A System for Graph-Based Visualization of the Evolution of Software
—
A (Short) Exercise in Reproduction

Christian Collberg
Stephen Kobourov
Jasvir Nagra
Jacob Pitts
Kevin Wampler
Todd Proebsting
Keith Alcock

SOFTVIS 2003
—
VISSOFT 2019

Department of Computer Science
University of Arizona

http://repeatability.cs.arizona.edu
http://findresearch.org

http://graphael.cs.arizona.edu
http://sandmark.cs.arizona.edu

A System for Graph-Based Visualization of the Evolution of Software

Christian Collberg[1*] Stephen Kobourov[1†] Jasvir Nagra[2‡] Jacob Pitts[1] Kevin Wampler[1†]





Abstract

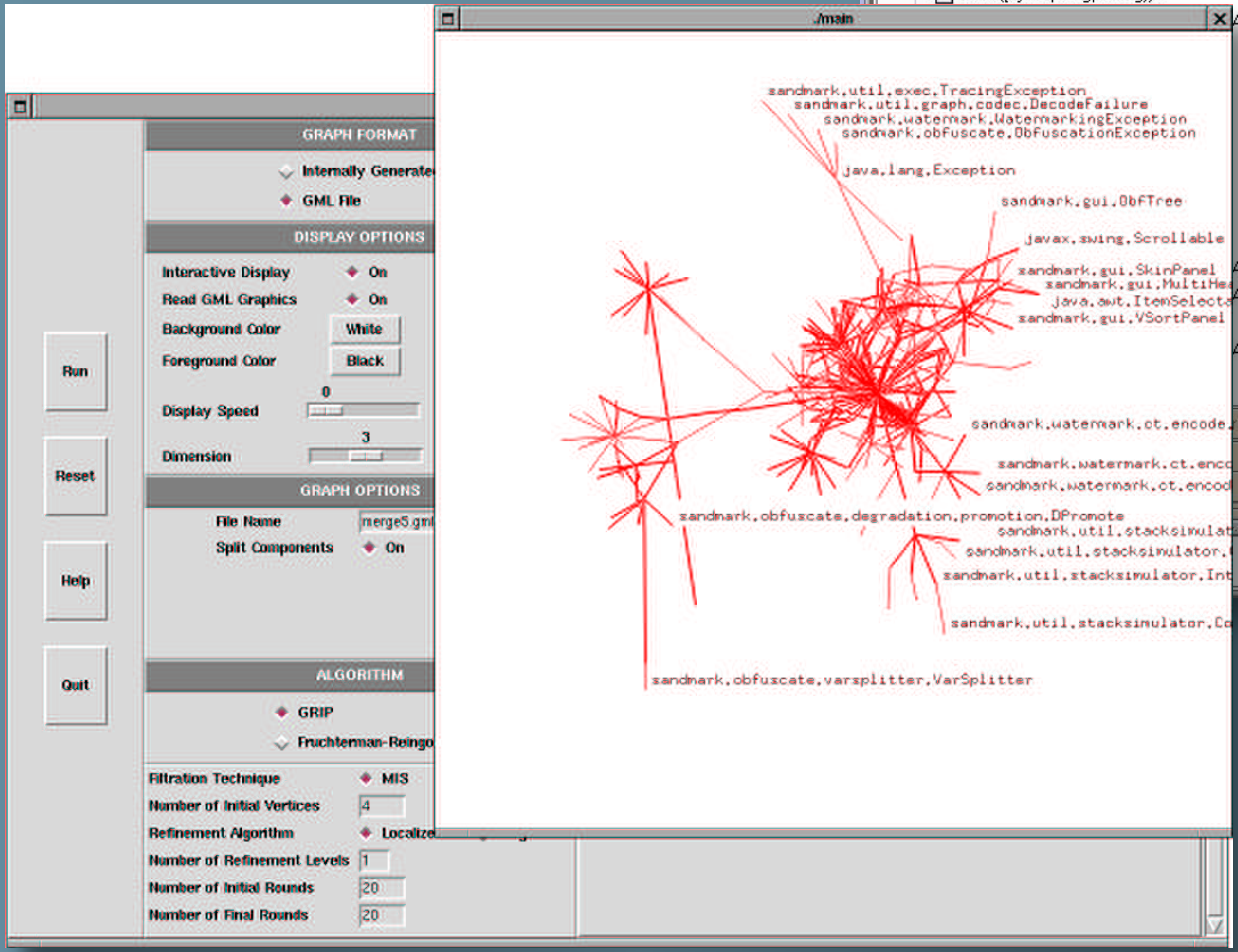
We describe GEVOL, a system that visualizes the evolution of software using a novel graph drawing technique for visualization of large graphs with temporal component. GEVOL extracts information about a Java program stored within a CVS version control system and displays it using a temporal graph visualizer. This information can be used by programmers to understand the evolution of a legacy program: Why is the program structured the way it is? Which programmers were responsible for which parts of the program during which time periods? Which parts of the program appear unstable over long periods of time and may need to be rewritten? This type of information will complement that produced by more static tools such as source code browsers, slicers, and static analyzers.

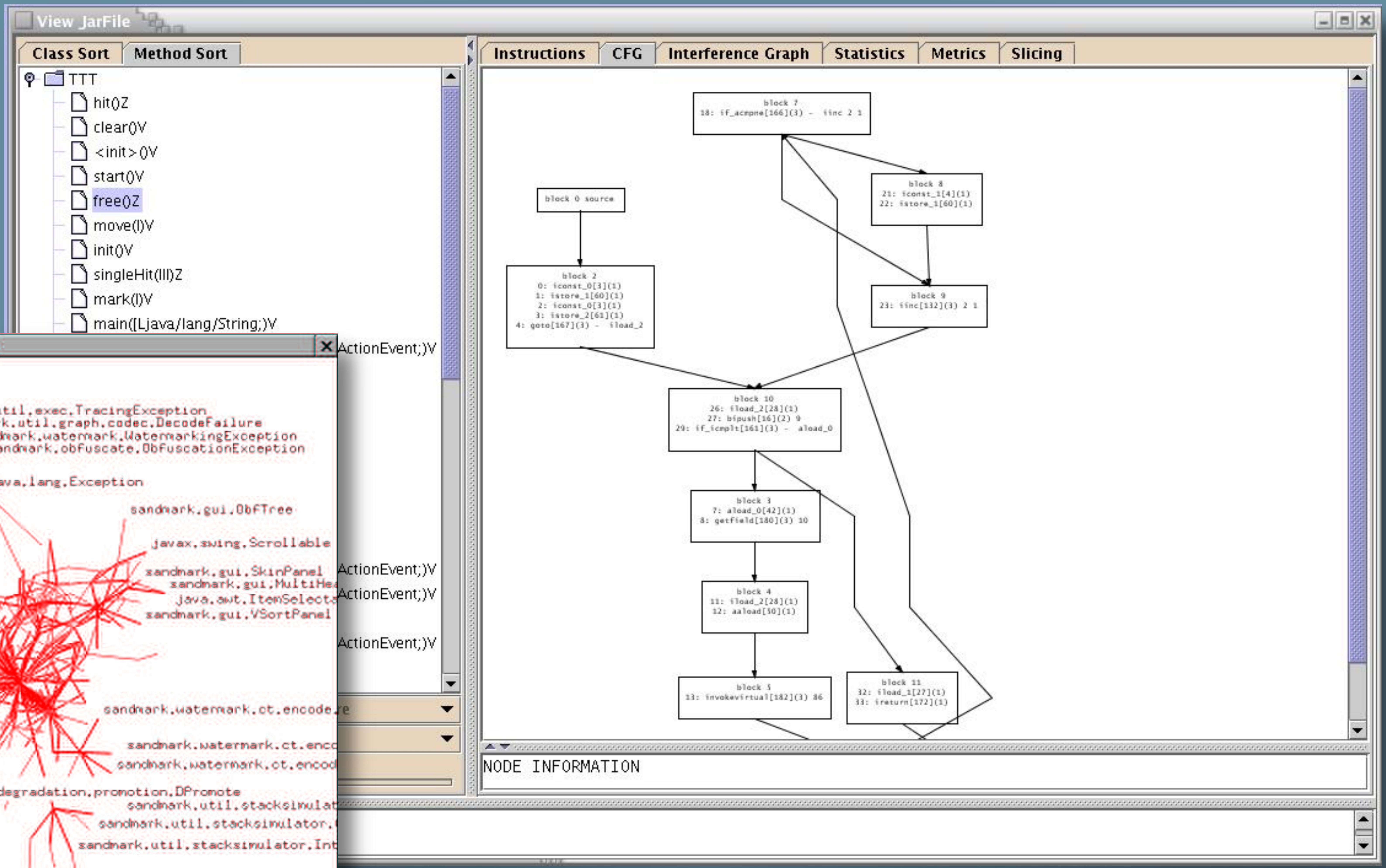
- Bob is asked to port P to a new operating system or architecture.

In many cases Bob will find that the program is undocumented, unstructured, and poorly written. Worse, the original developers may not be available to explain how the system works. Before he can start modifying the program he therefore needs to build a mental model of its structure. To aid in this discovery process he can run the program, examine the source code, and read any available documentation. Various tools such as source code browsers and static analyzers may be helpful in this respect.

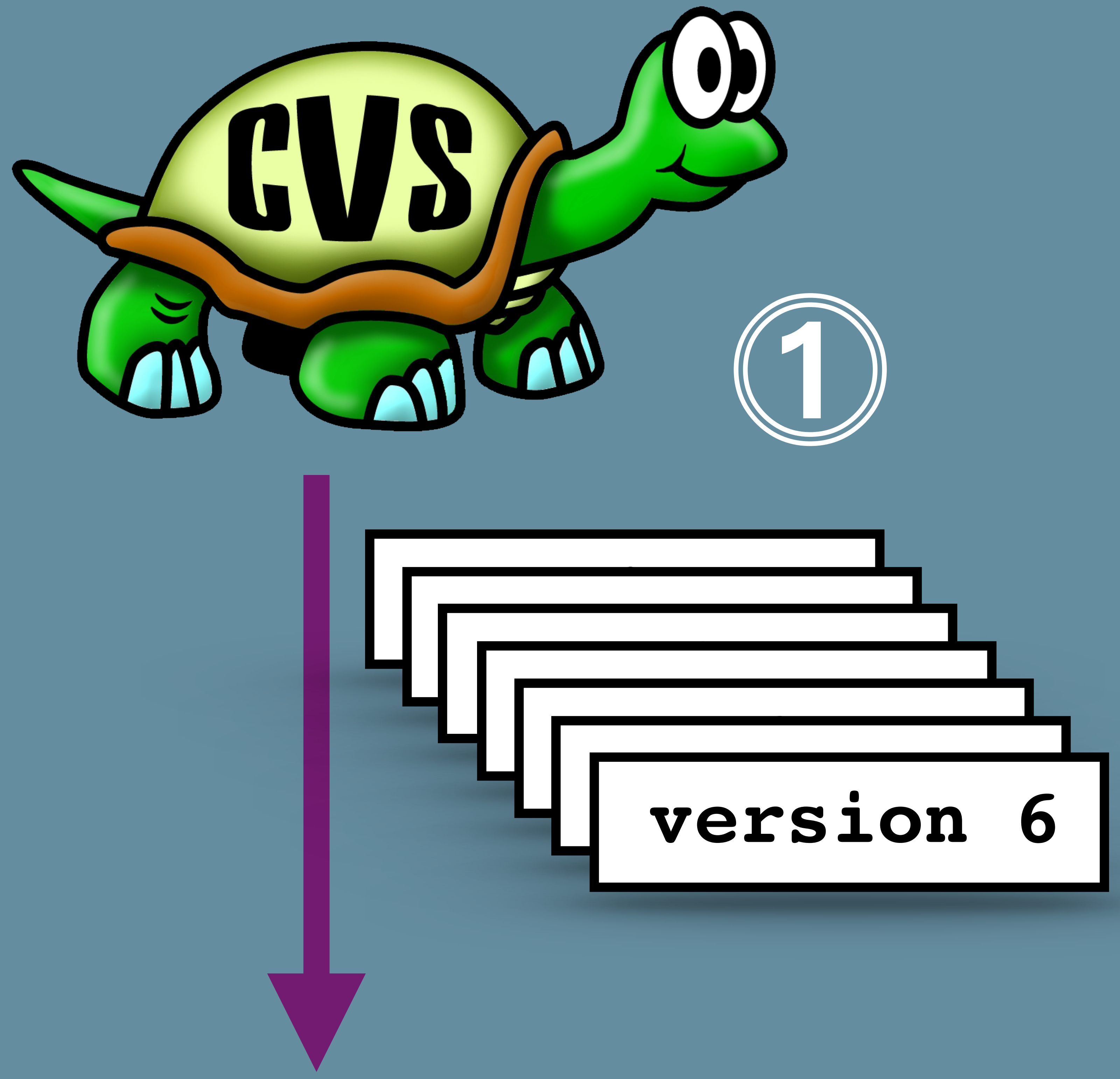
In this paper we will describe a new tool — GEVOL — that aid in the discovery of the structure of legacy systems. GEVOL discovers the *evolution* of a program by visualizing the changes the system has gone through. In particular, GEVOL extracts information about Java programs that are

tgrip



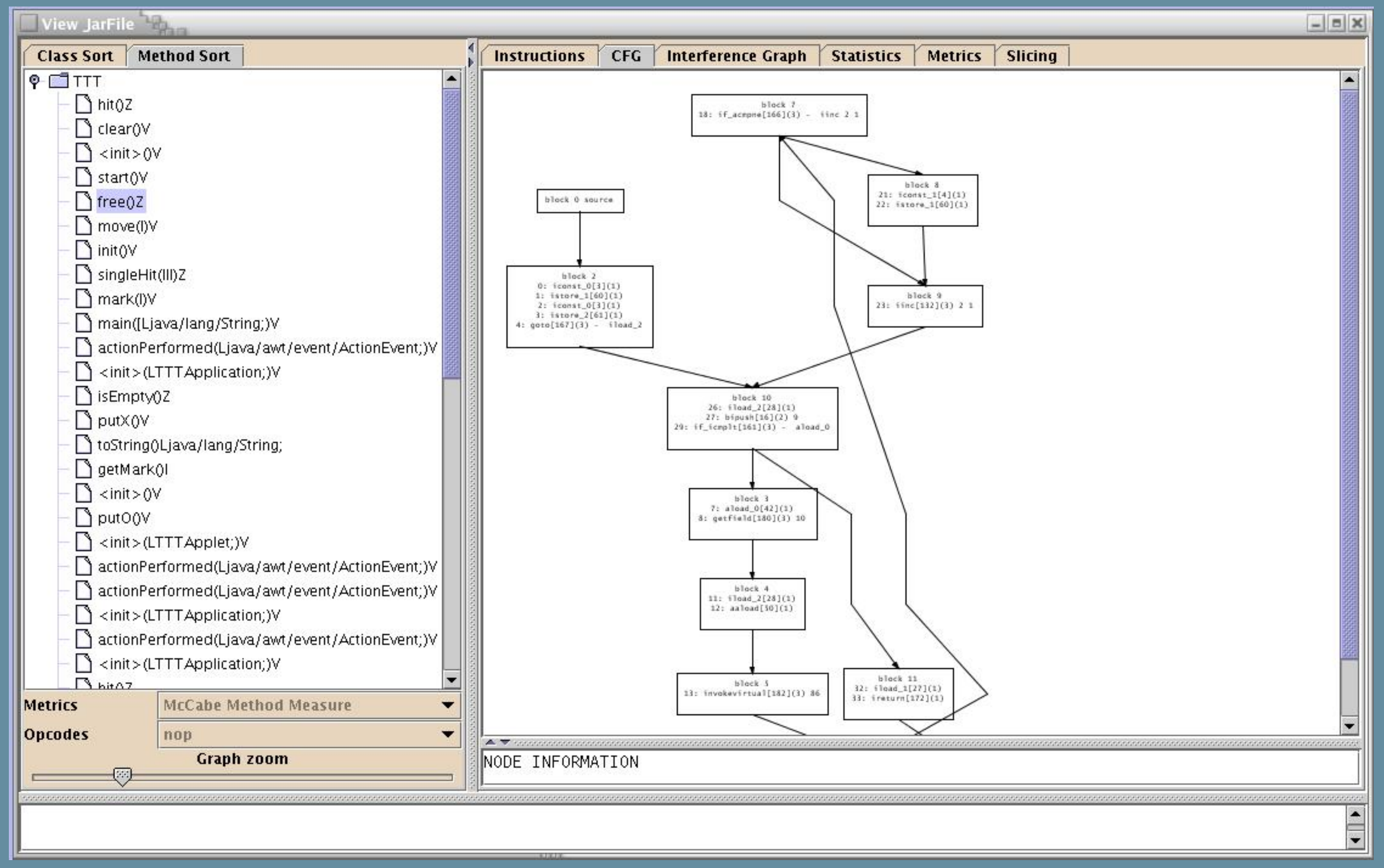


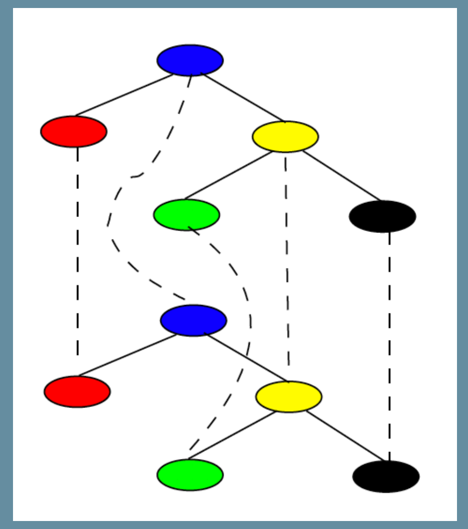
SandMark


CVS
1
version 6

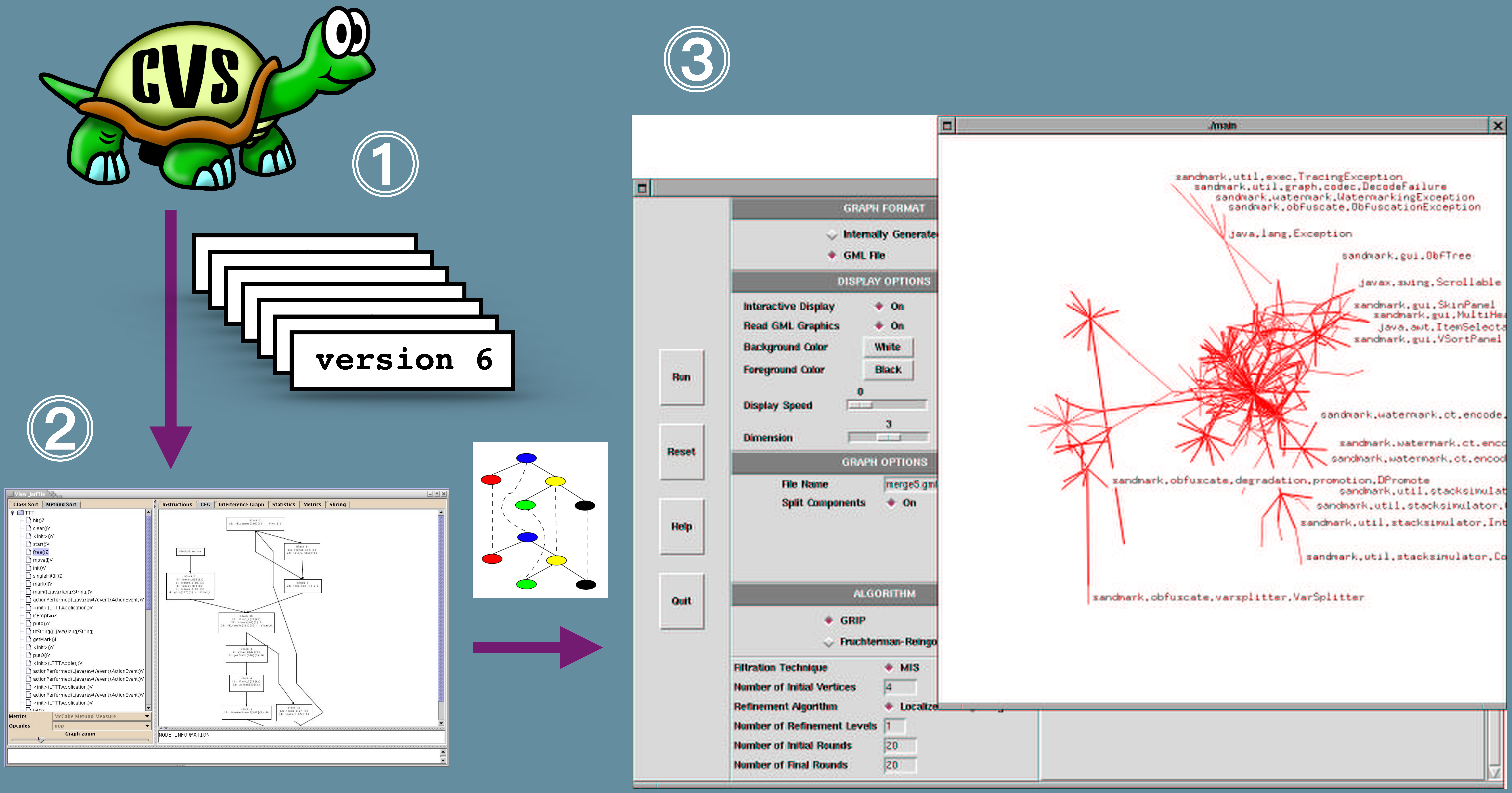
①

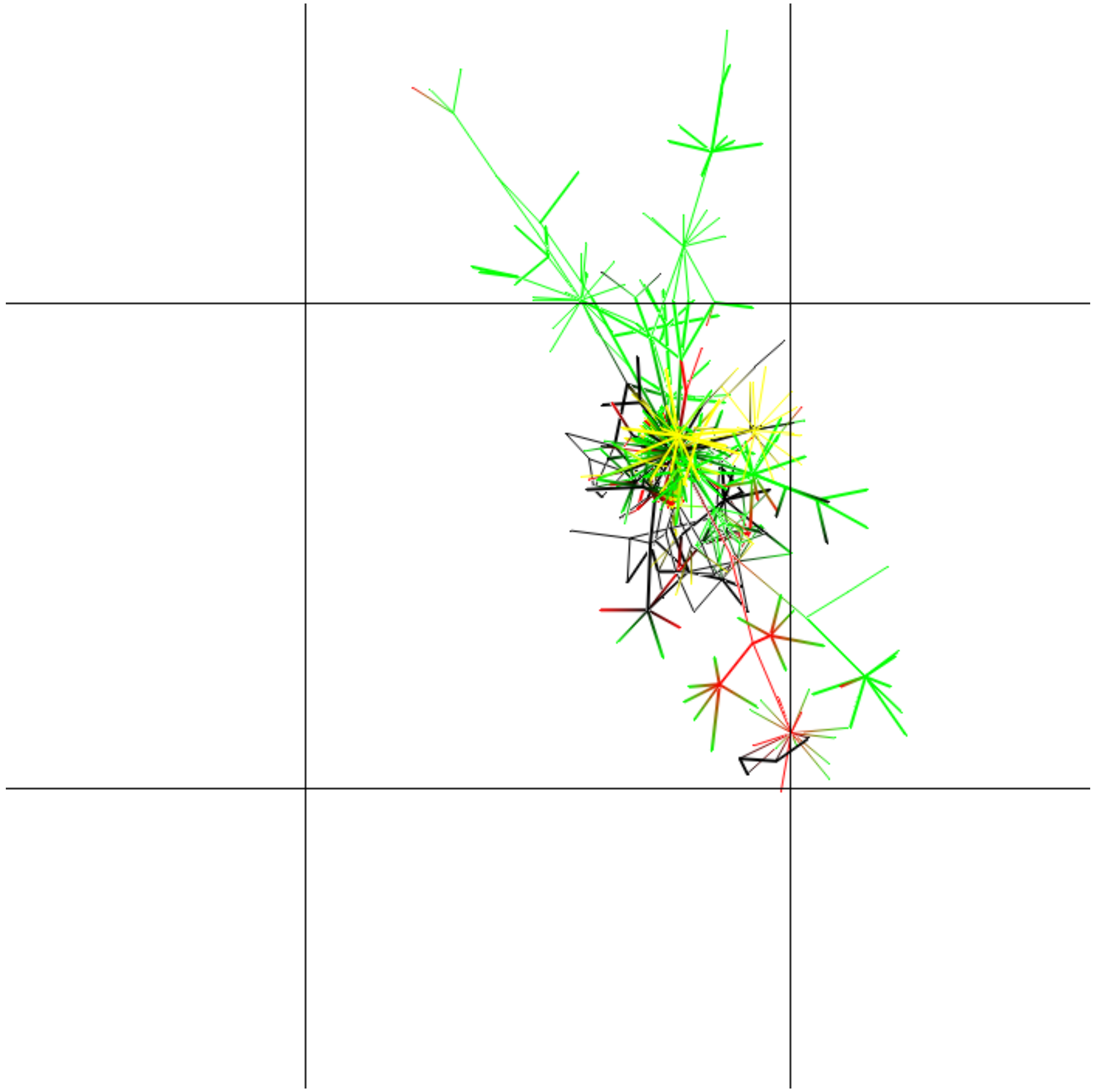
②





- control flow graphs
- inheritance graphs
- call graphs


CVS
①
version 6
②
③
GRAPH FORMAT
Internally Generated
GML File
DISPLAY OPTIONS
Interactive Display
On
Read GML Graphics
On
Background Color
White
Foreground Color
Black
Display Speed
Dimension
GRAPH OPTIONS
File Name
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingold
Filtration Technique
MIS
Number of Initial Vertices
Refinement Algorithm
Number of Refinement Levels
Number of Initial Rounds
Number of Final Rounds
Run
Reset
Help
Quit
./main
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote
sandmark.obfuscate.varsplitter.VarSplitter



