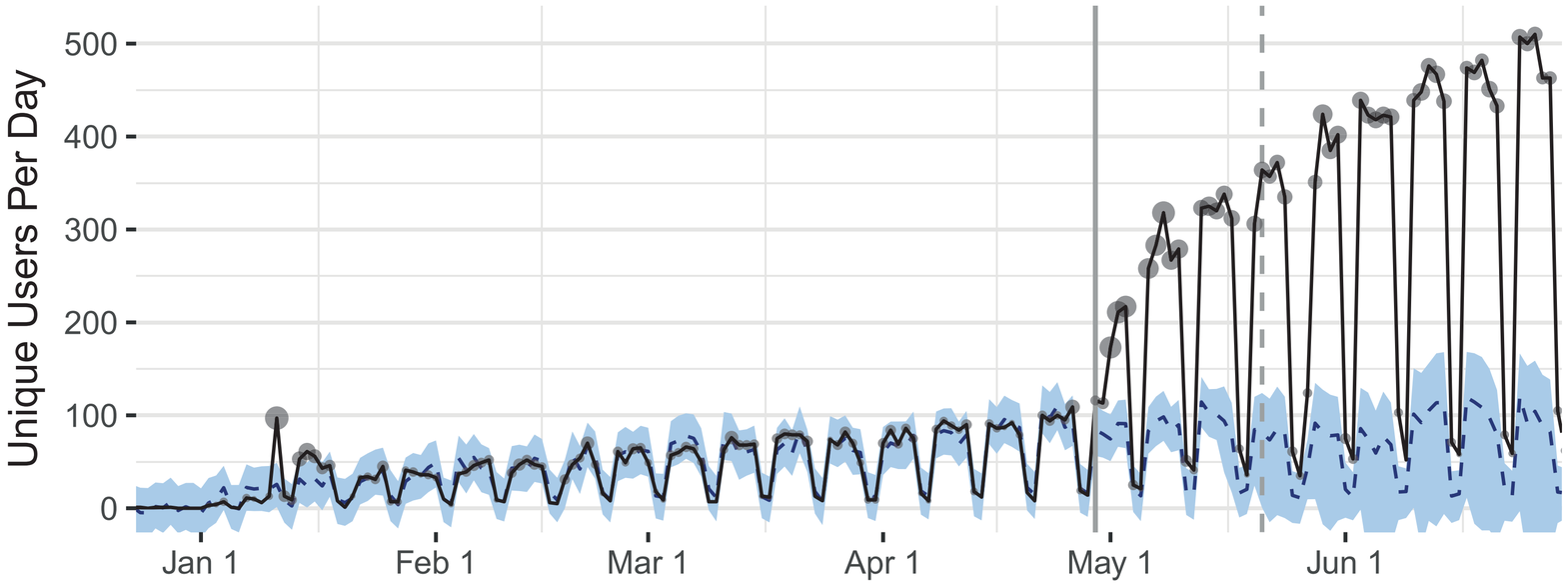


FIGURE 1. An example of the causal inference framework using a DID model showing response time before versus after the introduction of donation badges.

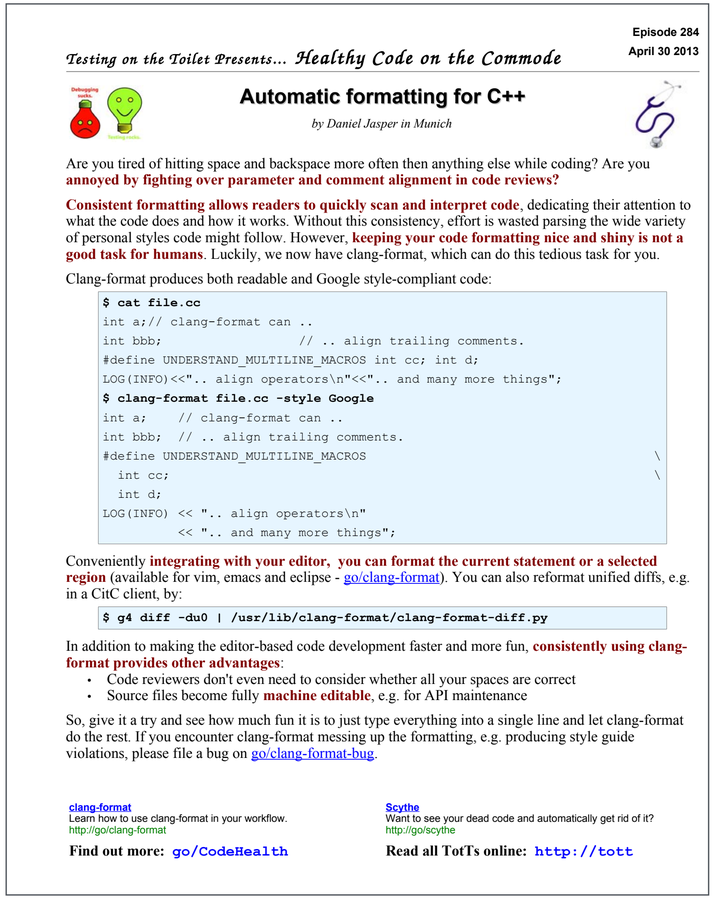
- Nakasai, Hata, & Matsumoto. Are donation badges appealing?: A case study of developer responses to Eclipse bug reports. IEEE Software 2018.