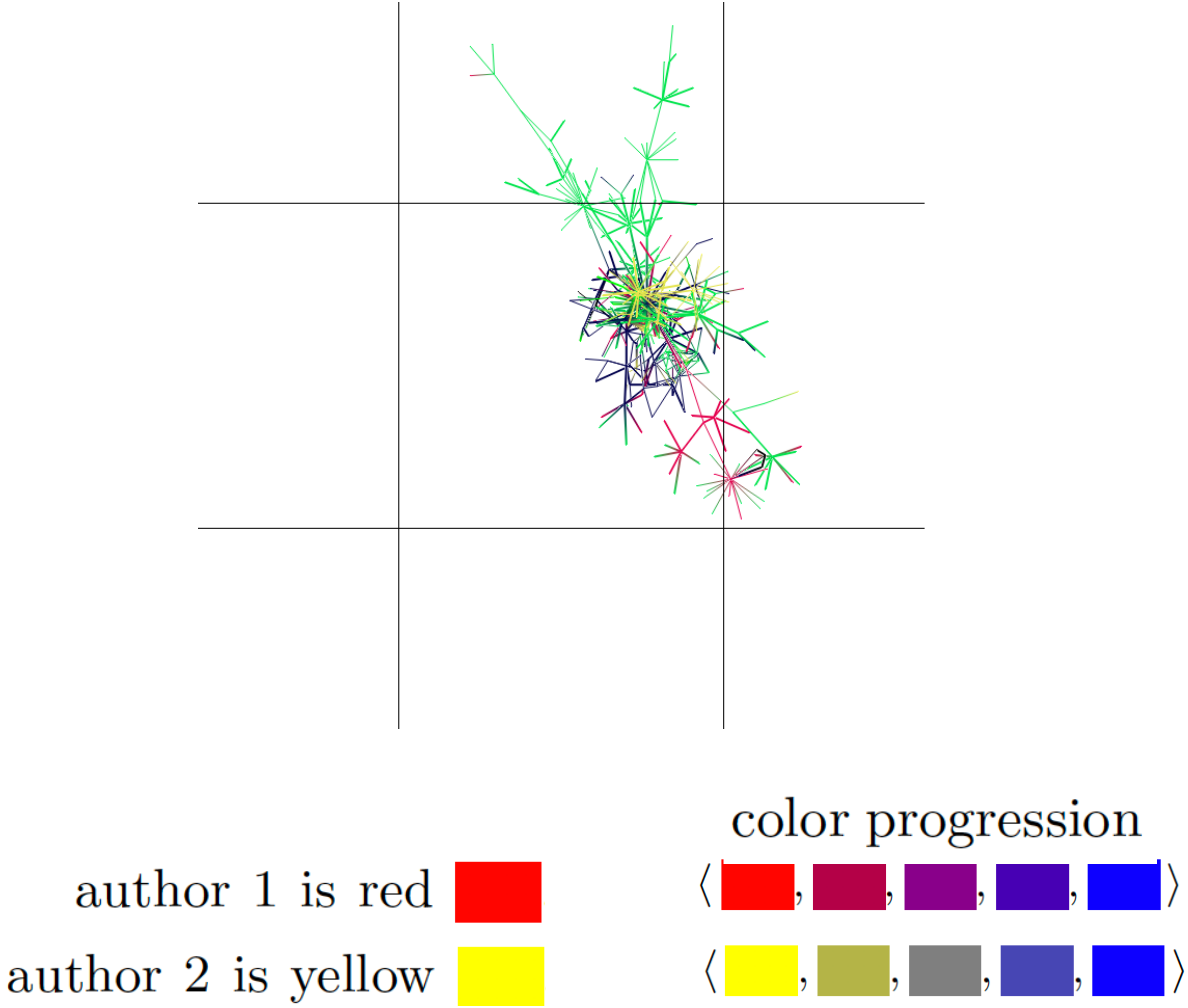
author 1 is red

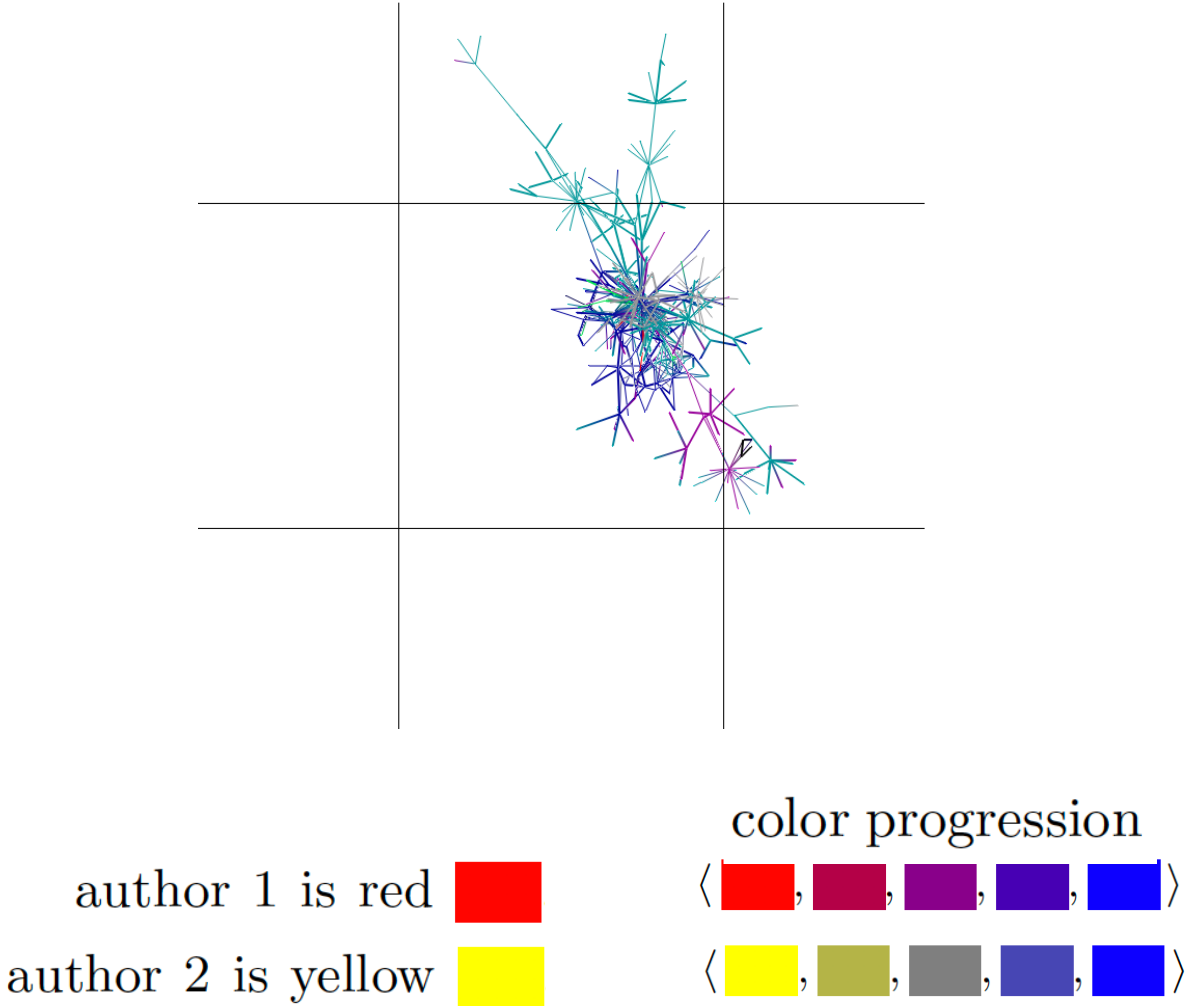
author 2 is yellow

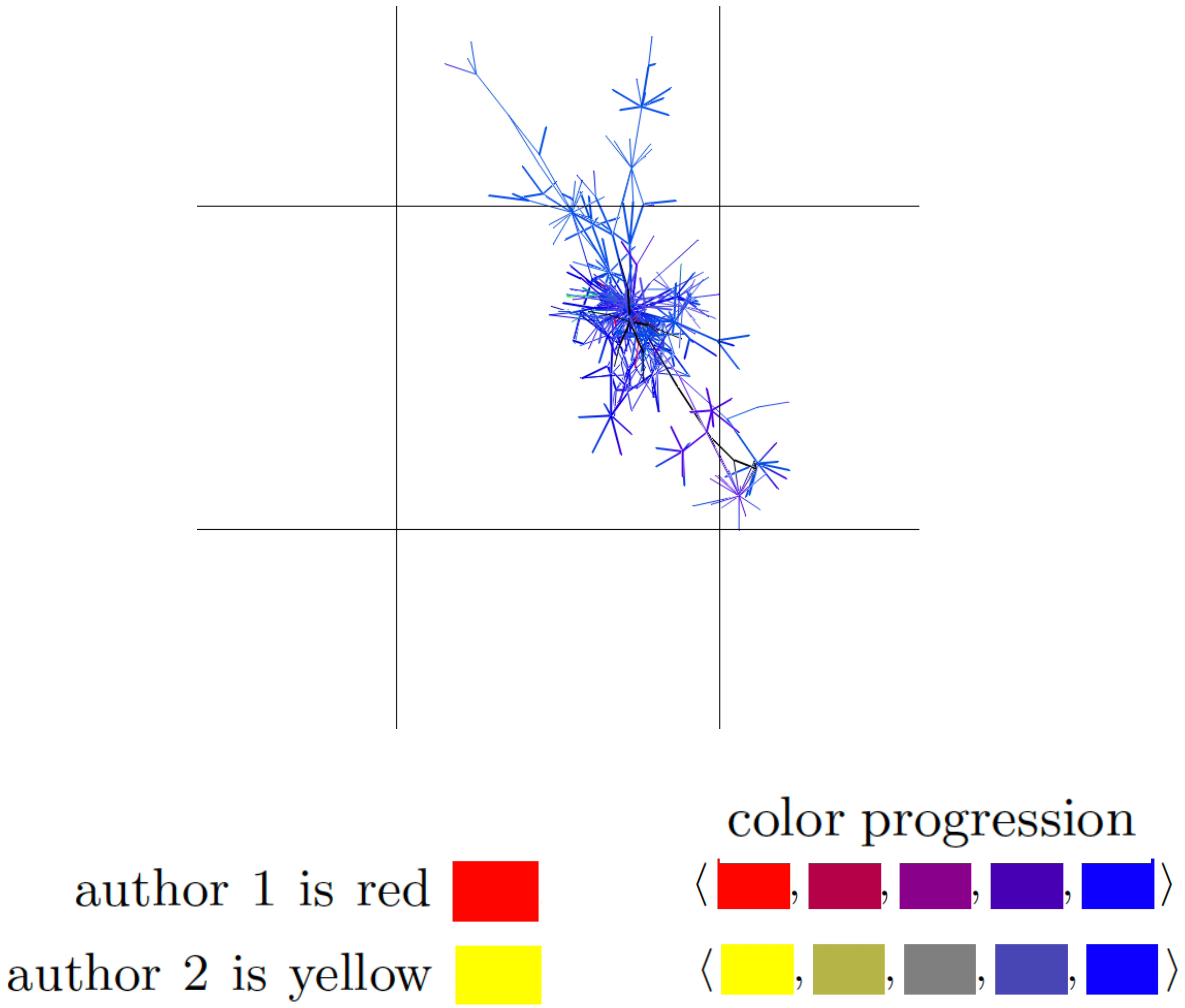
color progression

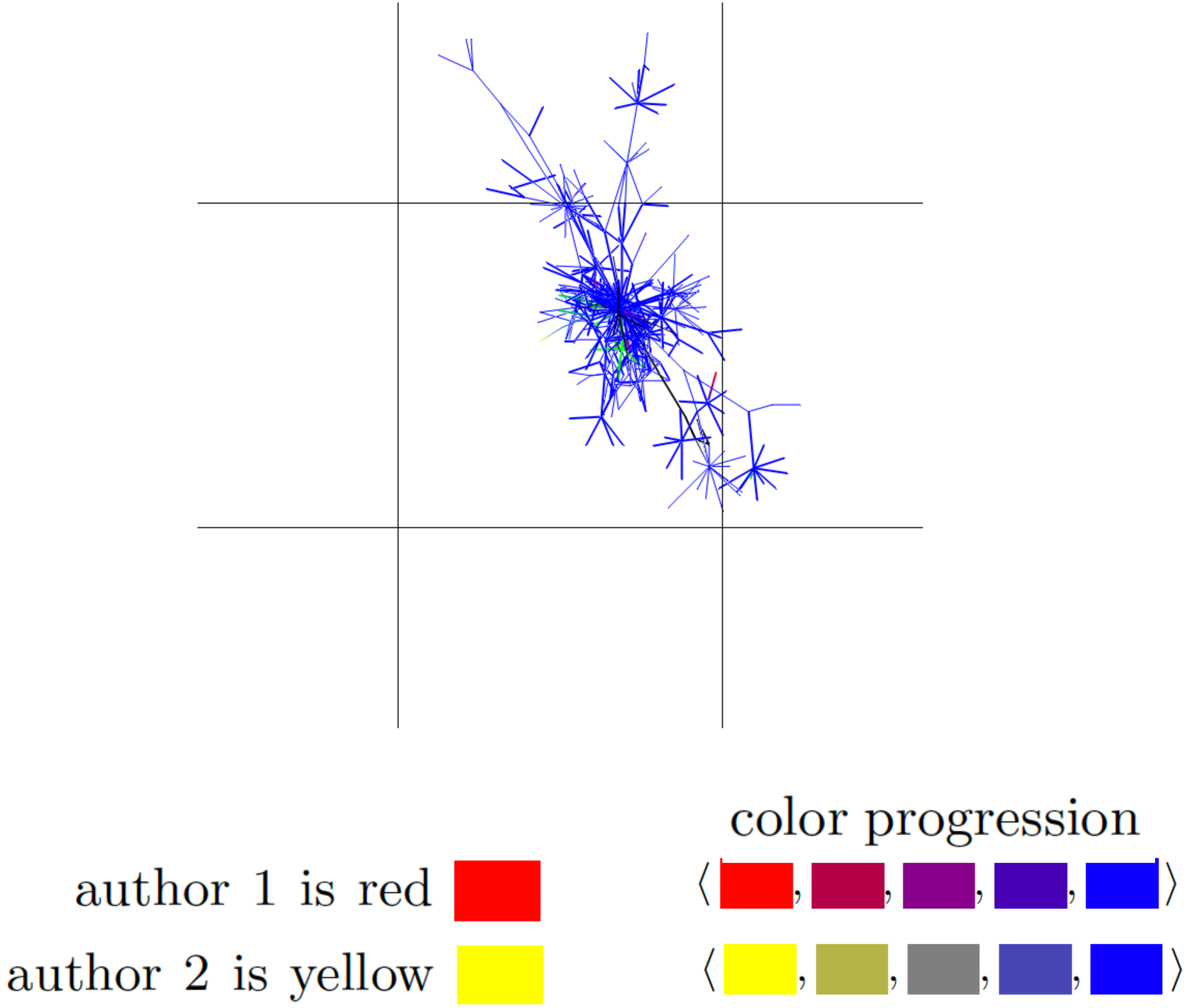
⟨ , , , , ⟩

⟨ , , , , ⟩


author 1 is red
author 2 is yellow
color progression

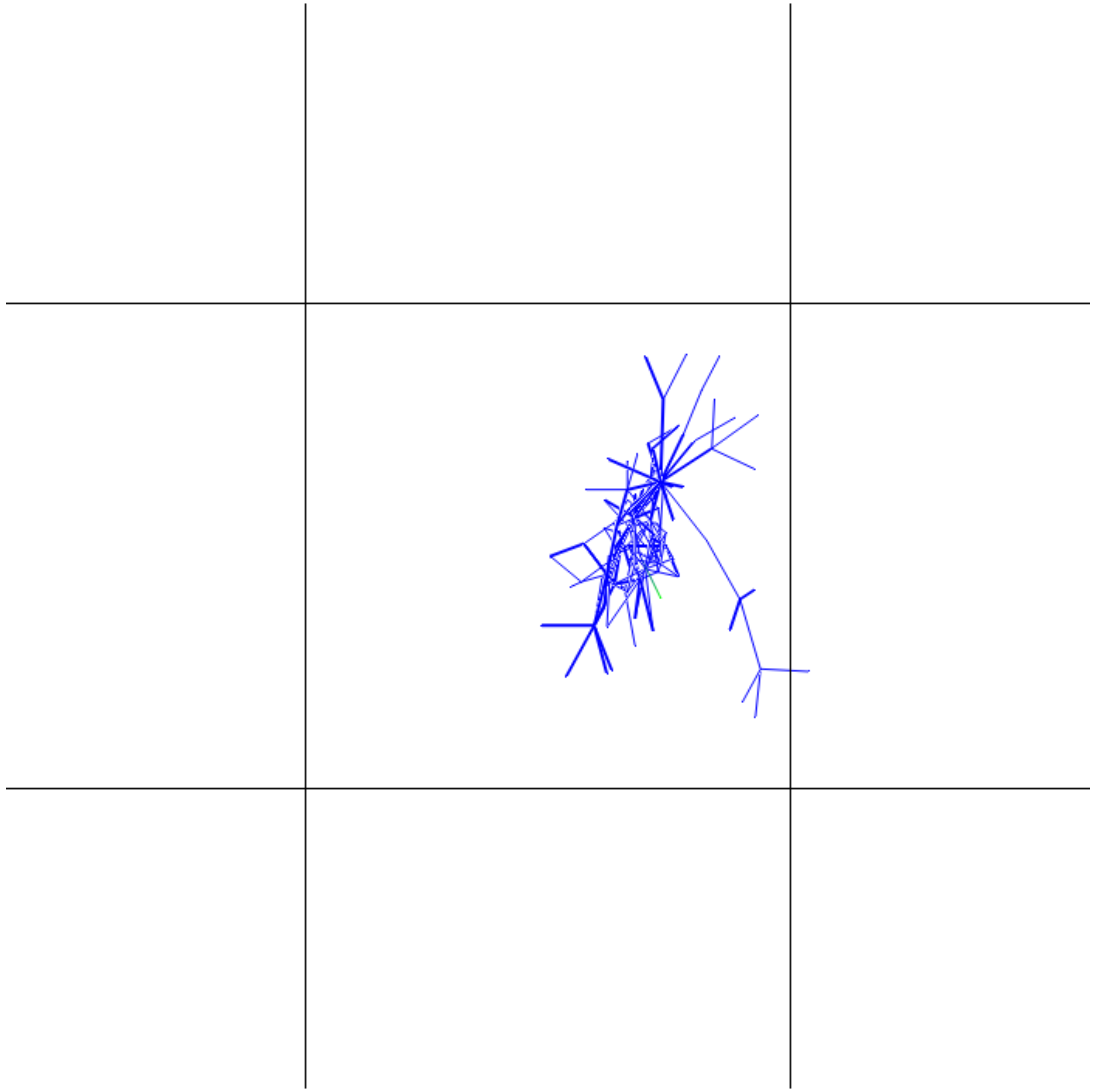

color progression
author 1 is red
author 2 is yellow


color progression
author 1 is red
author 2 is yellow



author 1 is red
author 2 is yellow
color progression

Bob broke the build!

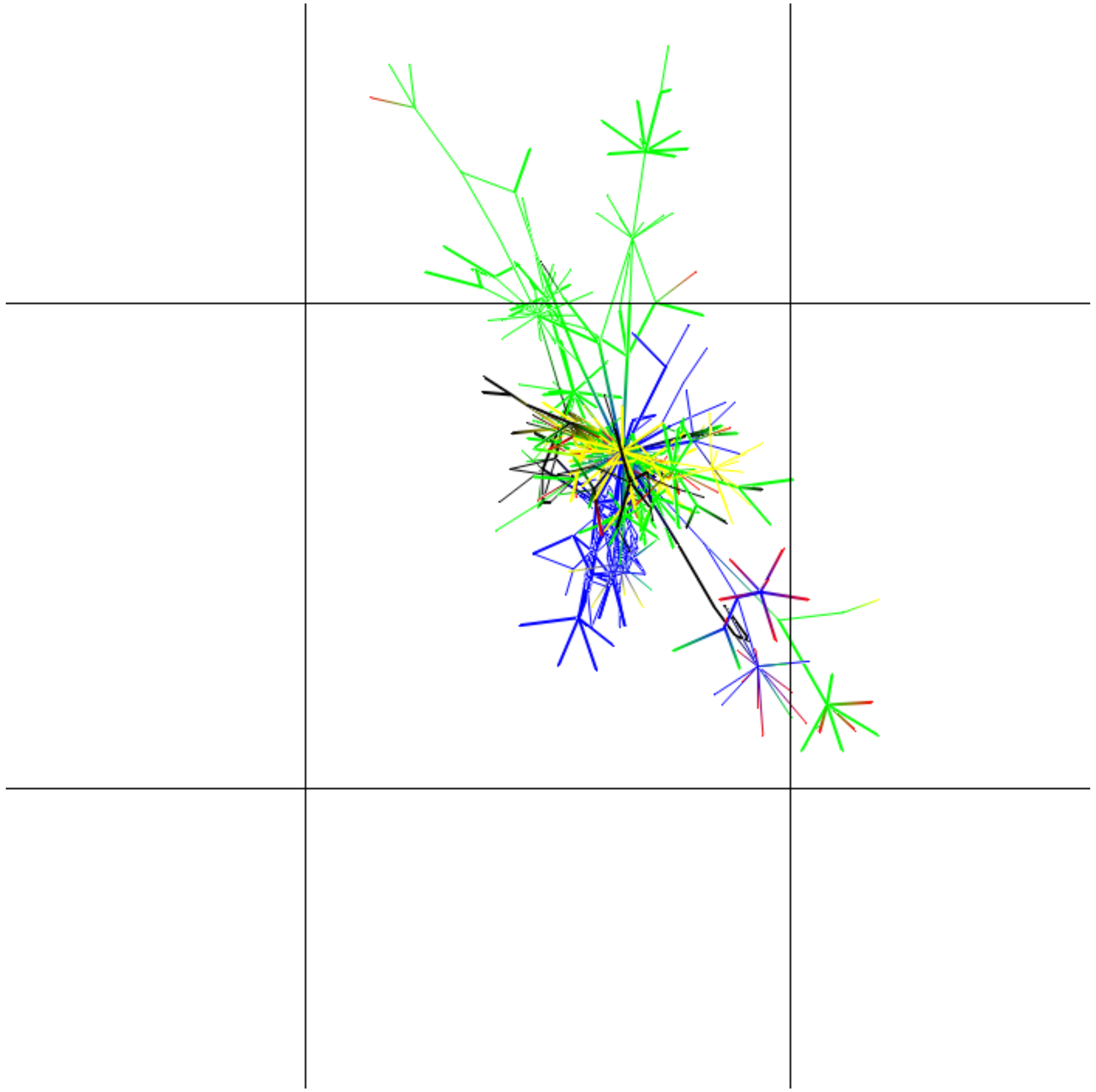


author 1 is red

author 2 is yellow

color progression

Bob broke the build!

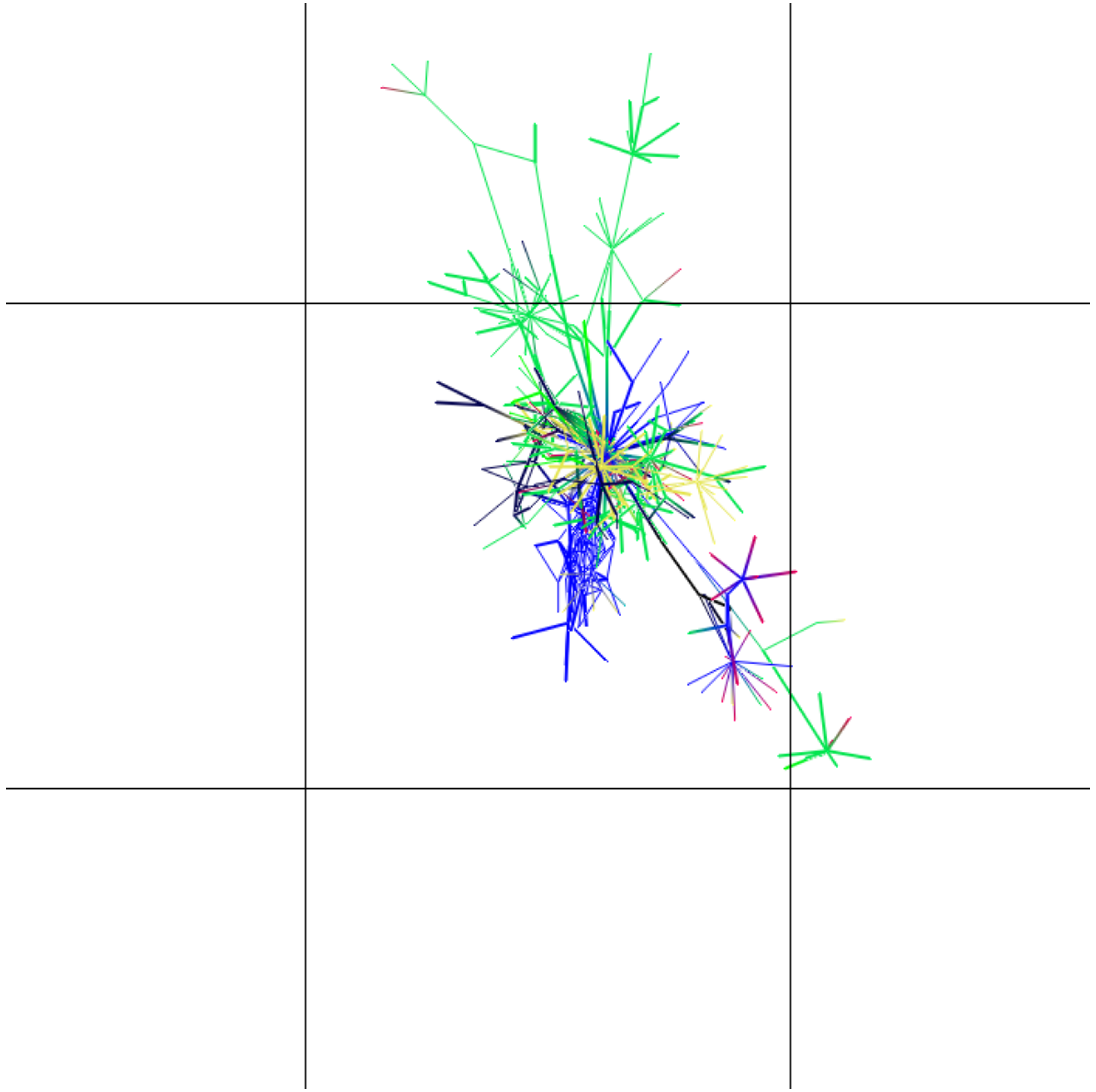


author 1 is red

author 2 is yellow

color progression

Bob broke the build!

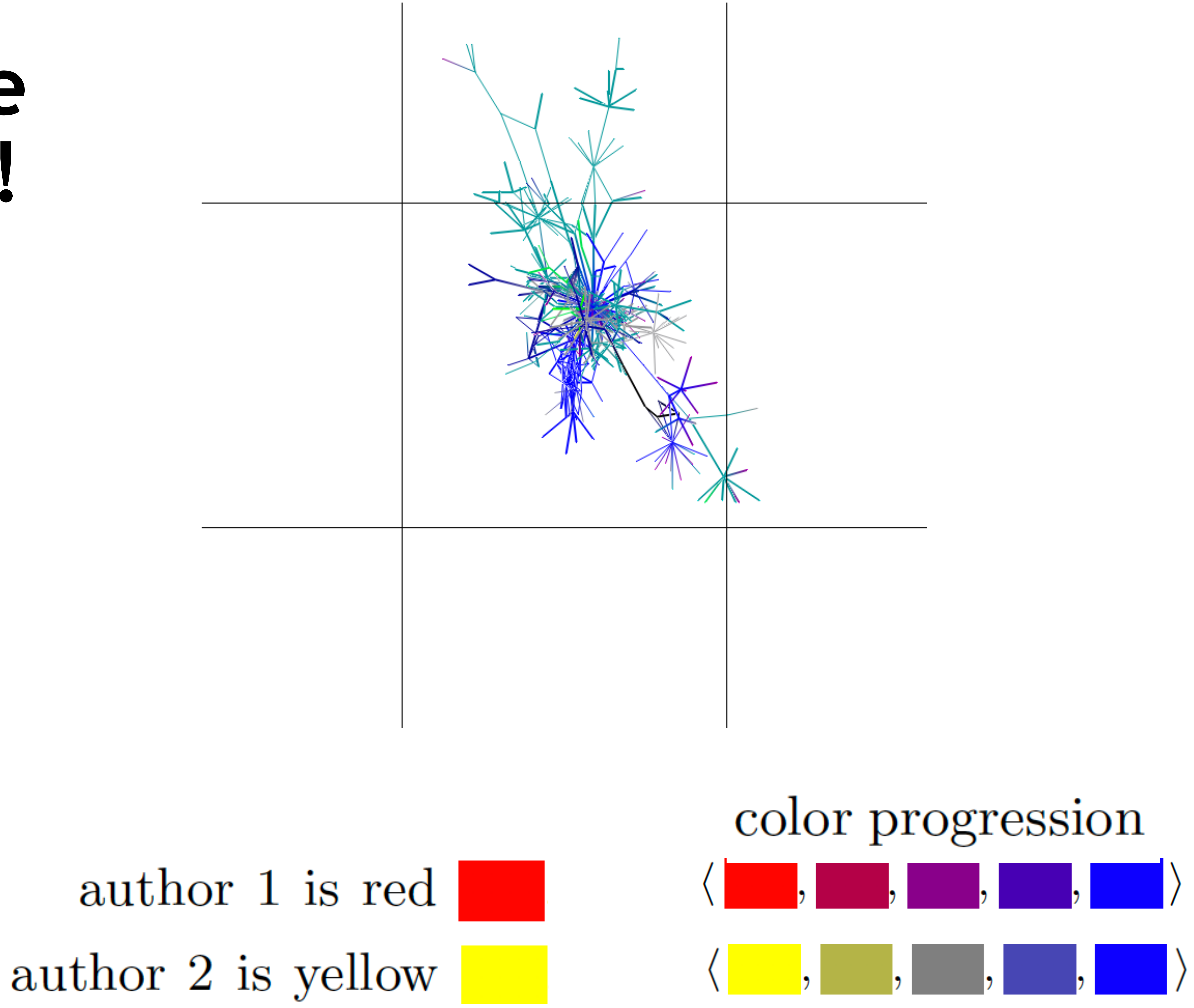


color progression

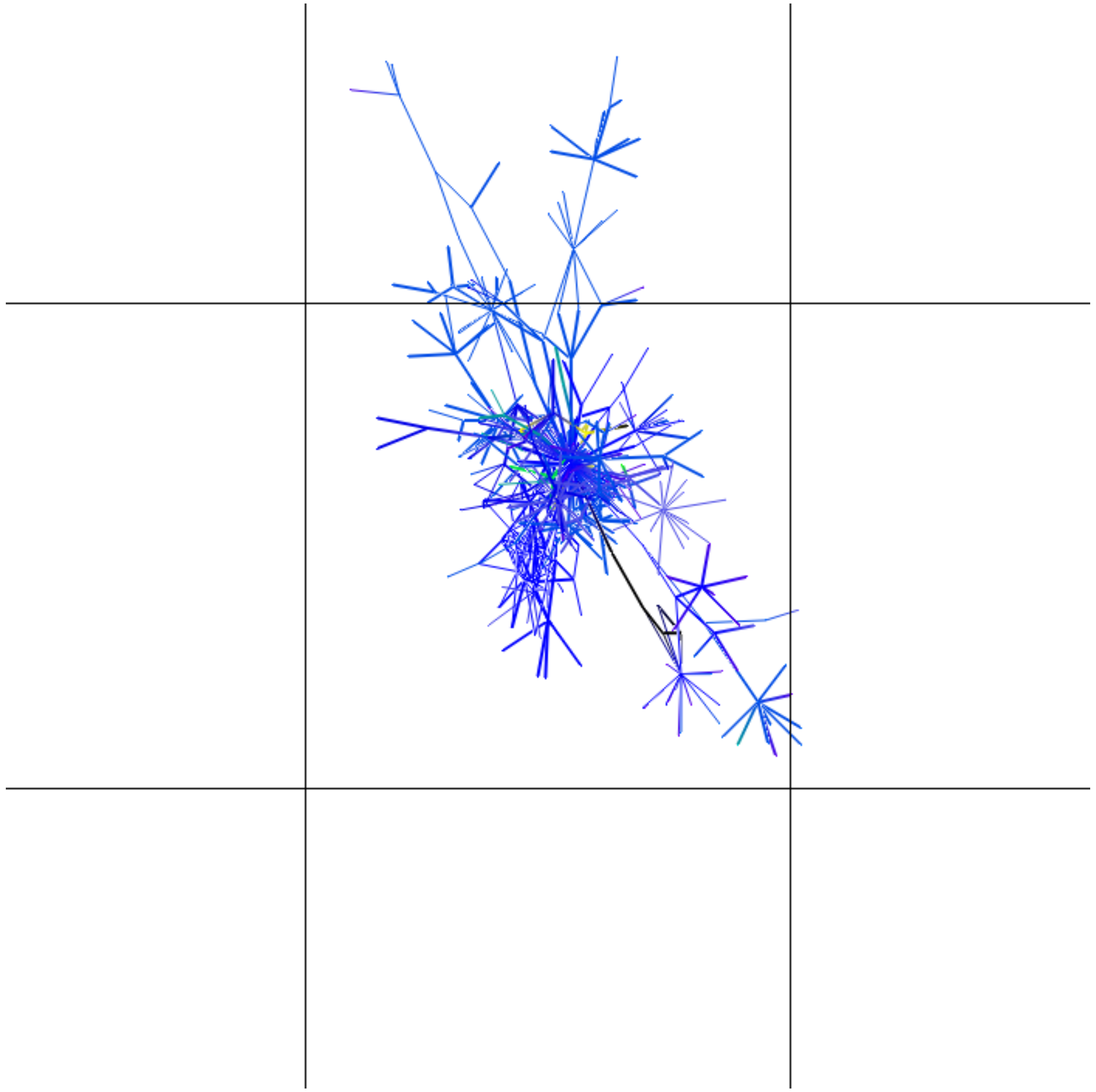
author 1 is red

author 2 is yellow

Bob broke the build!



Bob broke the build!

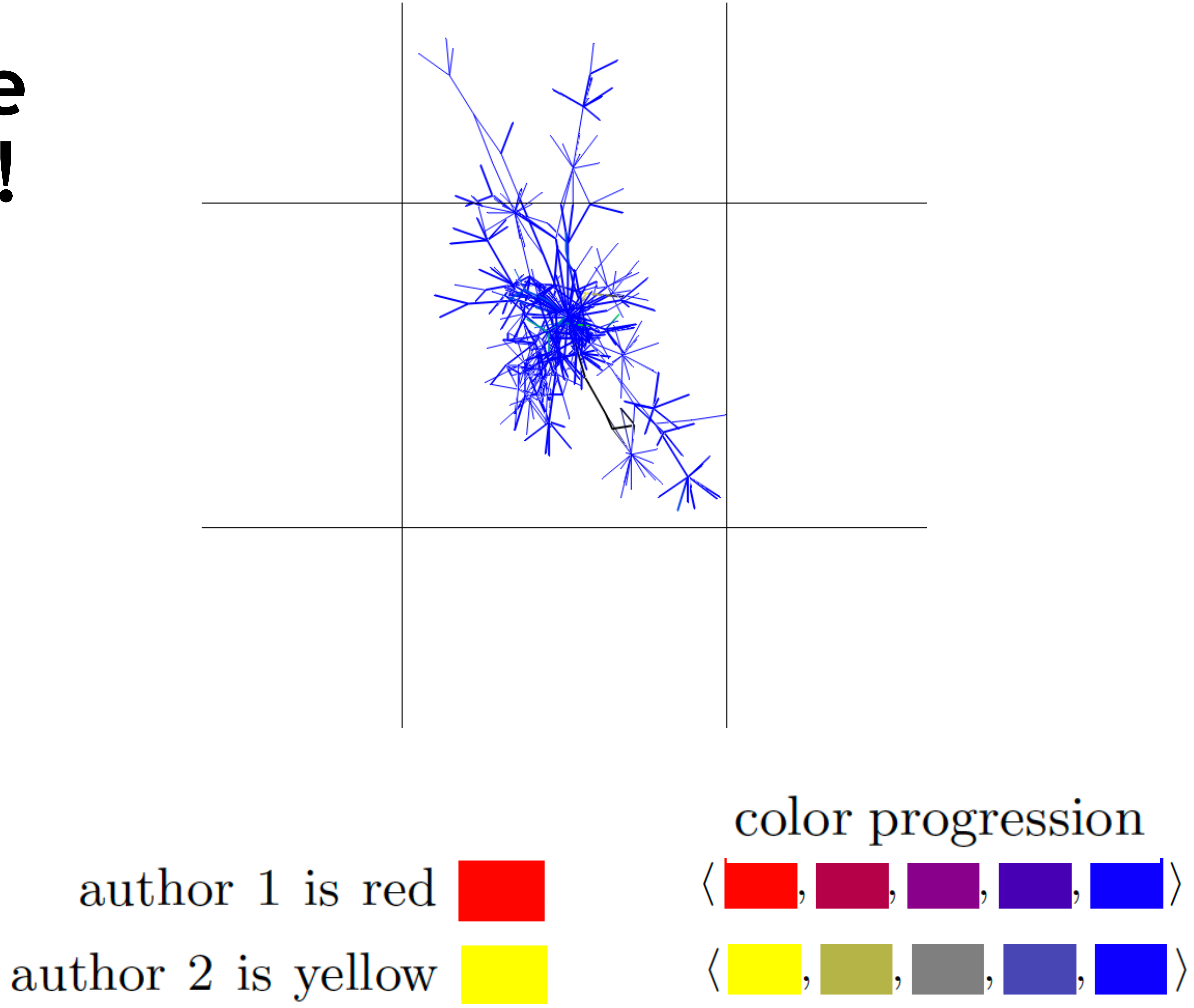


author 1 is red

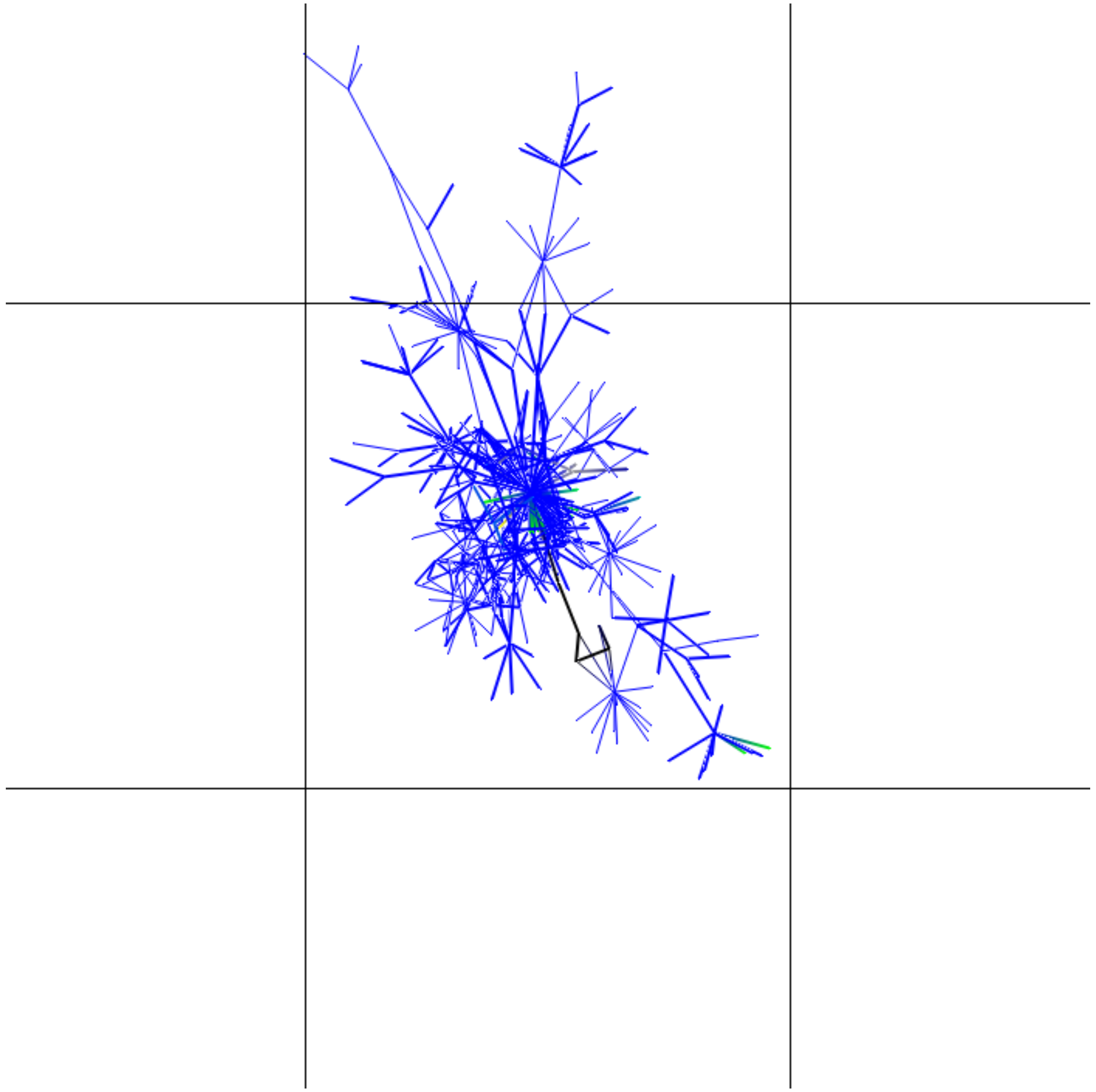
author 2 is yellow

color progression

Bob broke the build!



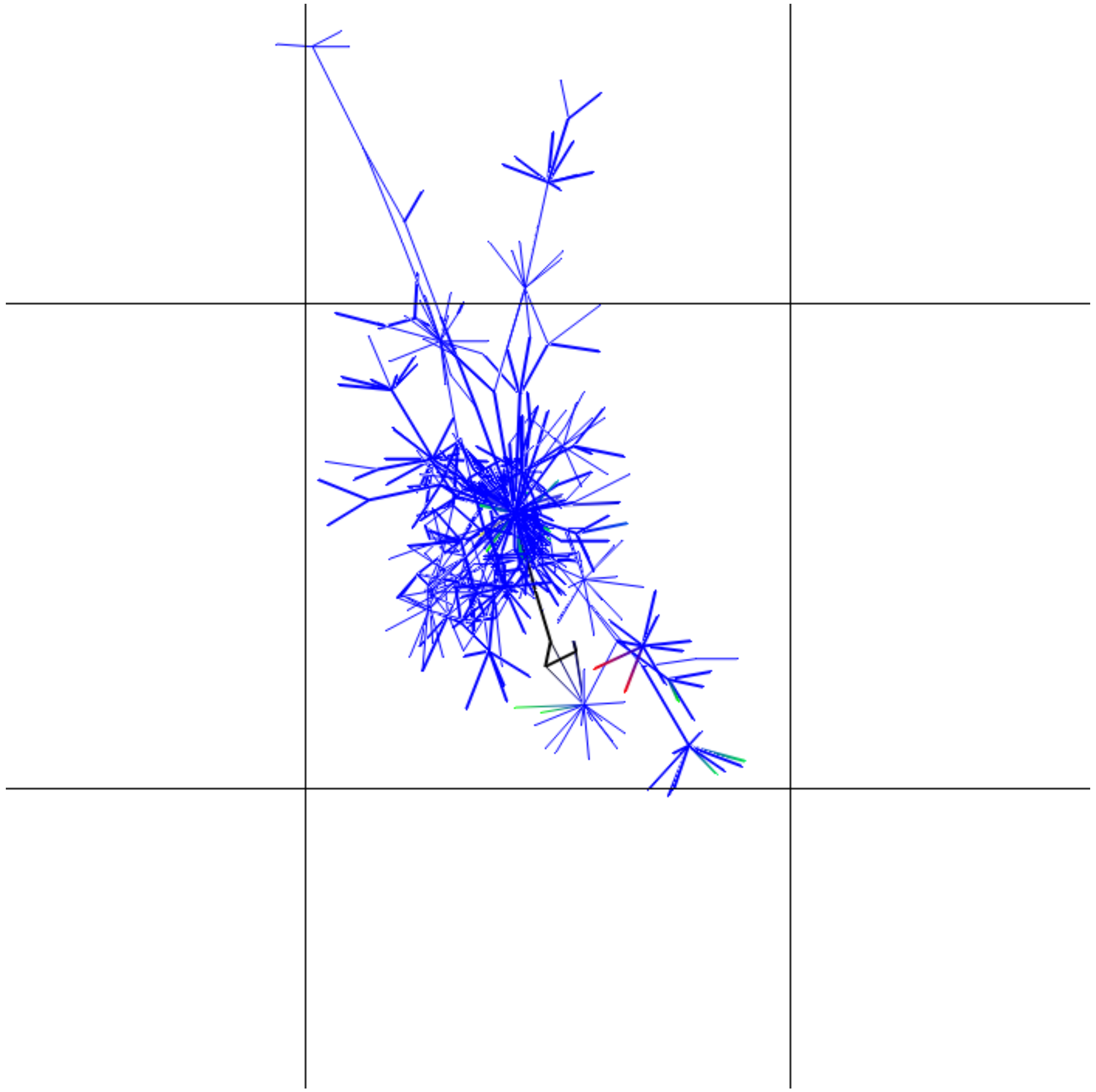
Bob broke the build!



author 1 is red
author 2 is yellow

color progression
⟨ , , , , ⟩
⟨ , , , , ⟩

Bob broke the build!

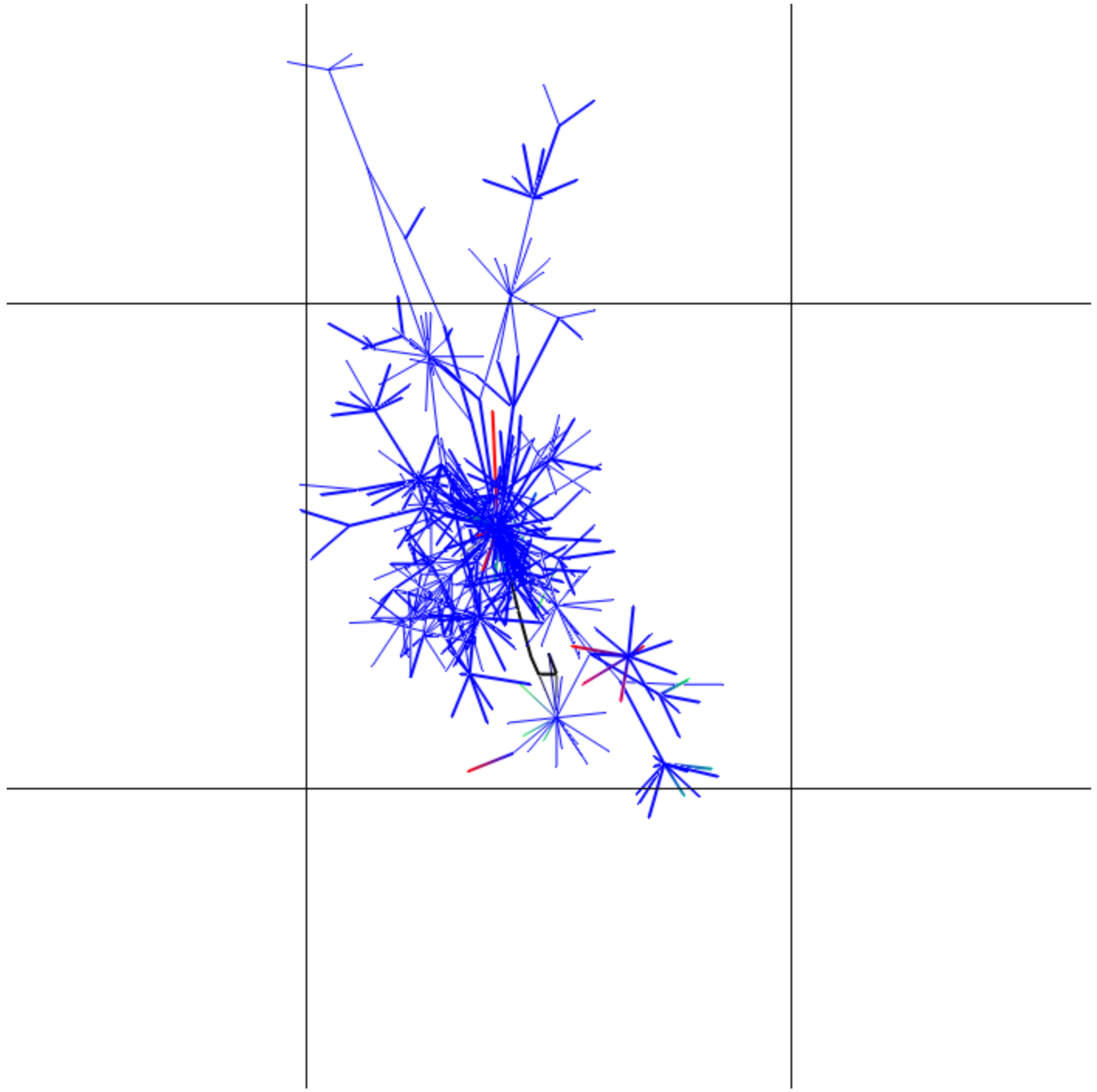


author 1 is red

author 2 is yellow

color progression

Bob broke the build!