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

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



(a) CLANG-FORMAT



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

Episode 284
April 30 2013

Automatic formatting for C++

by Daniel Jasper in Munich

Are you tired of hitting space and backspace more often then 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 CitC 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

Scythe
Want to see your dead code and automatically get rid of it?
http://go/scythe

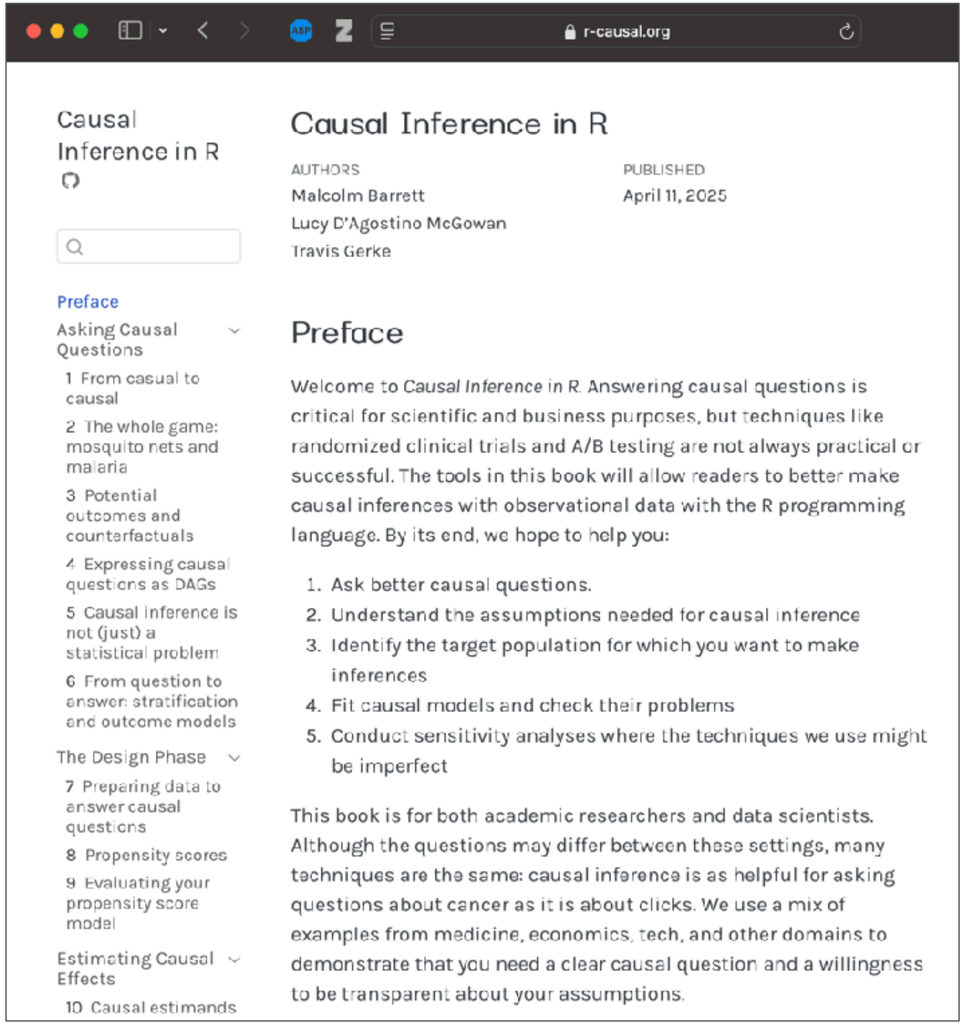
Find out more: go/CodeHealth

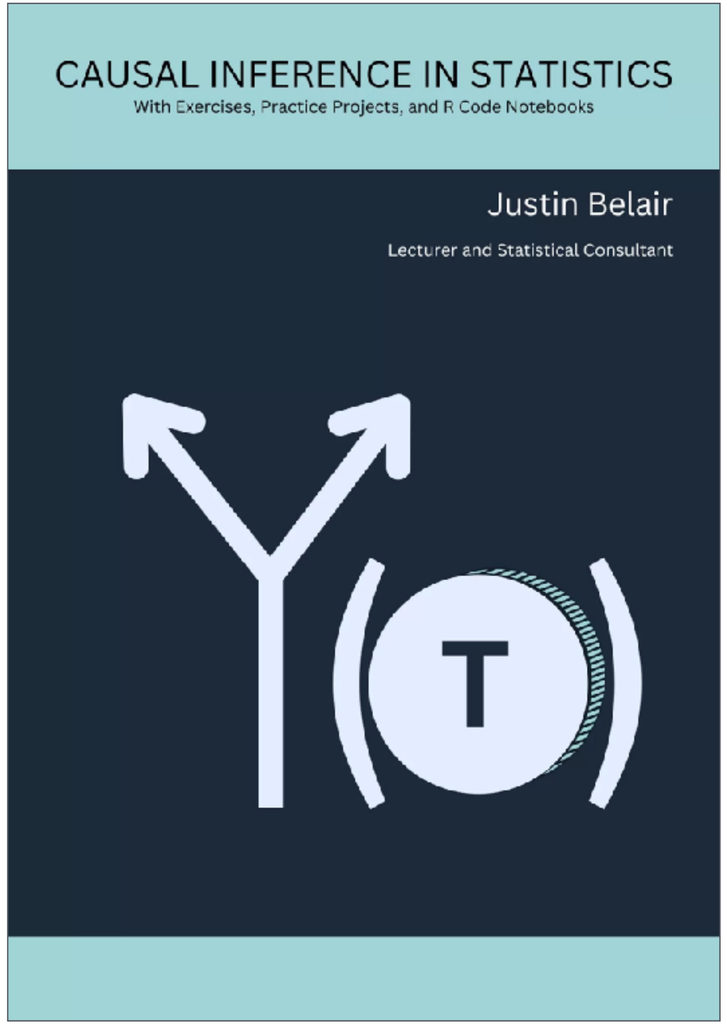
Read all TotTs online: http://tott

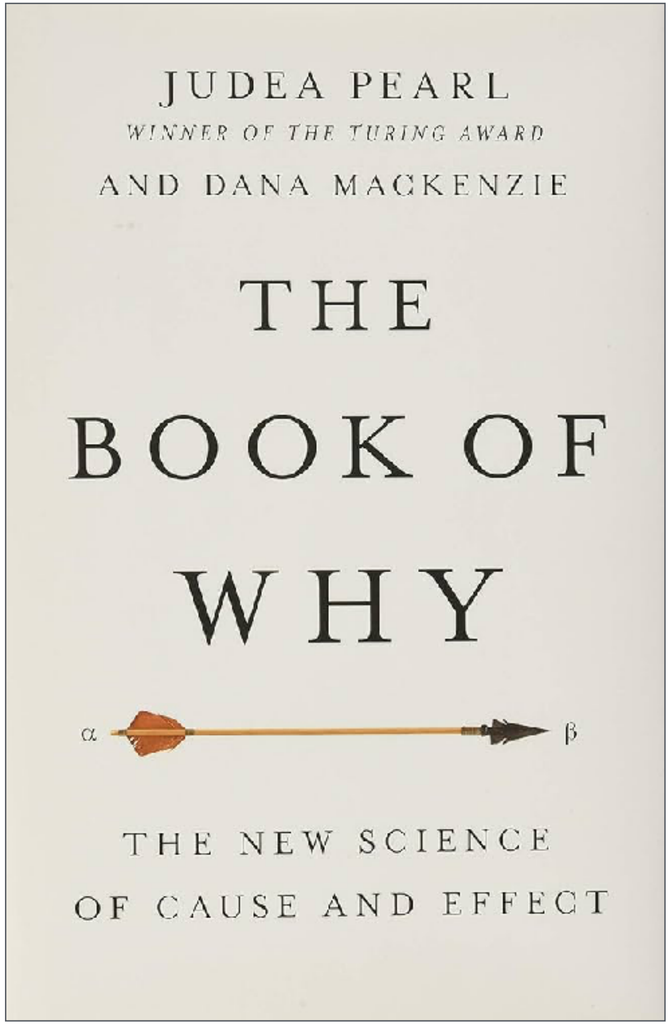
- Murphy-Hill, Smith, Sadowski, et al. Do developers discover new tools on the toilet?. ICSE 2019