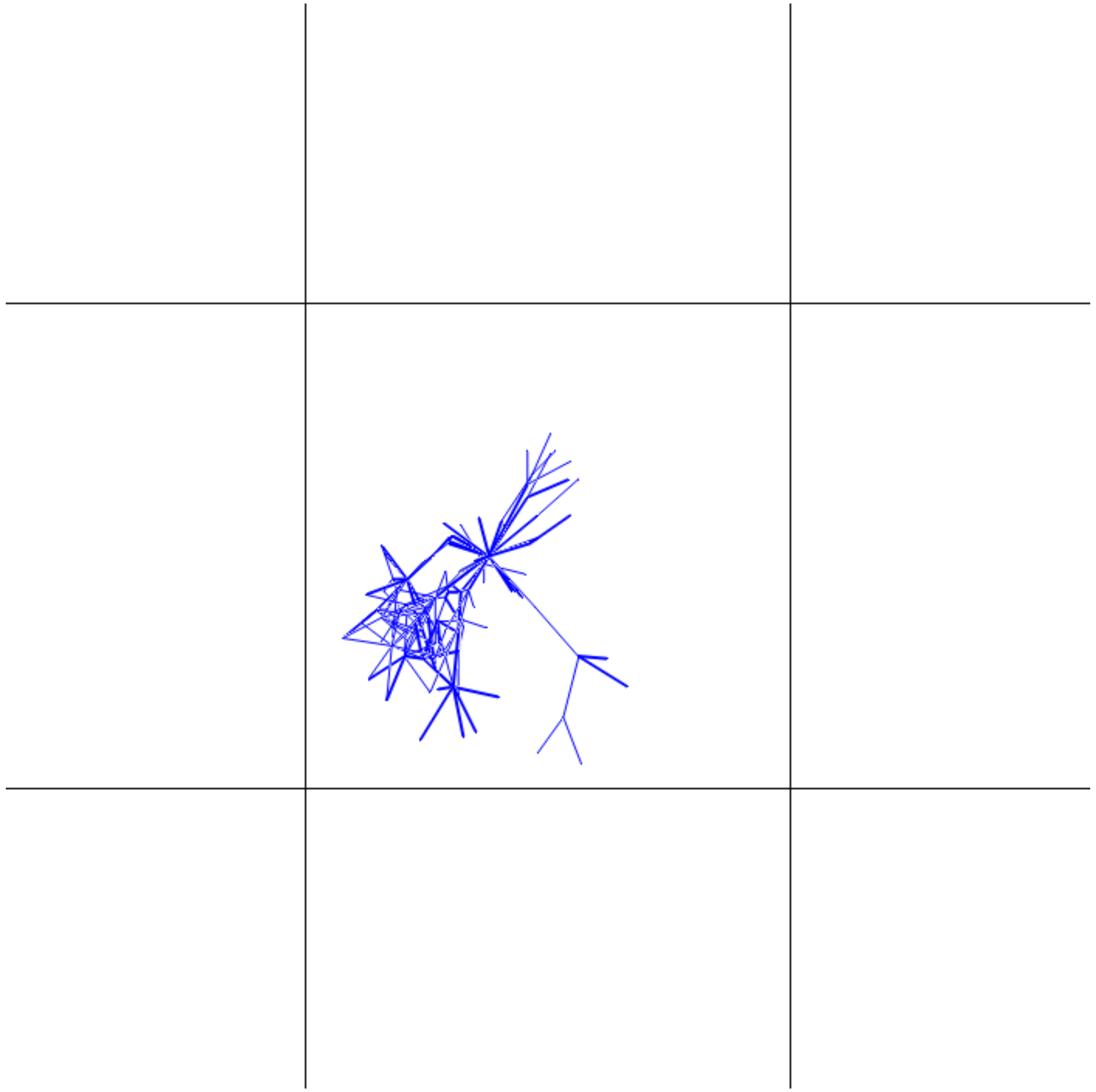
color progression

author 1 is red ⟨red, crimson, purple, indigo, blue⟩

author 2 is yellow ⟨yellow, olive, gray, slate blue, blue⟩

Bob broke the build!

Again!



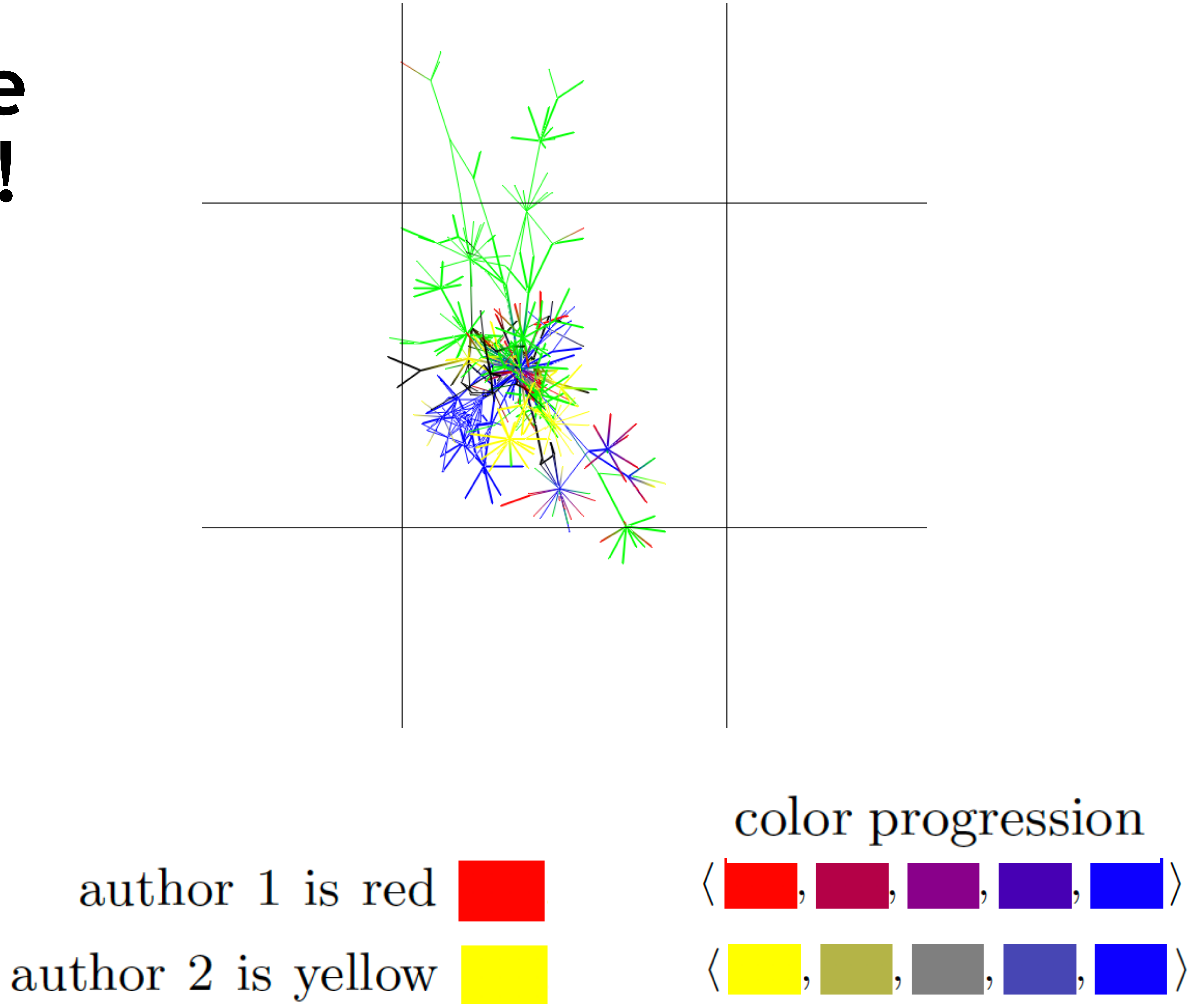
author 1 is red

author 2 is yellow

color progression

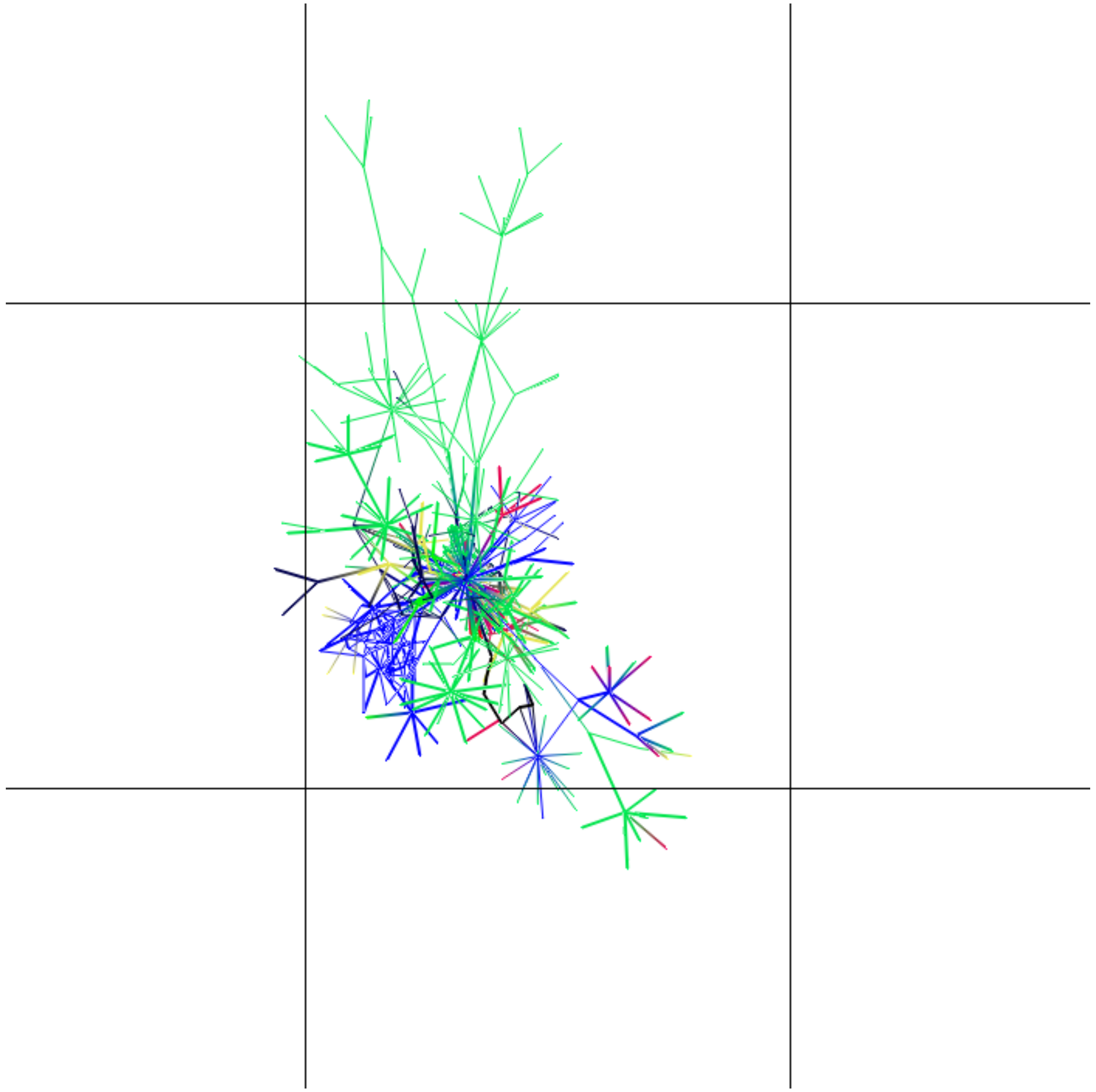
Bob broke the build!

Again!



Bob broke the build!

Again!



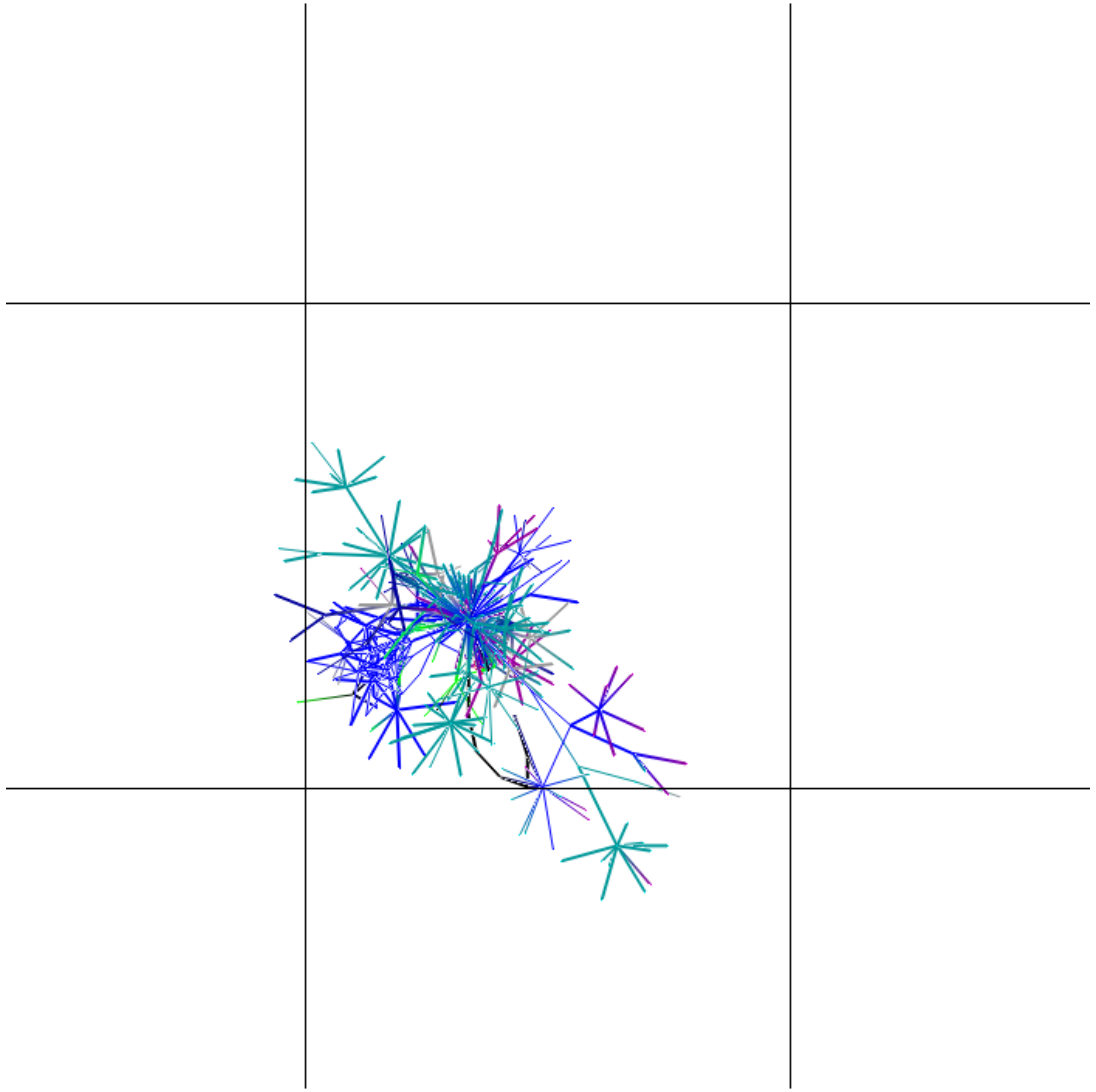
author 1 is red

author 2 is yellow

color progression

Bob broke the build!

Again!



10k LOC removed!

author 1 is red

author 2 is yellow

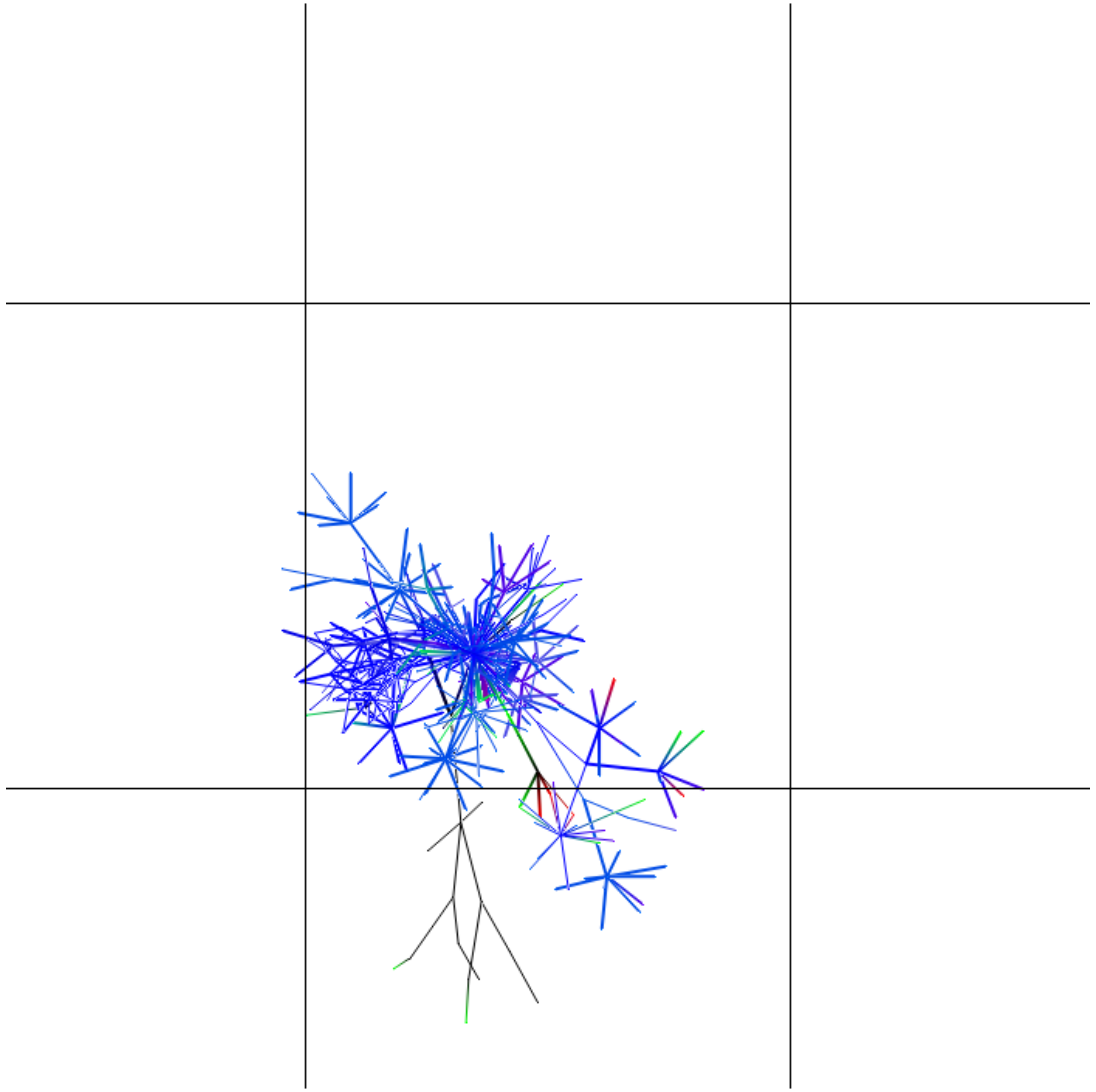
color progression

⟨ , , , , ⟩

⟨ , , , , ⟩

Bob broke the build!

Again!



10k LOC removed!

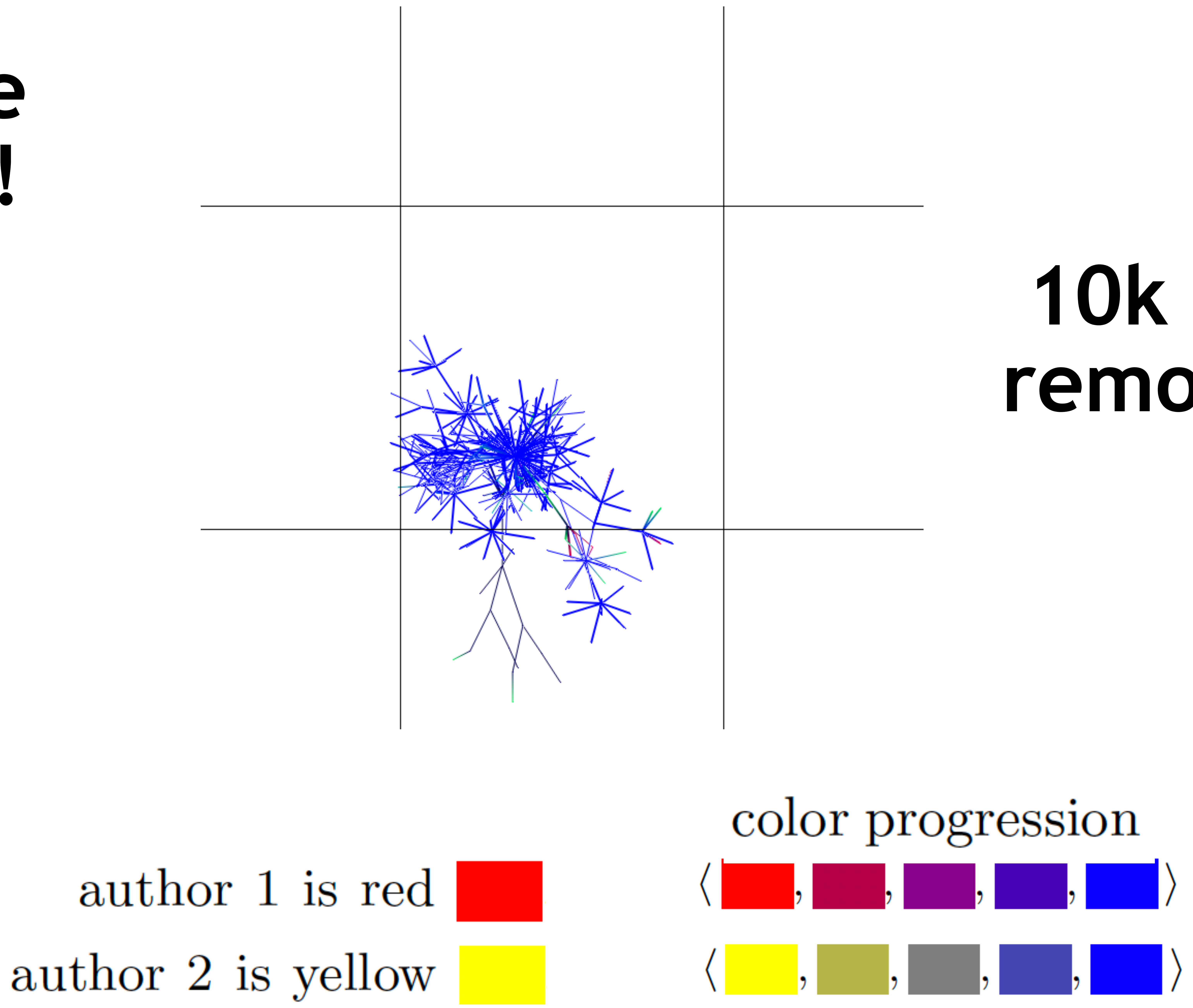
color progression

author 1 is red

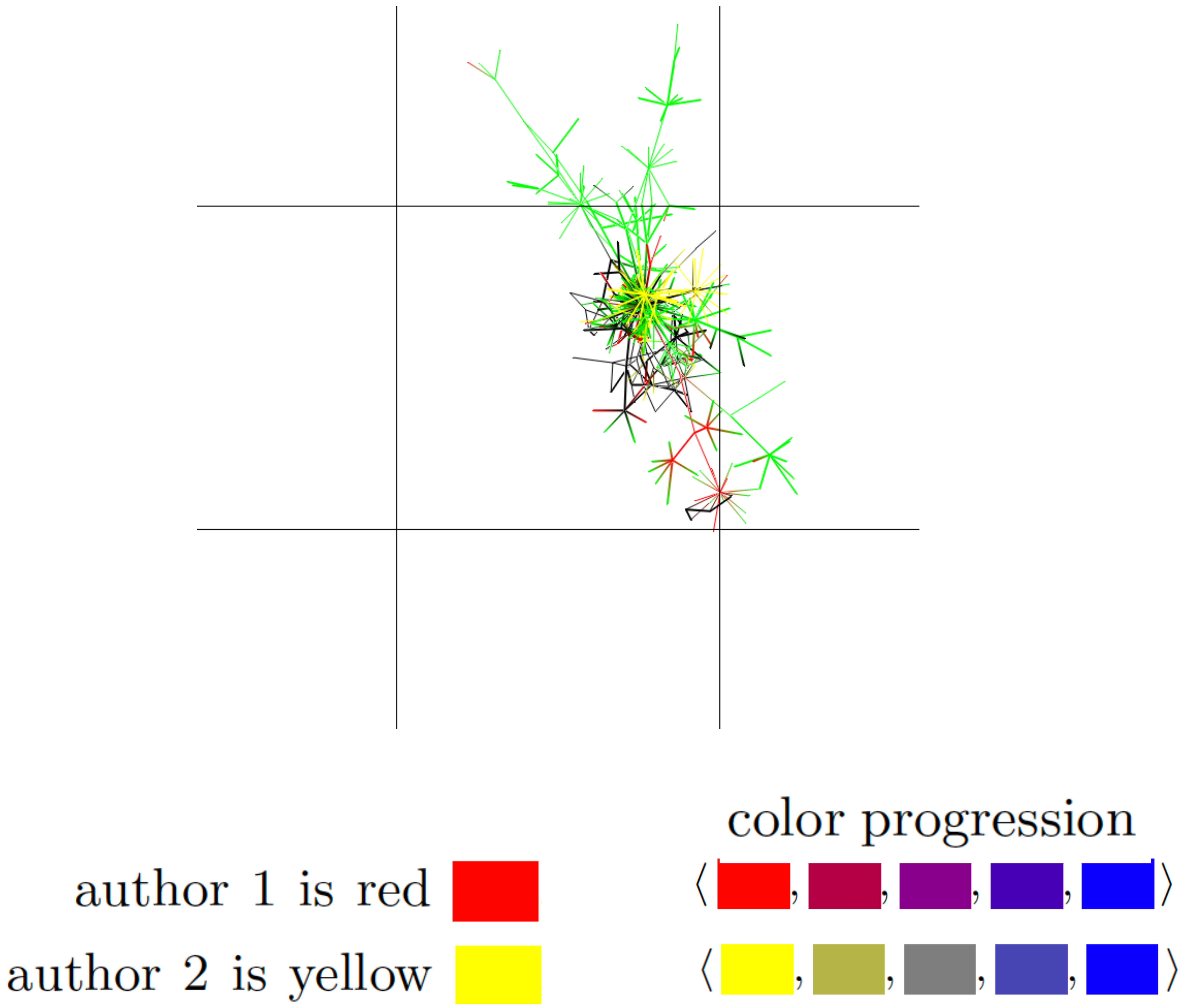
author 2 is yellow

Bob broke the build!

Again!

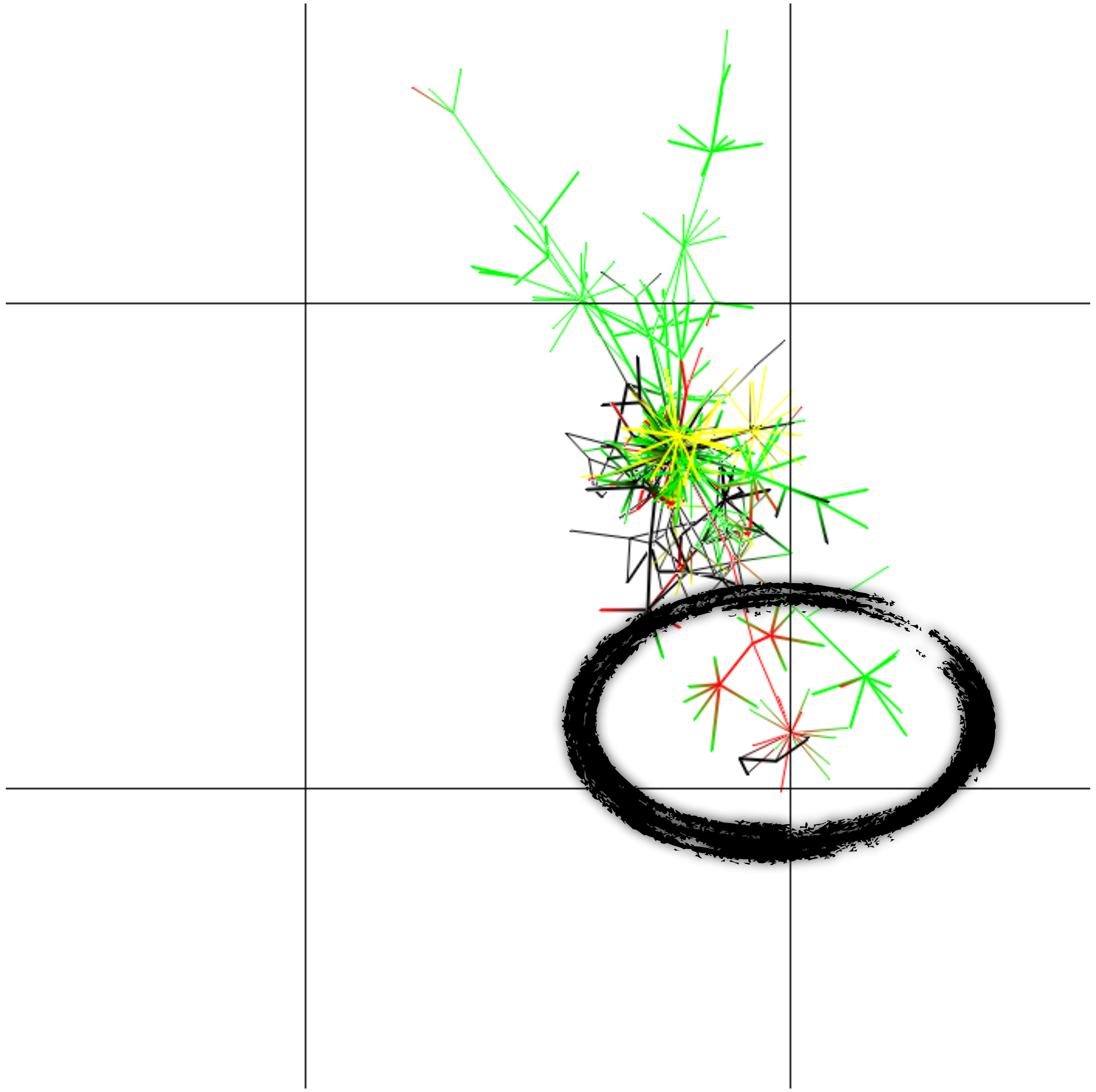


10k LOC removed!