Where to start

r-causal.org







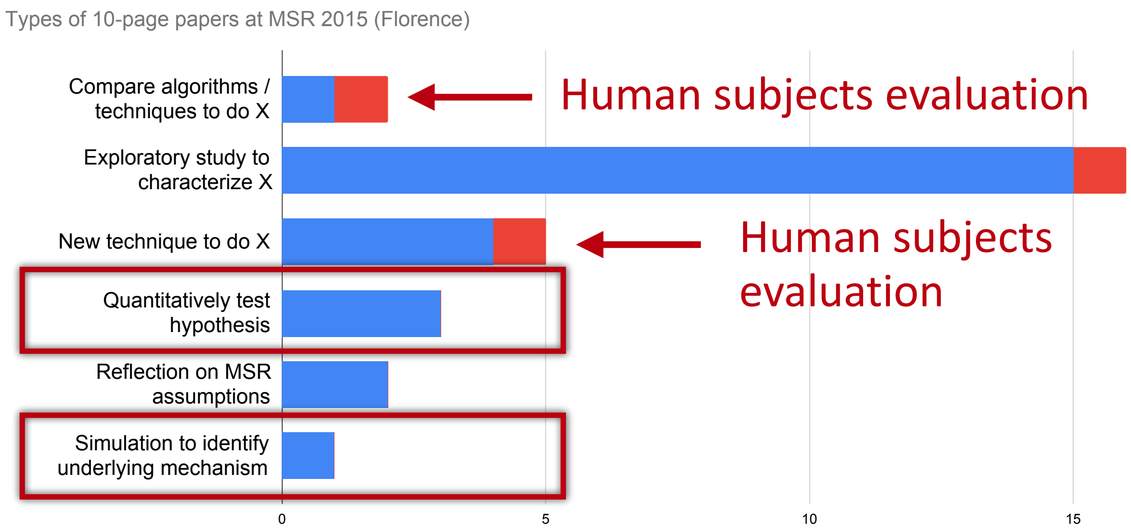
Summary: We are a methods conference, let's step up our methods game!

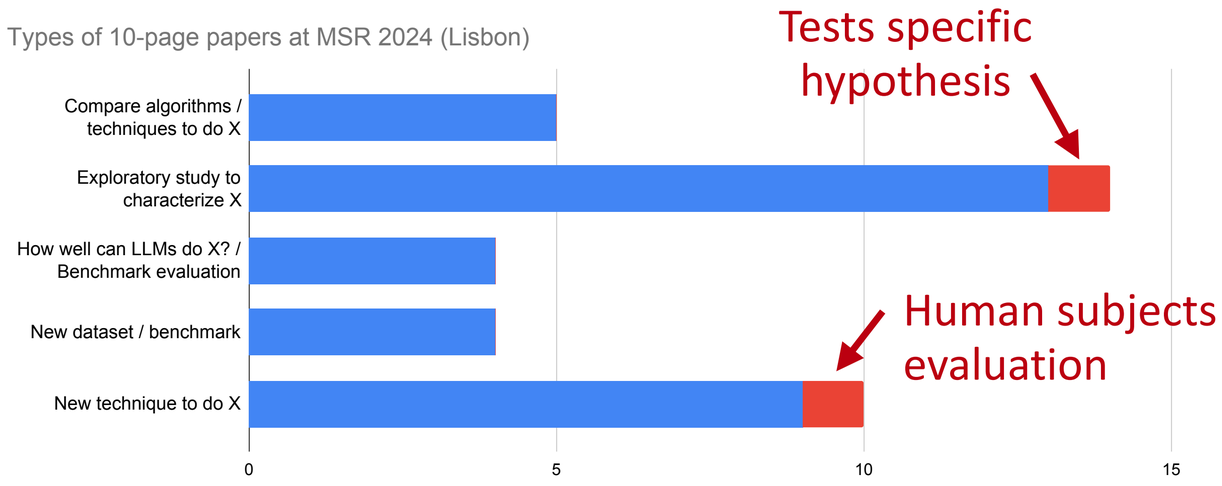
Understanding the problem:

- 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?





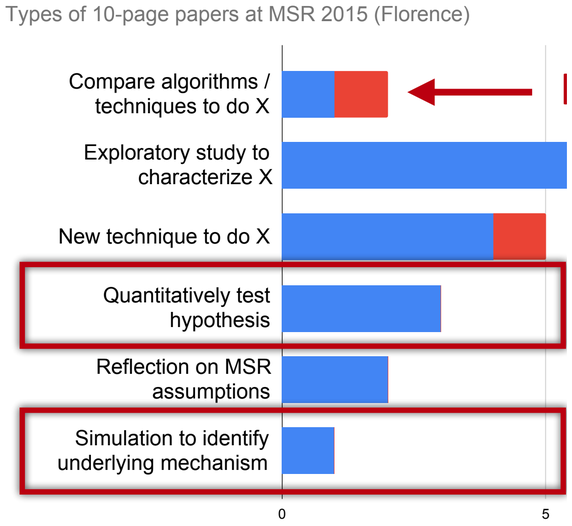
Summary: We are a methods conference, let's step up our methods game!

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?



COUNT_SCRATCHULA