author 1 is red
author 2 is yellow
color progression

Author 1 is a programmer

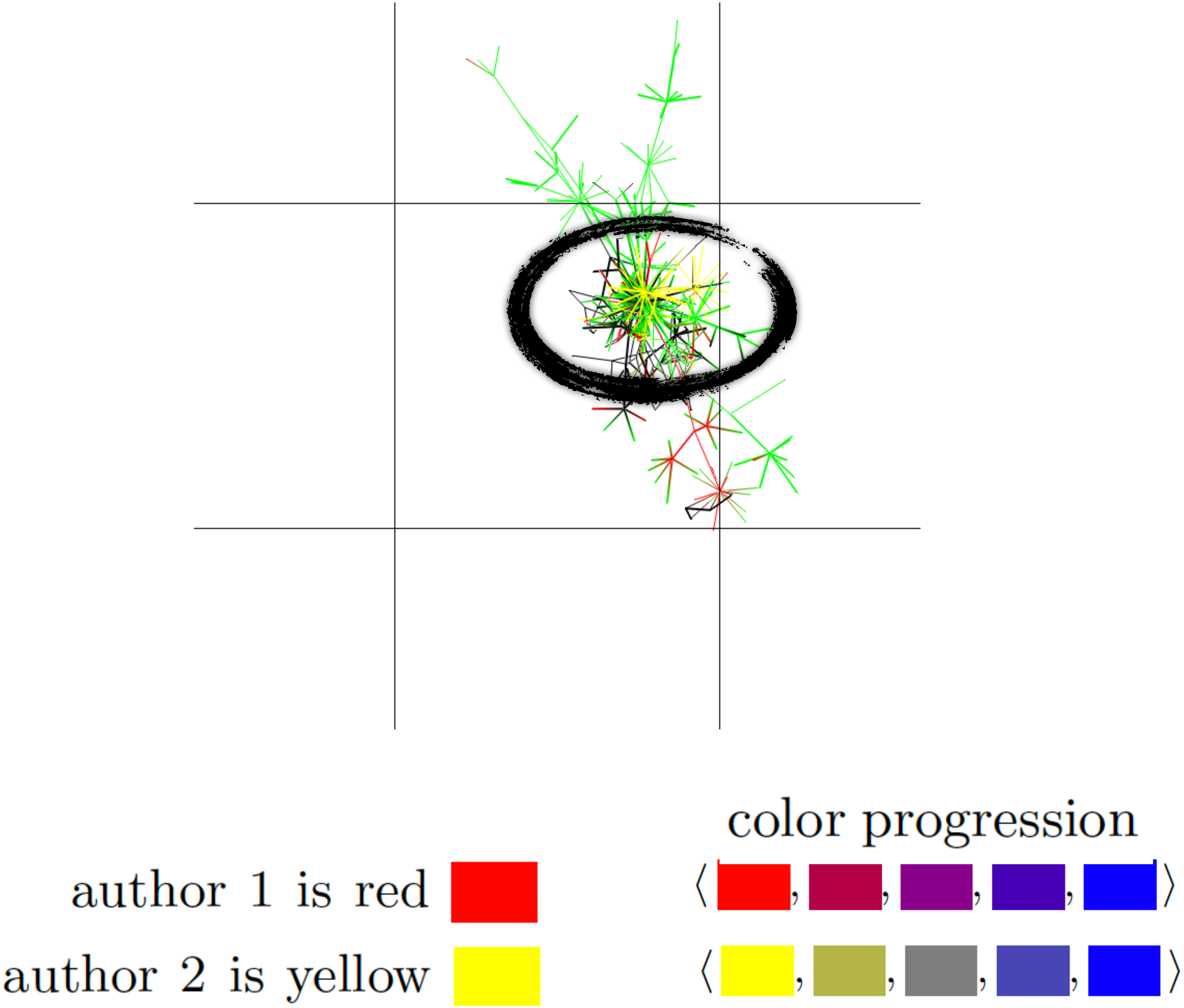


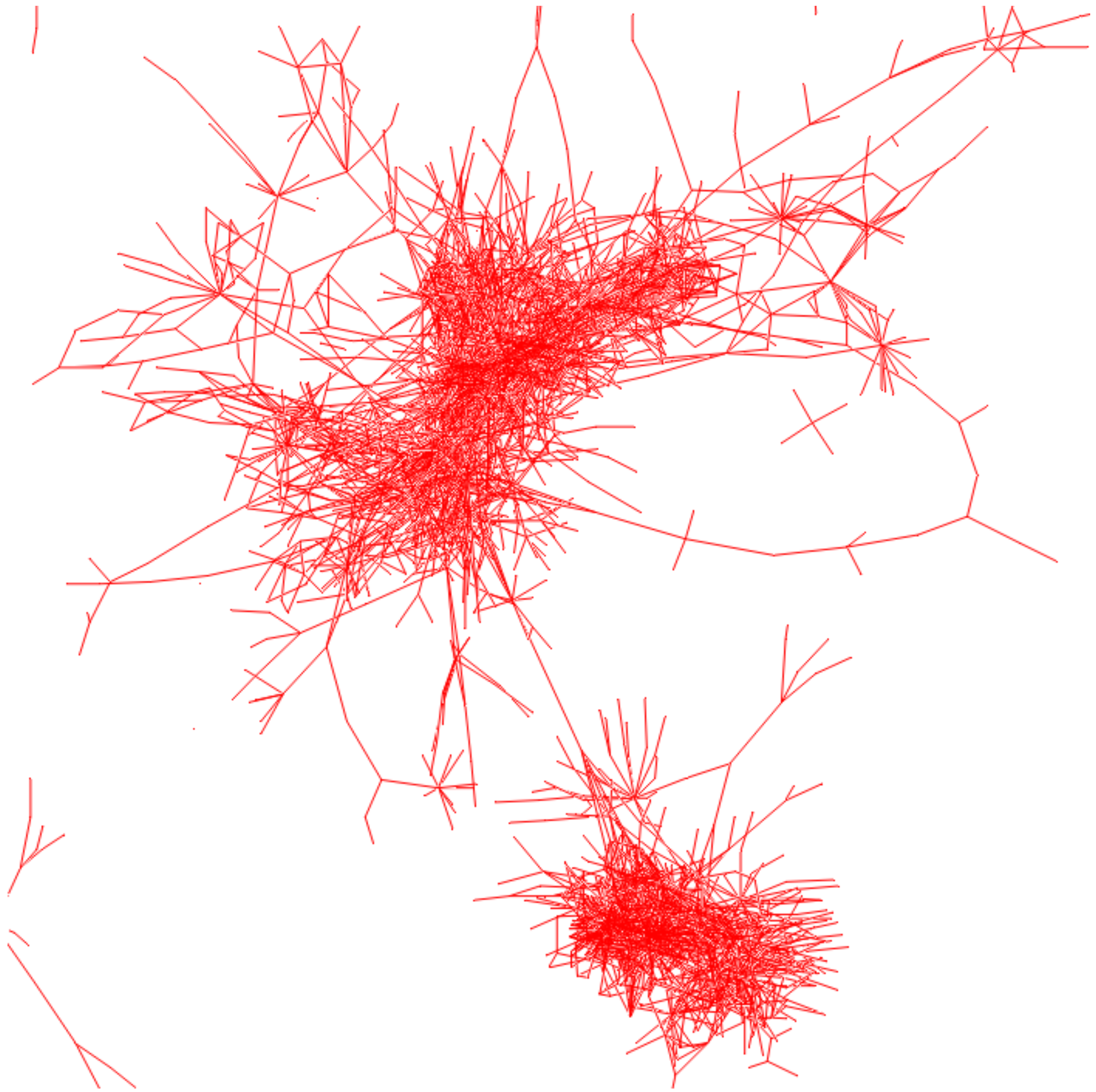
author 1 is red

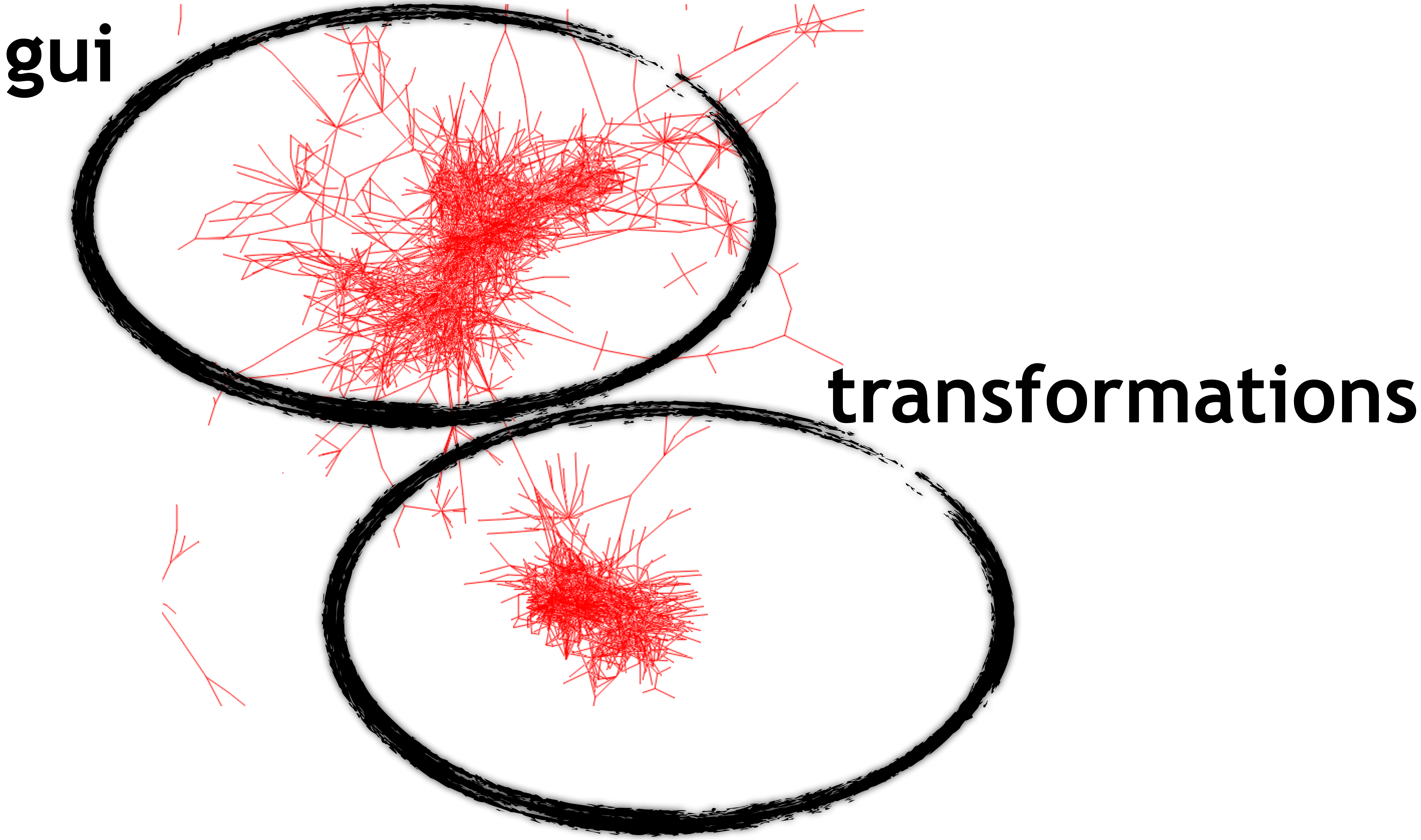
author 2 is yellow

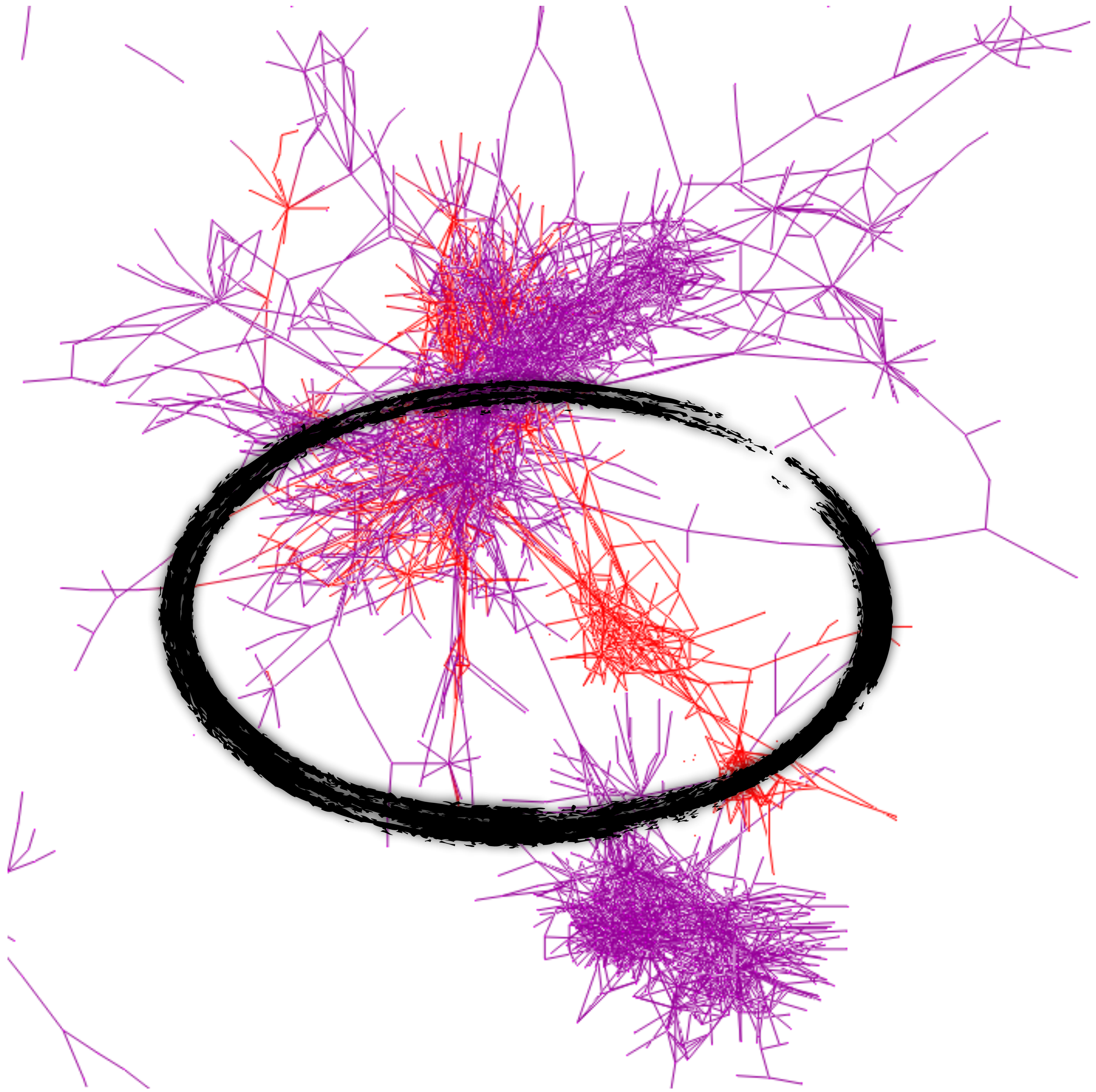
color progression

Author 2 is an architect





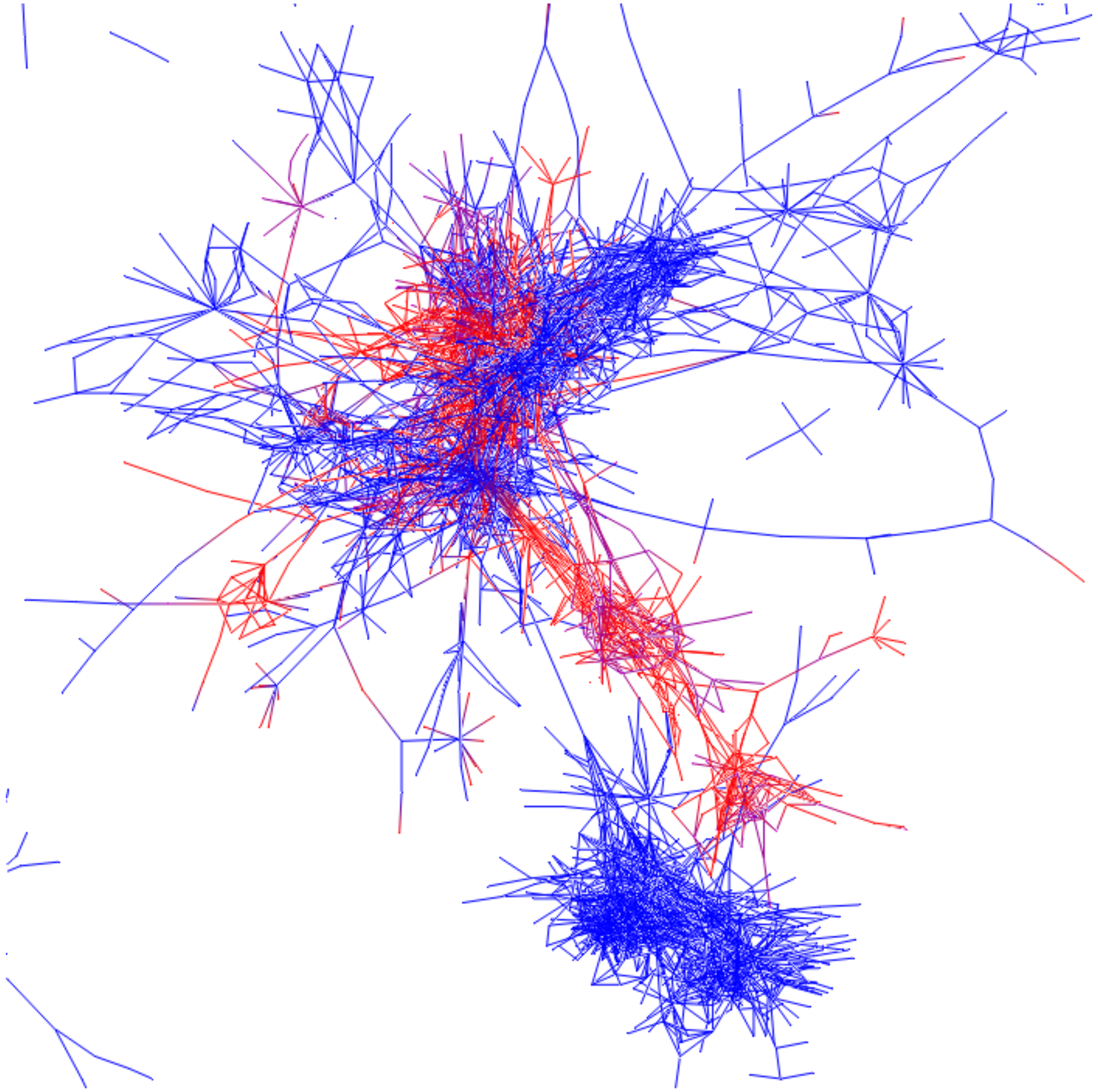

gui
transformations

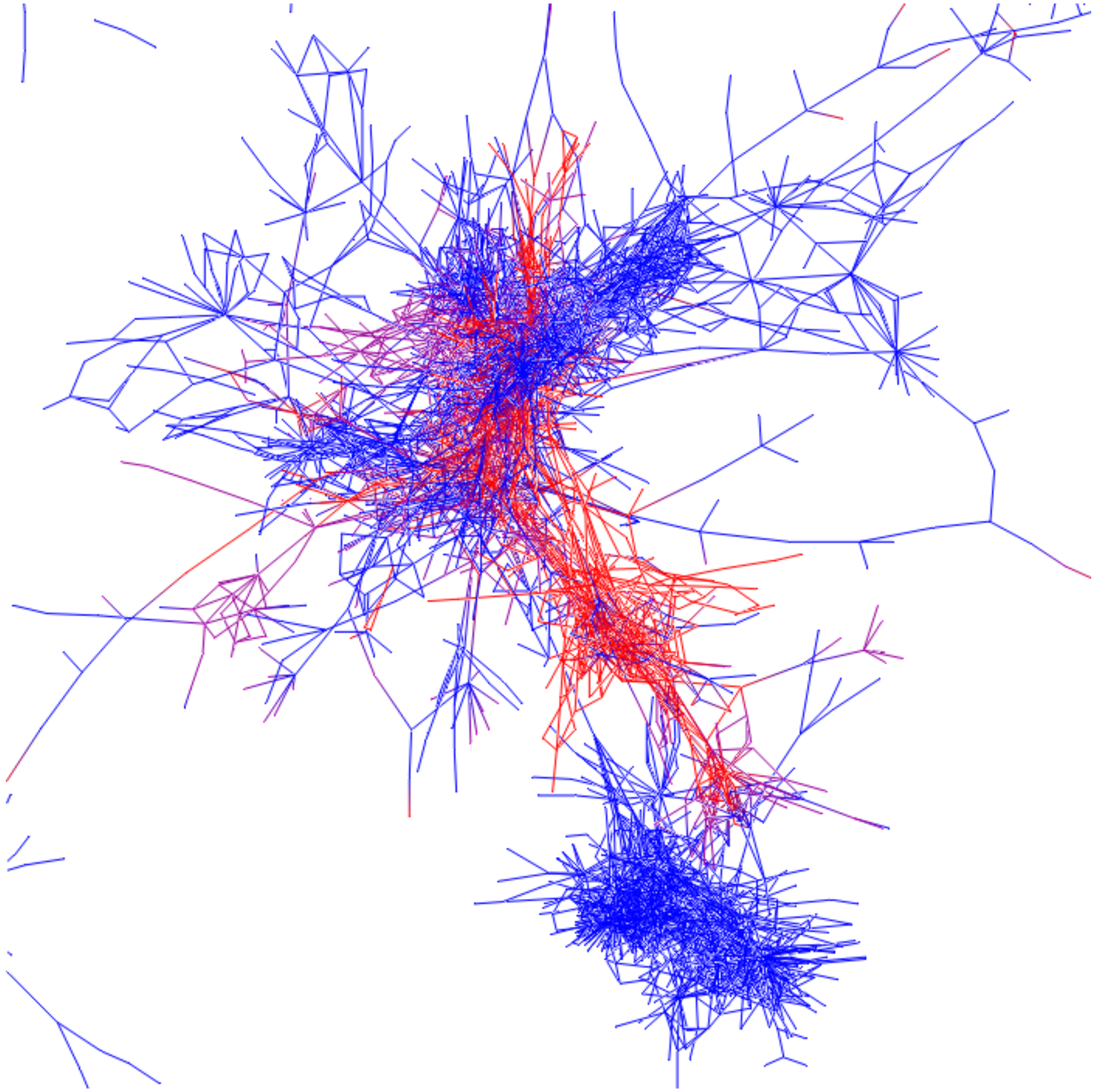


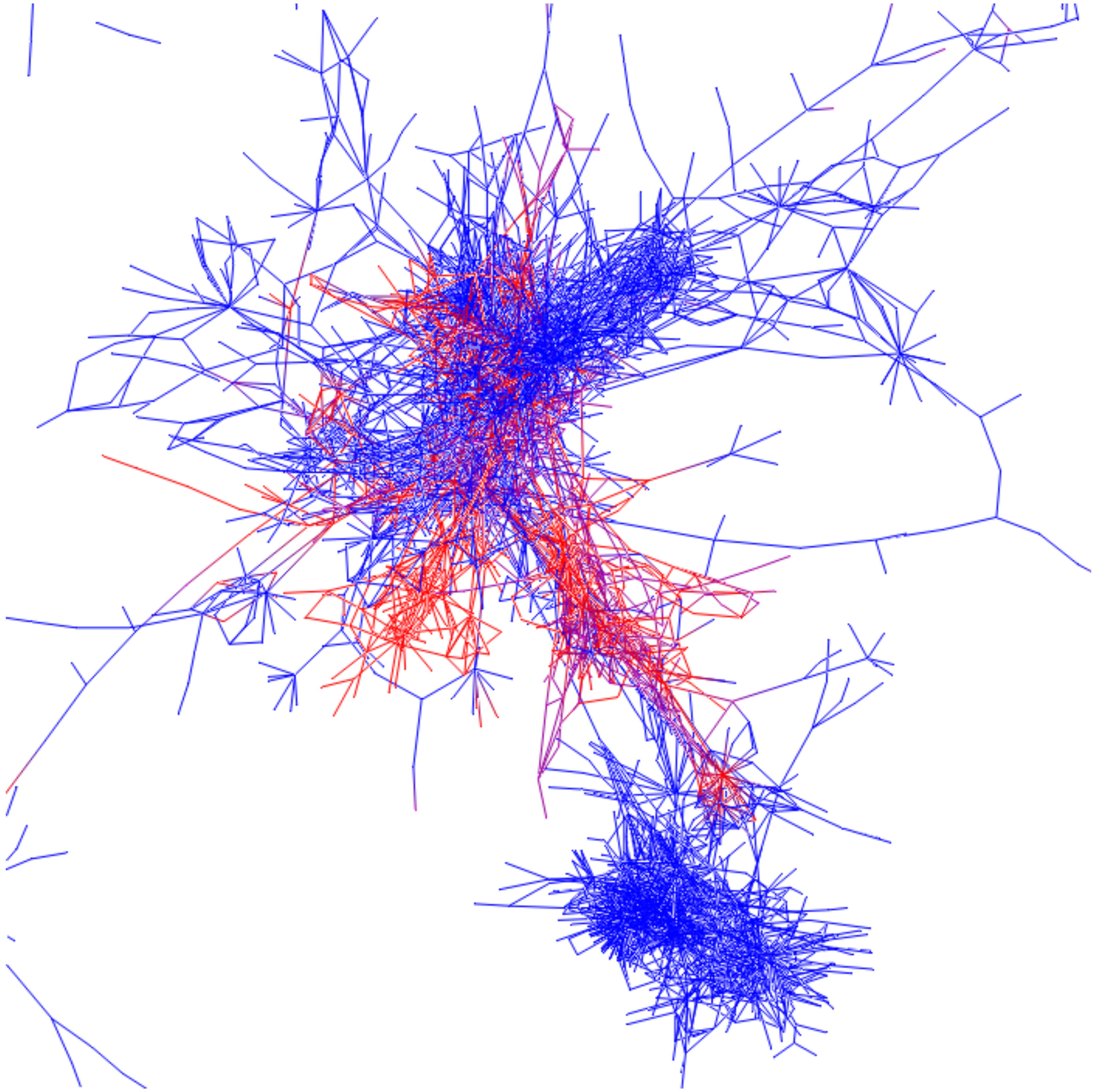
june 2002

—

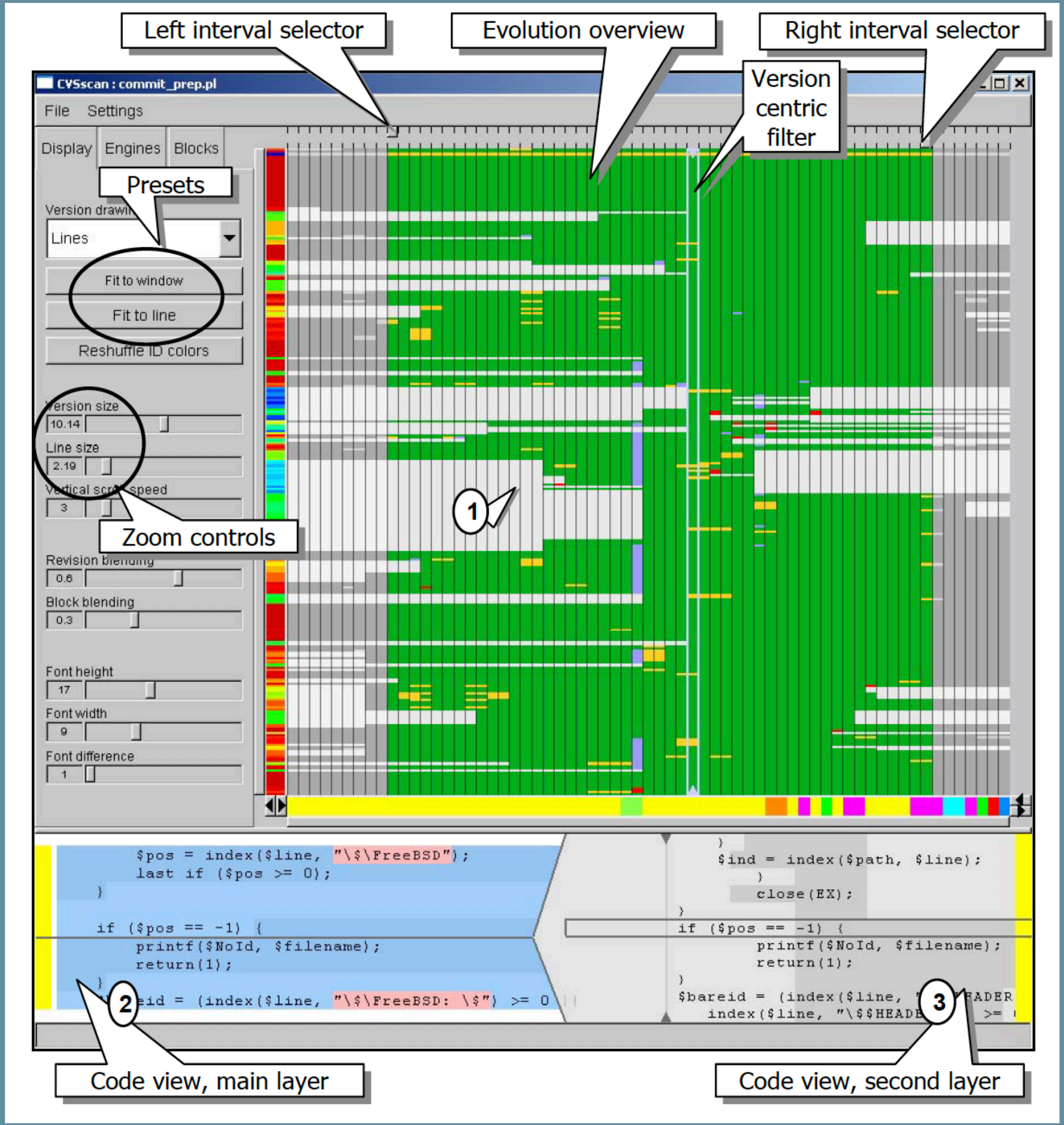
mediation point introduced



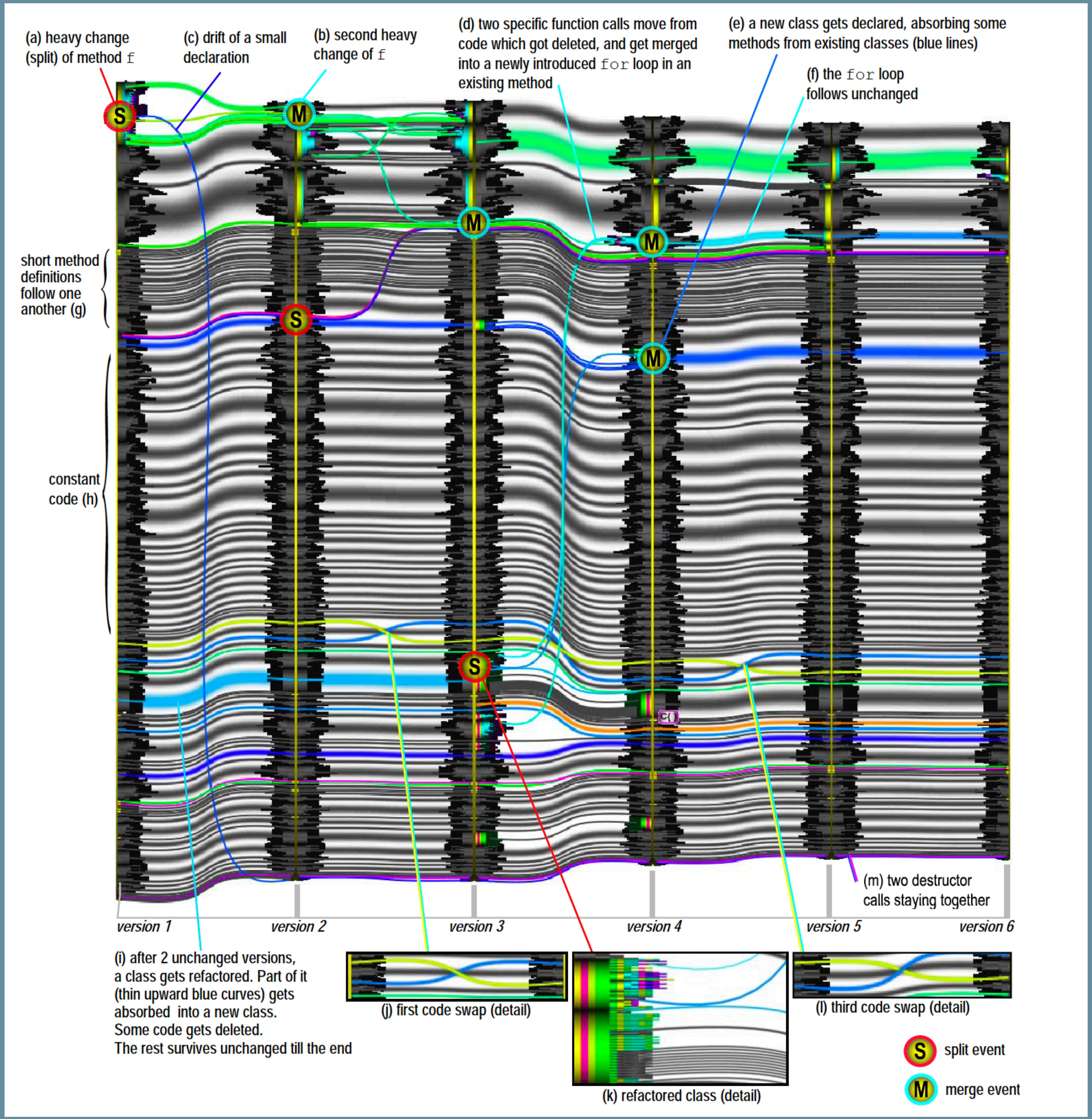




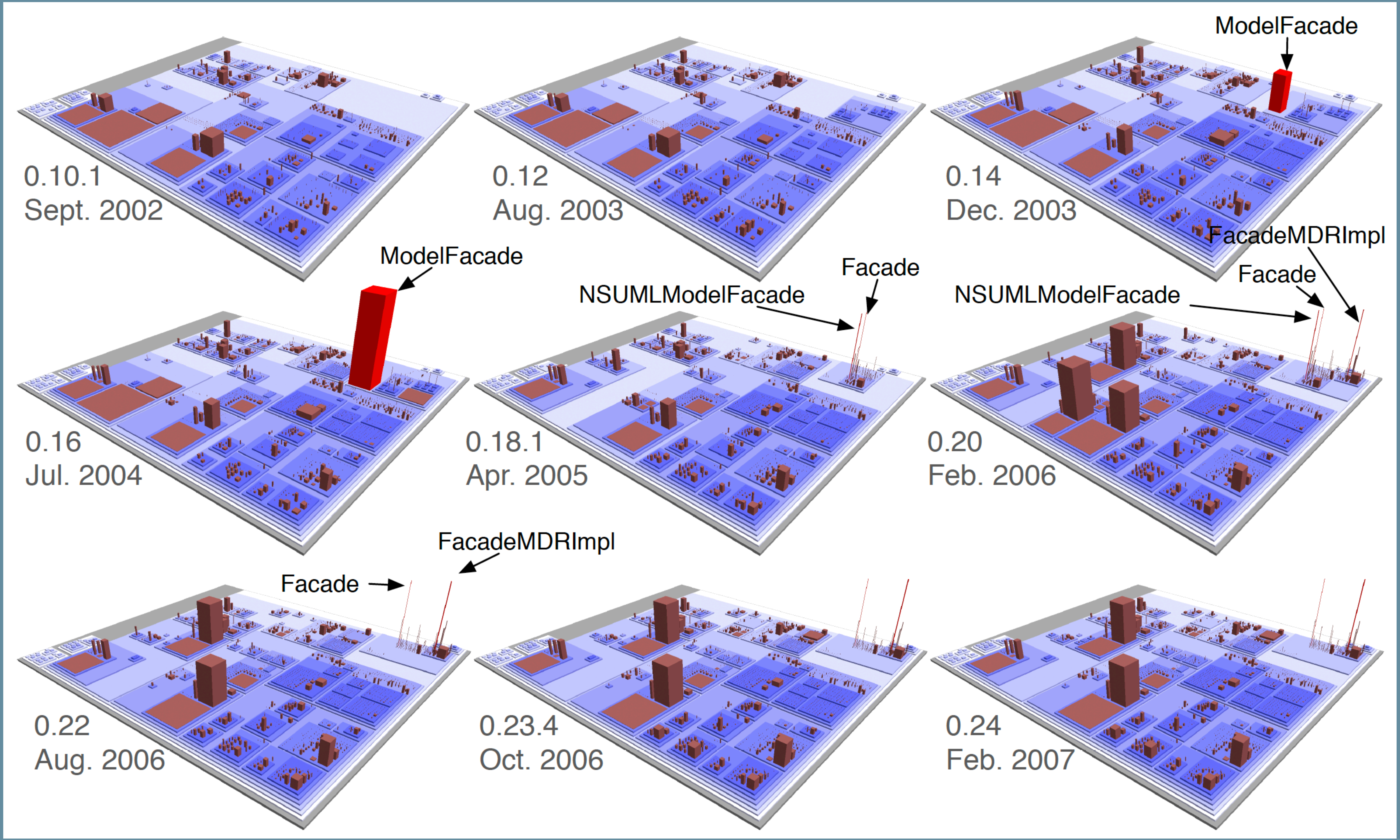
OMG
—
Bob broke the build again!!!



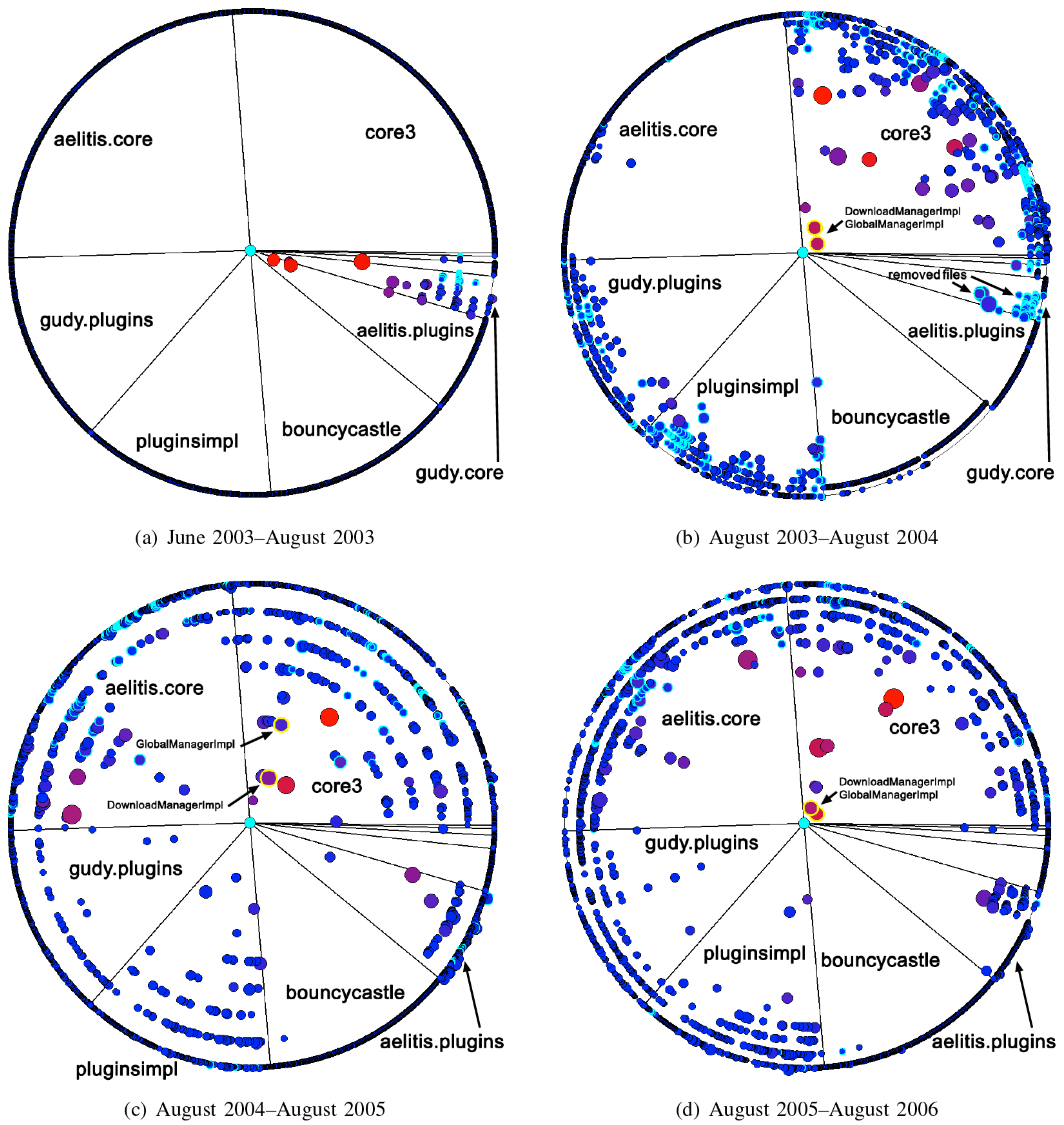
CVSscan: Visualization of Code Evolution



Code Flows: Visualizing Structural Evolution of Source Code



Visual Exploration of Large-Scale System Evolution



(a) June 2003–August 2003 (b) August 2003–August 2004 (c) August 2004–August 2005 (d) August 2005–August 2006

Visualizing Co-Change Information with the Evolution Radar

So, What Happened Next?

Visualizing Software Evolution

Christian Collberg, Stephen Kobourov, Jasvir Nagra

Jacob Pitts and Kevin Wampler

Jasvir Nagra

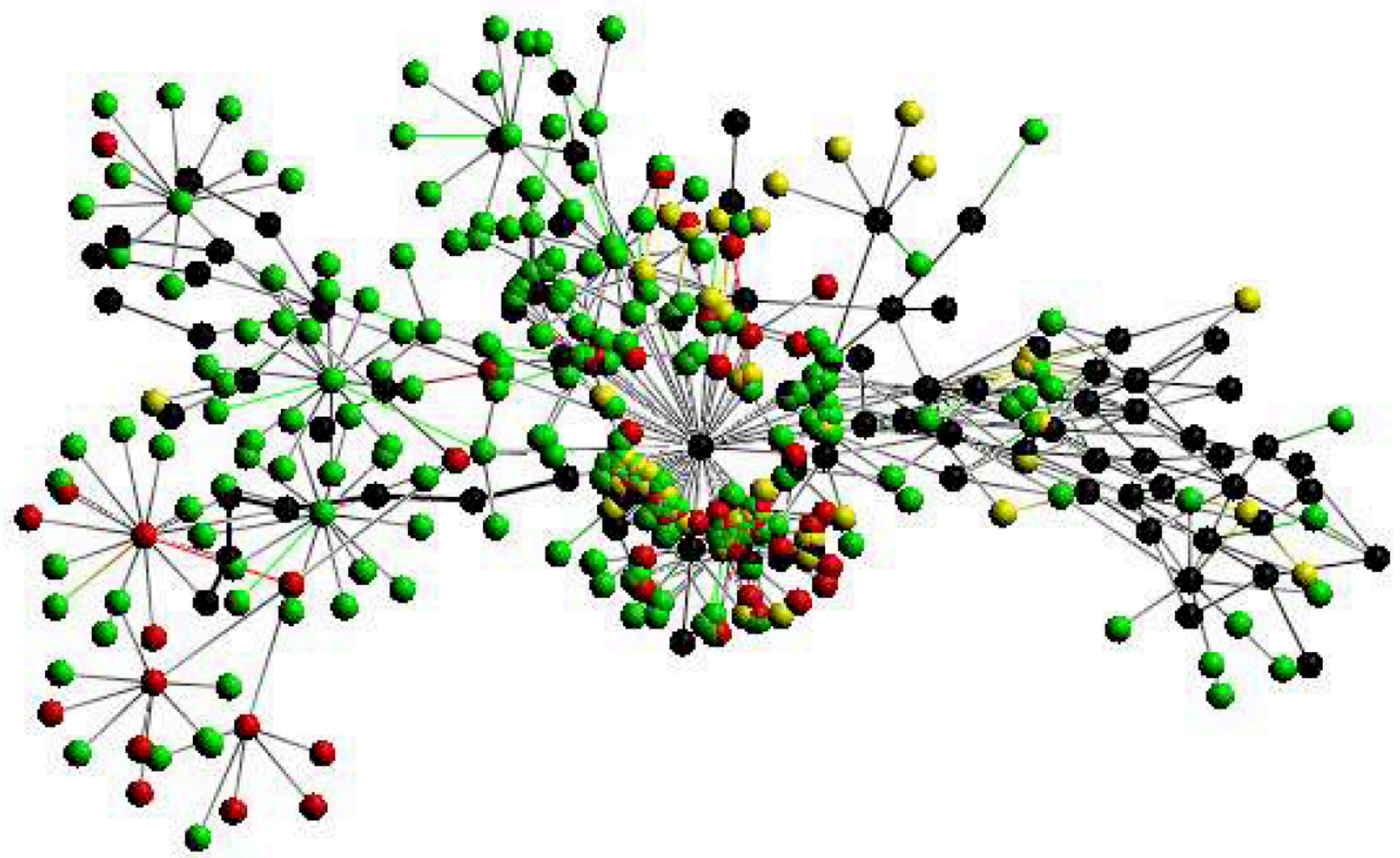
jas@cs.auckland.ac.nz

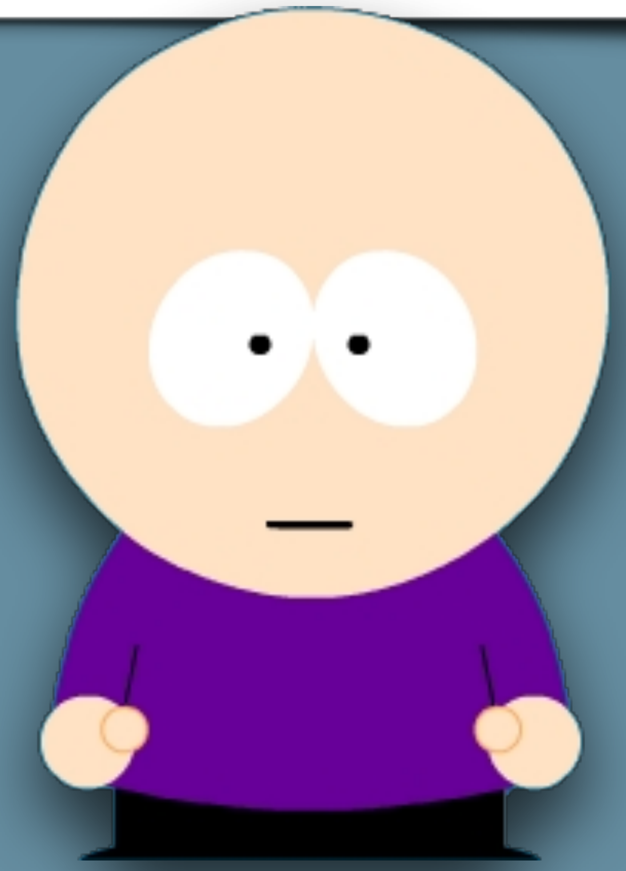
University of Auckland

New Zealand

Inheritanc

Nodes colored by recent author and timestamp





Me

Jasvir

Visualizing Software Evolution

Christian Collberg, Stephen Kobourov, Jasvir Nagra

Jacob Pitts and Kevin Wampler

Jasvir Nagra

jas@cs.auckland.ac.nz

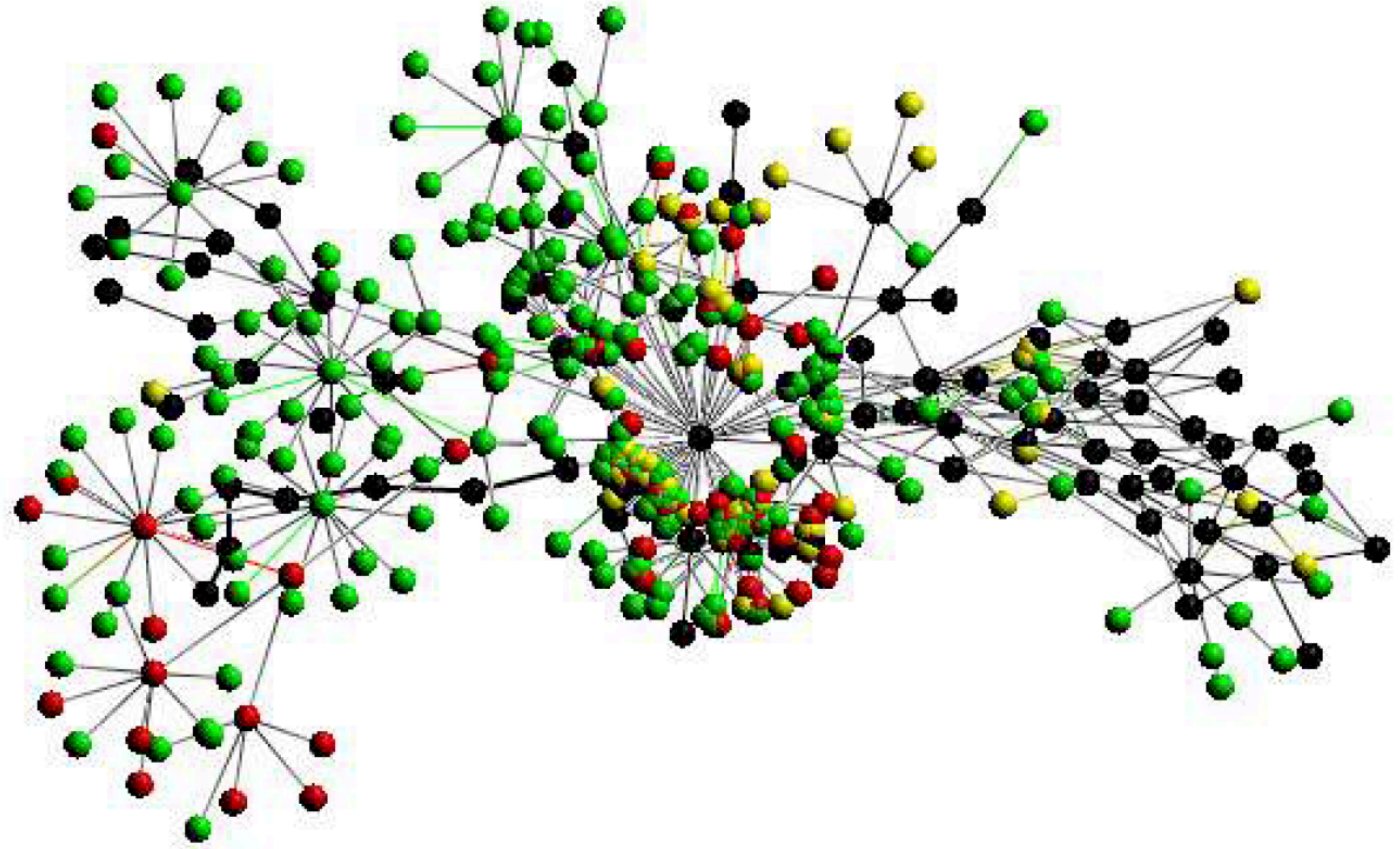
University of Auckland

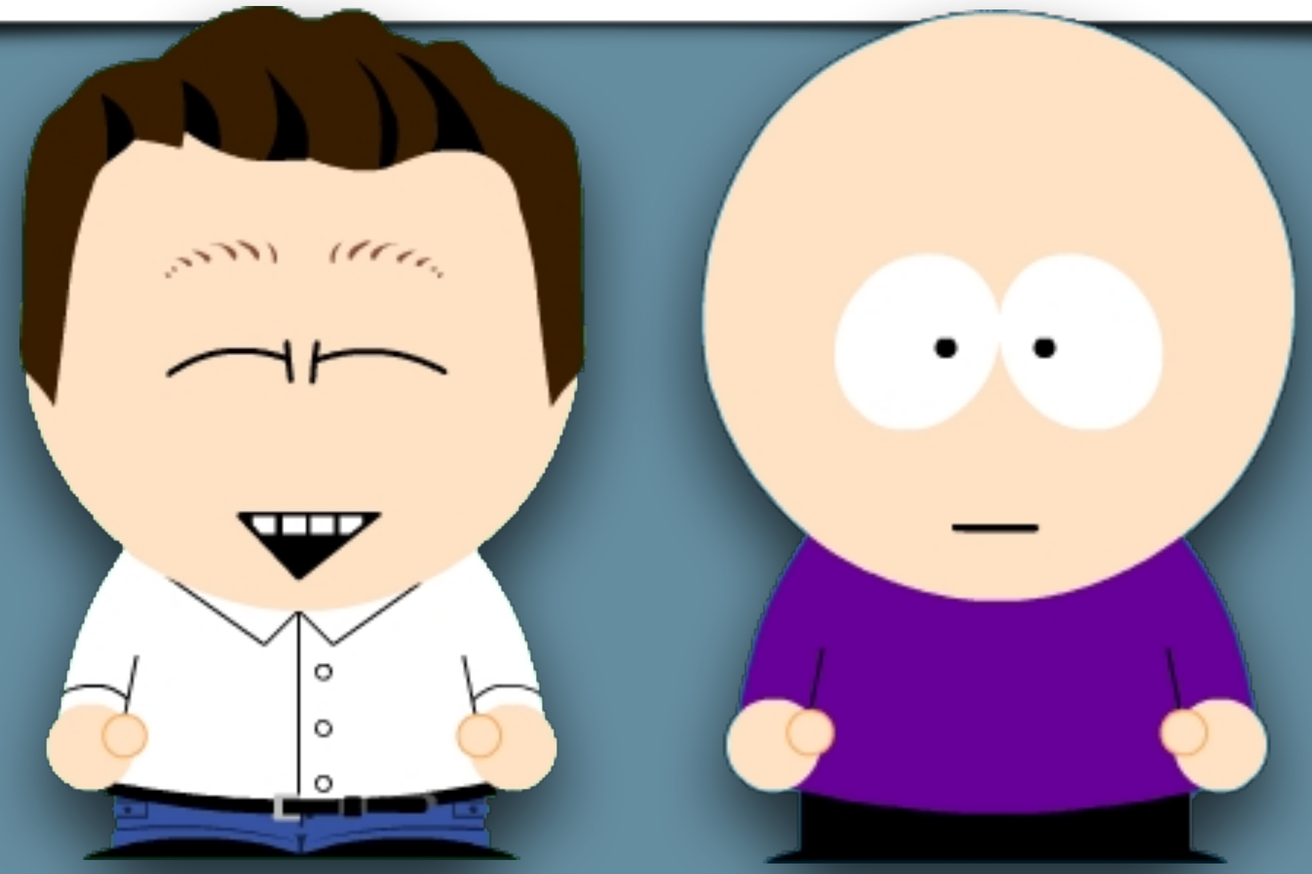
New Zealand

Jasvir

Inheritanc

Nodes colored by recent author and timestamp





My “friend” Todd

Me


Visualizing Software Evolution
Christian Collberg, Stephen Kobourov, Jasvir Nagra
Jacob Pitts and Kevin Wampler
Jasvir Nagra
jas@cs.auckland.ac.nz
A
Jasvir
Inheritanc
Nodes colored by recent author and timestamp
My "friend" Todd
Me

Visualizing Software Evolution
Christian Collberg, Stephen Kobourov, Jasvir Nagra
Jacob Pitts and Kevin Wampler
Jasvir Nagra
jas@cs.auckland.ac.nz
B+
Jasvir
Inheritanc
Nodes colored by recent author and timestamp
My "friend" Todd
Me


Visualizing Software Evolution
Christian Collberg, Stephen Kobourov, Jasvir Nagra
Jacob Pitts and Kevin Wampler
Jasvir Nagra
jas@cs.auckland.ac.nz
B-
Jasvir
Inheritanc
Nodes colored by recent author and timestamp
My "friend" Todd
Me


Visualizing Software Evolution
Christian Collberg, Stephen Kobourov, Jasvir Nagra
Jacob Pitts and Kevin Wampler
Jasvir Nagra
jas@cs.auckland.ac.nz
C
Jasvir
Inheritanc
Nodes colored by recent author and timestamp
My “friend” Todd
Me


Visualizing Software Evolution
Christian Collberg, Stephen Kobourov, Jasvir Nagra
Jacob Pitts and Kevin Wampler
Jasvir Nagra
jas@cs.auckland.ac.nz
D+
Jasvir
Inheritanc
Nodes colored by recent author and timestamp
My “friend” Todd
Me


A R C T I C O C E A N
Greenland
Norwegian Sea
ICELAND
C A N A D A
Hudson Bay
Labrador Sea
Alaska (US)
Bering Sea
U N I T E D S T A T E S
M E X I C O
Gulf of Mexico
Caribbean Sea
A T L A N T I C
O C E A N
P A C I F I C
O C E A N
French Polynesia
COLOMBIA
VENEZUELA
PERU
B R A Z I L
BOLIVIA
PARAGUAY
ARGENTINA
CHILE
R U S S I A
GERMANY
POLAND
UKRAINE
FRANCE
TURKEY
K A Z A K H S T A N
MONG
C H
ALGERIA
LIBYA
EGYPT
MAURITANIA
MALI
NIGER
CHAD
SUDAN
NIGERIA
ETHIOPIA
CENTRAL AFRICAN REP.
SOMALIA
KENYA
DEM. REP. OF THE CONGO
TANZANIA
ANGOLA
NAMIBIA
BOTSWANA
SOUTH AFRICA
MADAGASCAR
MOZAMBIQUE
SAUDI ARABIA
YEMEN
OMAN
IRAN
AFGHANISTAN
PAKISTAN
INDIA
SRI LANKA
Arabian Sea
THAILAND
PHILIPPINES
MALAYSIA
I N D O N E S I A
PAPUA NEW GUINEA
MICRONESIA
I N D I A N
O C E A N
A U S T R A L I A
Coral Sea
Tasman Sea
NEW ZEALAND
French Southern Territories

Visualizing Software Evolution

Christian Collberg, Stephen Kobourov, Jasvir Nagra

Jacob Pitts and Kevin Wampler

Jasvir Nagra

jas@cs.auckland.ac.nz

University of Auckland

New Zealand


ARCTIC OCEAN
Greenland
CANADA
UNITED STATES
MEXICO
ATLANTIC
OCEAN
PACIFIC
OCEAN
BRAZIL
ARGENTINA
ALGERIA
LIBYA
EGYPT
CHAD
SUDAN
NIGERIA
ETHIOPIA
ANGOLA
NAMIBIA
SOUTH AFRICA
SAUDI ARABIA
RUSSIA
KAZAKHSTAN
INDIA
INDIAN
OCEAN
INDONESIA
AUSTRALIA
PHILIPPINES
PAPUA NEW GUINEA
ICELAND

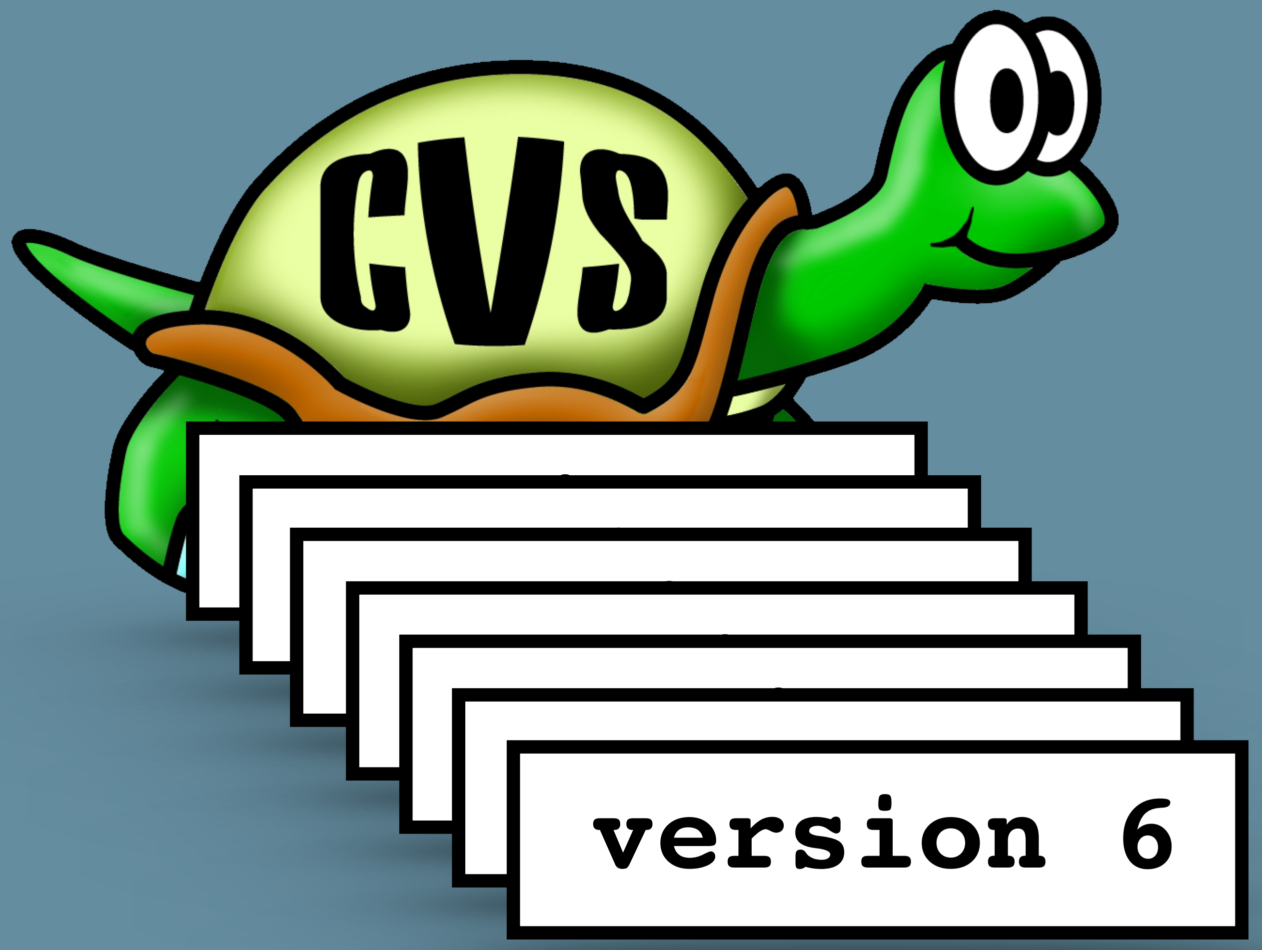
nzherald.co.nz
BEST DAILY NEWSPAPER
THURSDAY, SEPTEMBER 16, 2010 AUCKLAND EDITION $1.90
The New Zealand Herald
You're not in Guatemala now... A5
LONELY PLANET'S LATEST VERDICT ON NEW ZEALAND
TimeOut LIFTOUT Best TV, movies, music
BRANDON ROUTH, SUPERMAN AND SPECIAL AGENT
Why the housing market is GOOD NEWS B2
PLUS Aucklander YOUR CITY, YOUR NEWS
TOASTMASTERS INTERNATIONAL
WHERE LEADERS ARE MADE
BUT STILL NOT OUT
The political life and times of DAVID GARRETT
NOVEMBER 2008
JUNE 2009
MAY 9, 2010
SEPTEMBER 2010
PACIFIC
OCEAN
INDIAN
AUSTRALIA
INDONESIA
INDIA
PHILIPPINES
PAPUA NEW GUINEA


ECE


TOASTMASTERS
INTERNATIONAL
T
A R C T I C O C E A N
Greenland
Alaska
C A N A D A
R U S S I A
ATLANTIC
OCEAN
P A
O C E A N
INDIAN
O C E A N
O C E A N
A L G E R I A
L I B Y A
E G Y P T
S U D A N
C H A D
NIGERIA
ETHIOPIA
KENYA
TANZANIA
ANGOLA
NAMIBIA
BOTSWANA
SOUTH
AFRICA
SAUDI
ARABIA
I N D I A
PAKISTAN
KAZAKHSTAN
MONG
THAILAND
PHILIPPINES
INDONESIA
PAPUA NEW GUINEA
A U S T R A L I A
NEW ZEALAND
B R A Z I L
MAURITANIA
UKRAINE
TURKEY
YEMEN
OMAN
MADAGASCAR

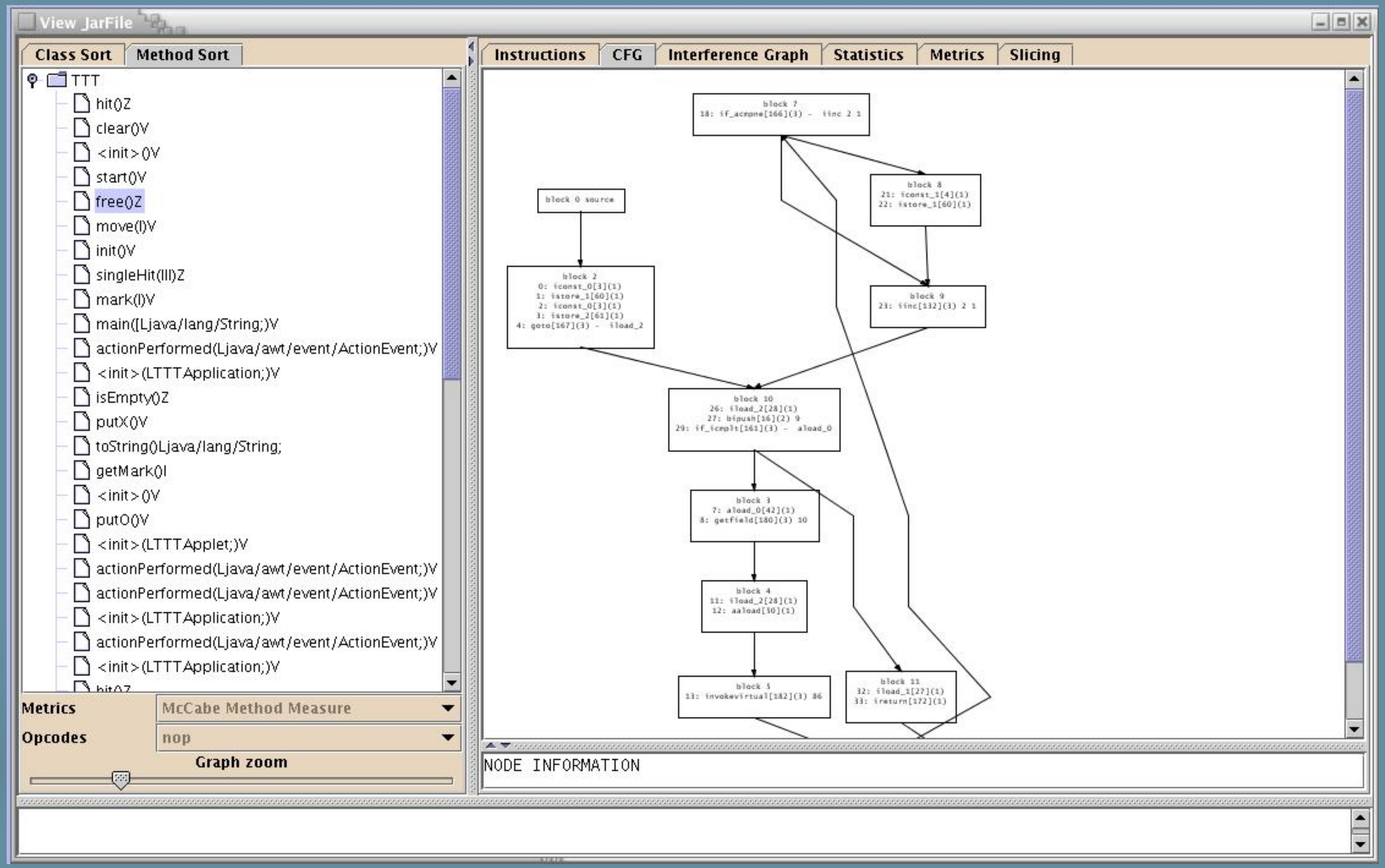

TOASTMASTERS
INTERNATIONAL
Semi Finals

But, can it be Replicated?



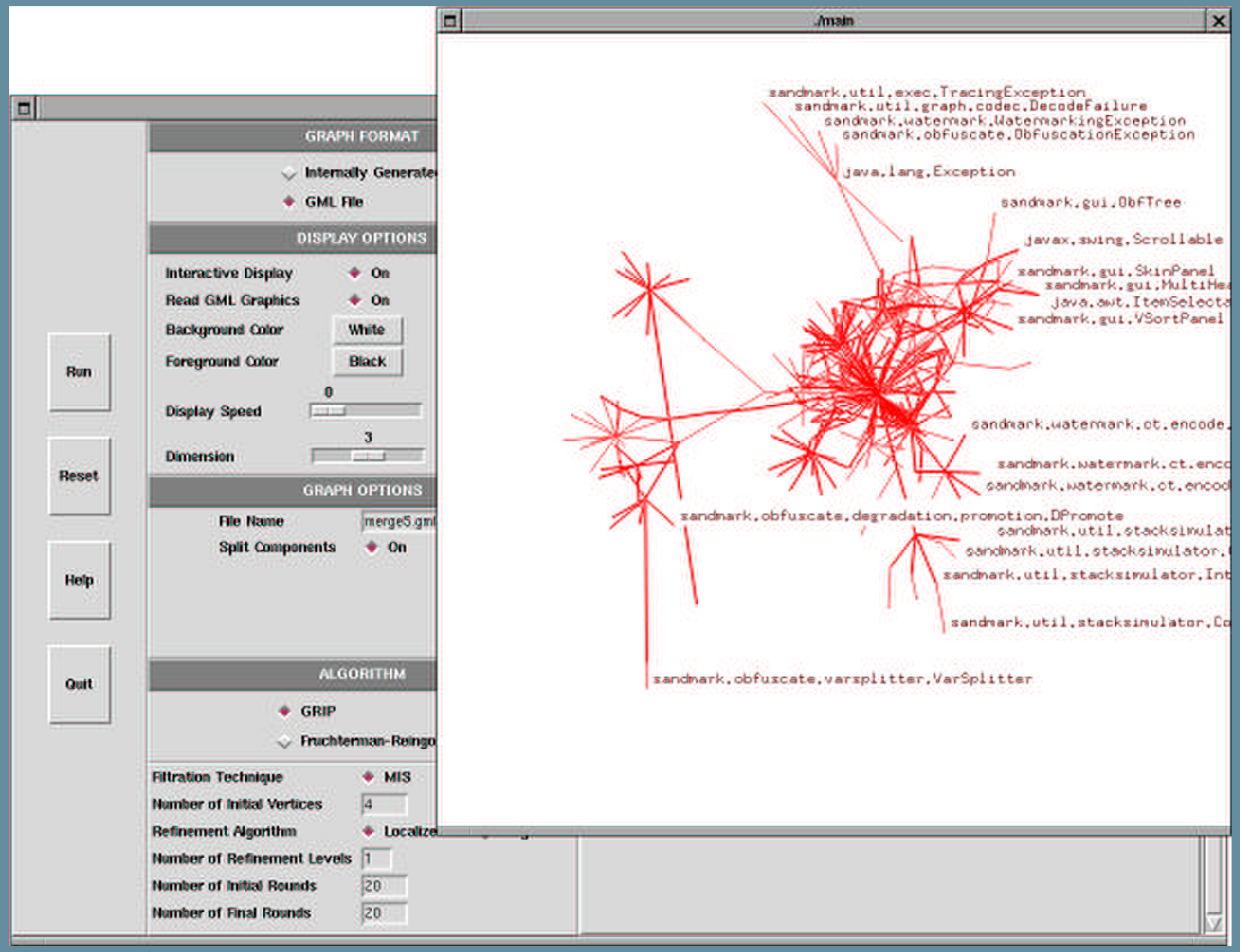
①

SandMark



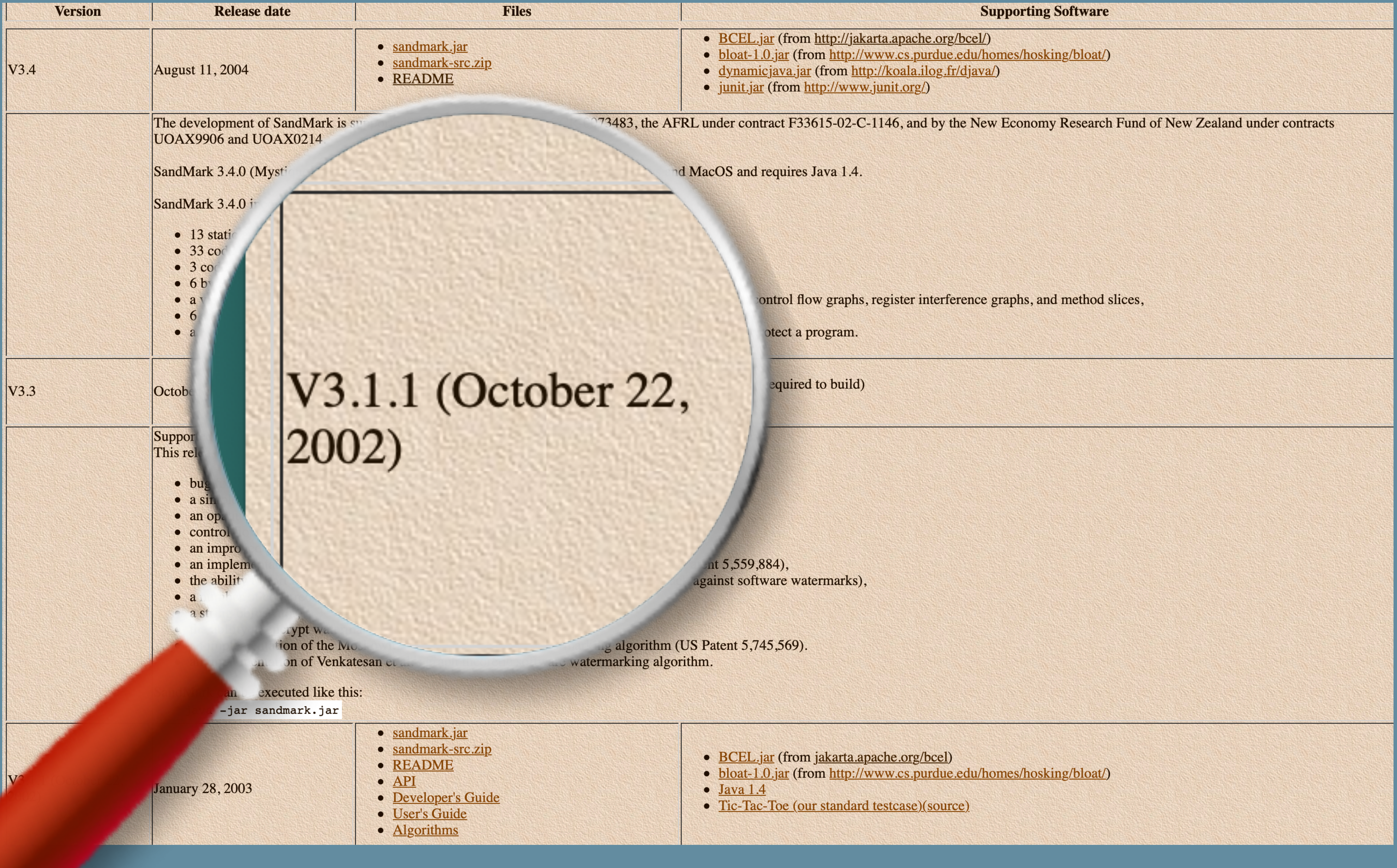
②

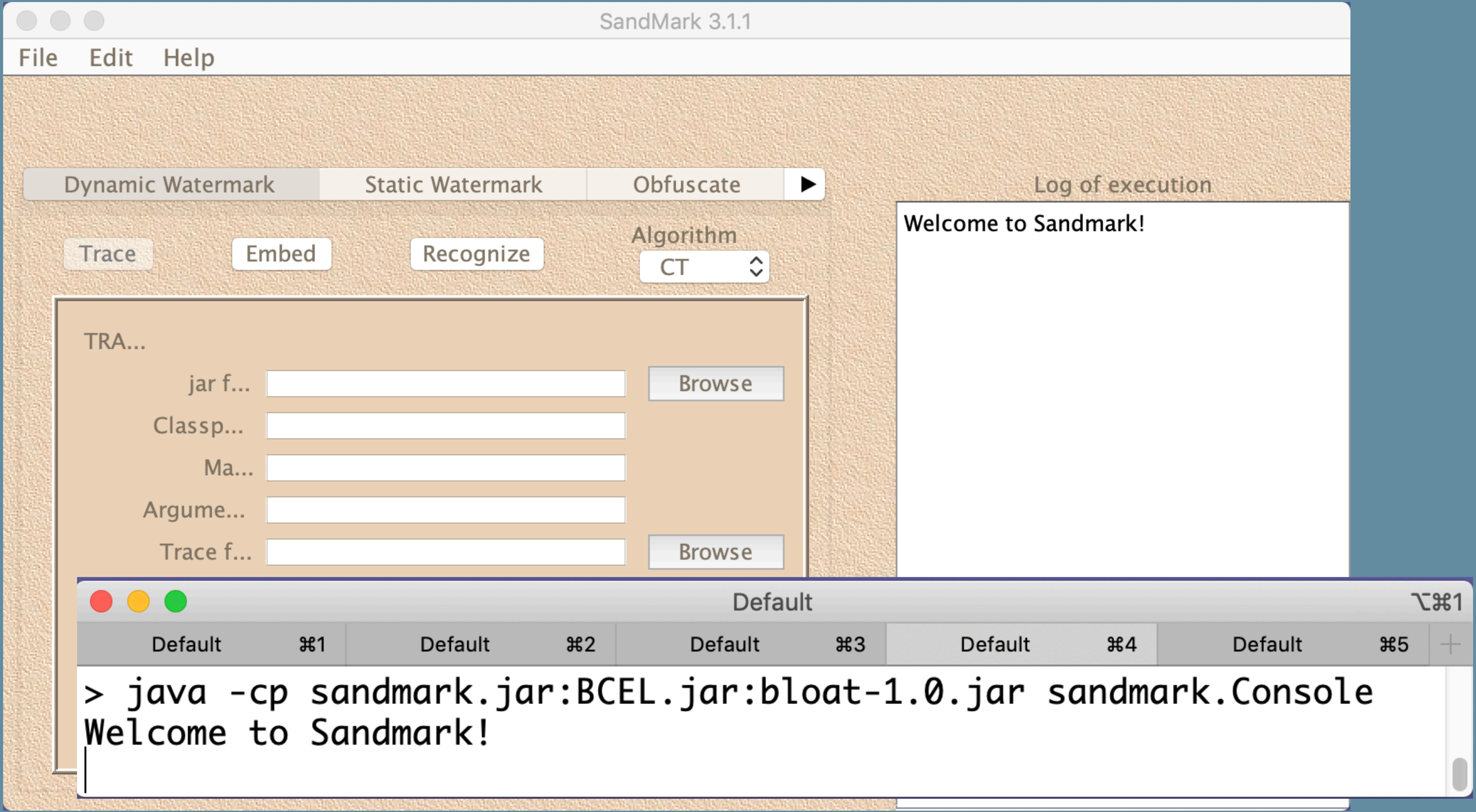
tgrip

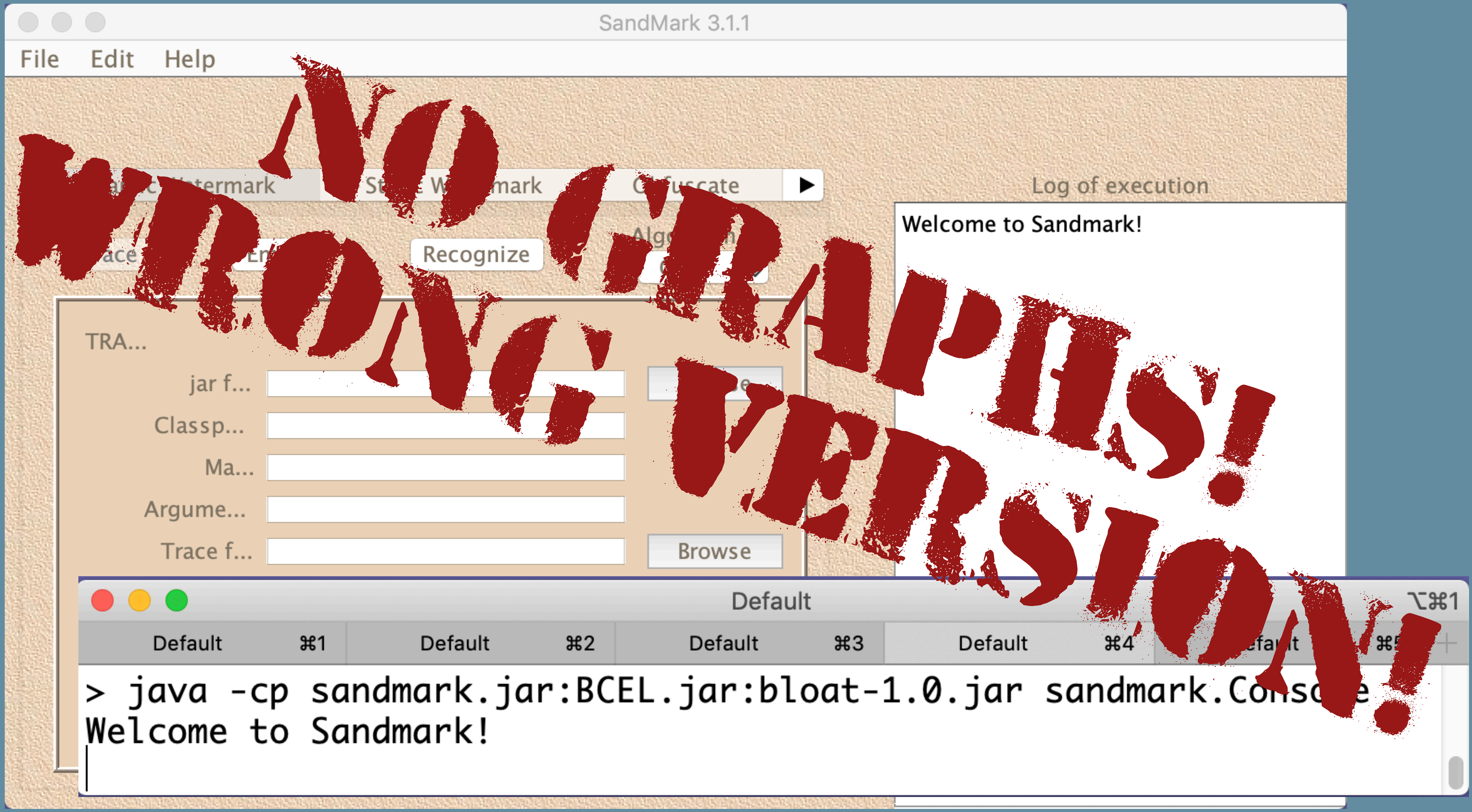


③

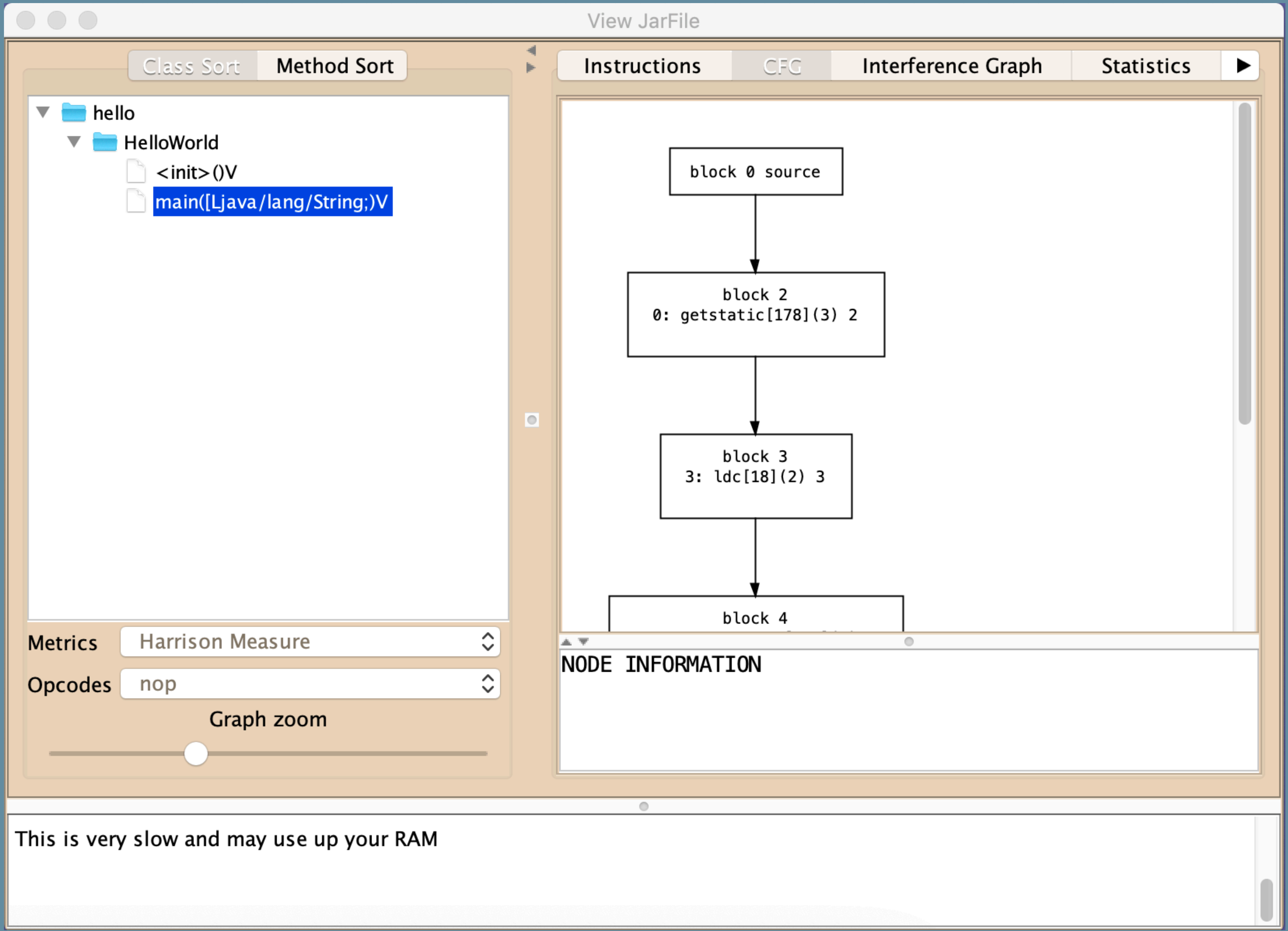
sandmark.cs.arizona.edu

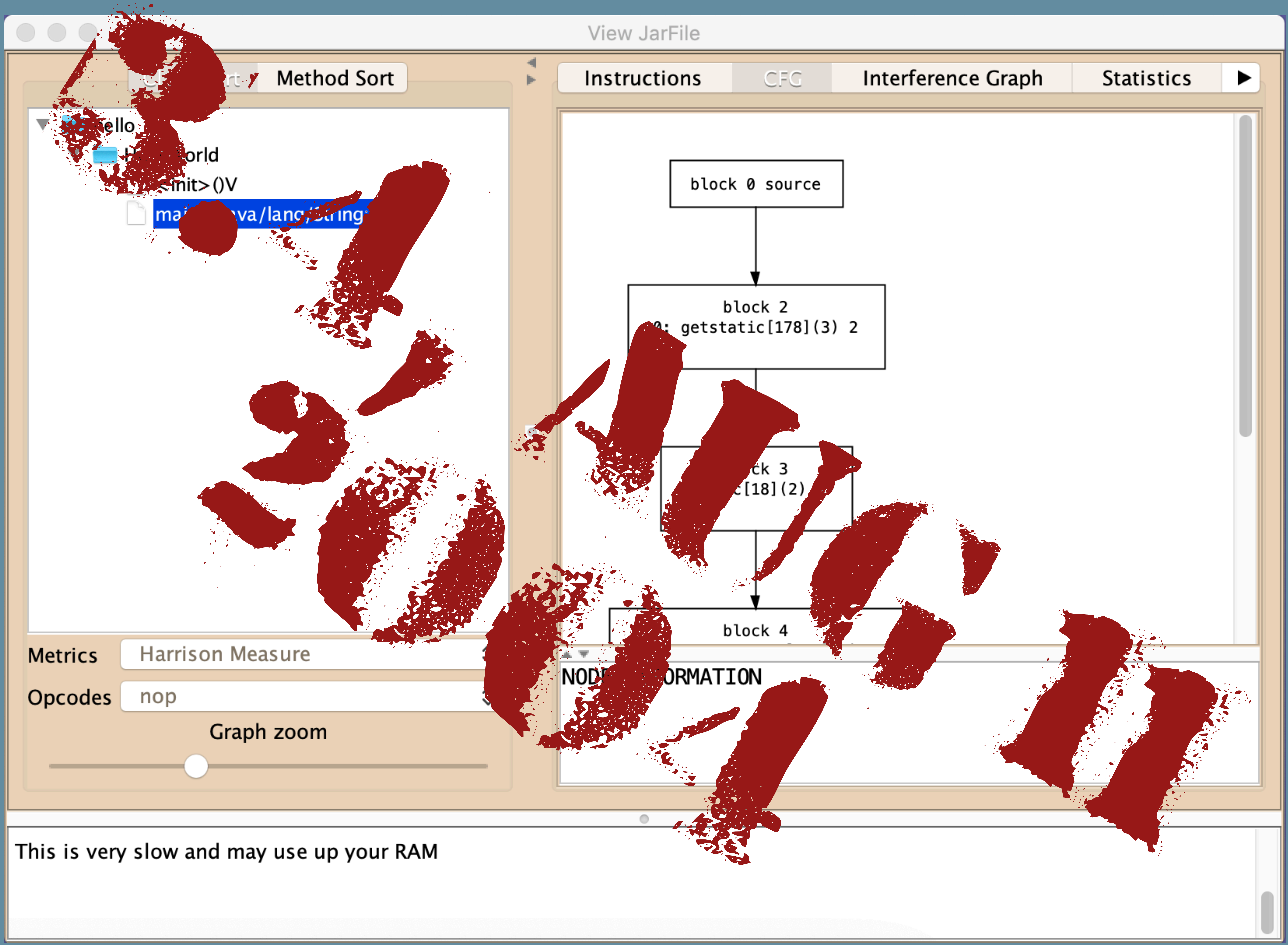



SandMark 3.1.1
File Edit Help
Dynamic Watermark
Static Watermark
Obfuscate
Log of execution
Trace
Embed
Recognize
Algorithm
CT
Welcome to Sandmark!
TRA...
jar f...
Browse
Classp...
Ma...
Argume...
Trace f...
Browse
Default
⌥⌘1
Default ⌘1
Default ⌘2
Default ⌘3
Default ⌘4
Default ⌘5
> java -cp sandmark.jar:BCEL.jar:bloat-1.0.jar sandmark.Console
Welcome to Sandmark!

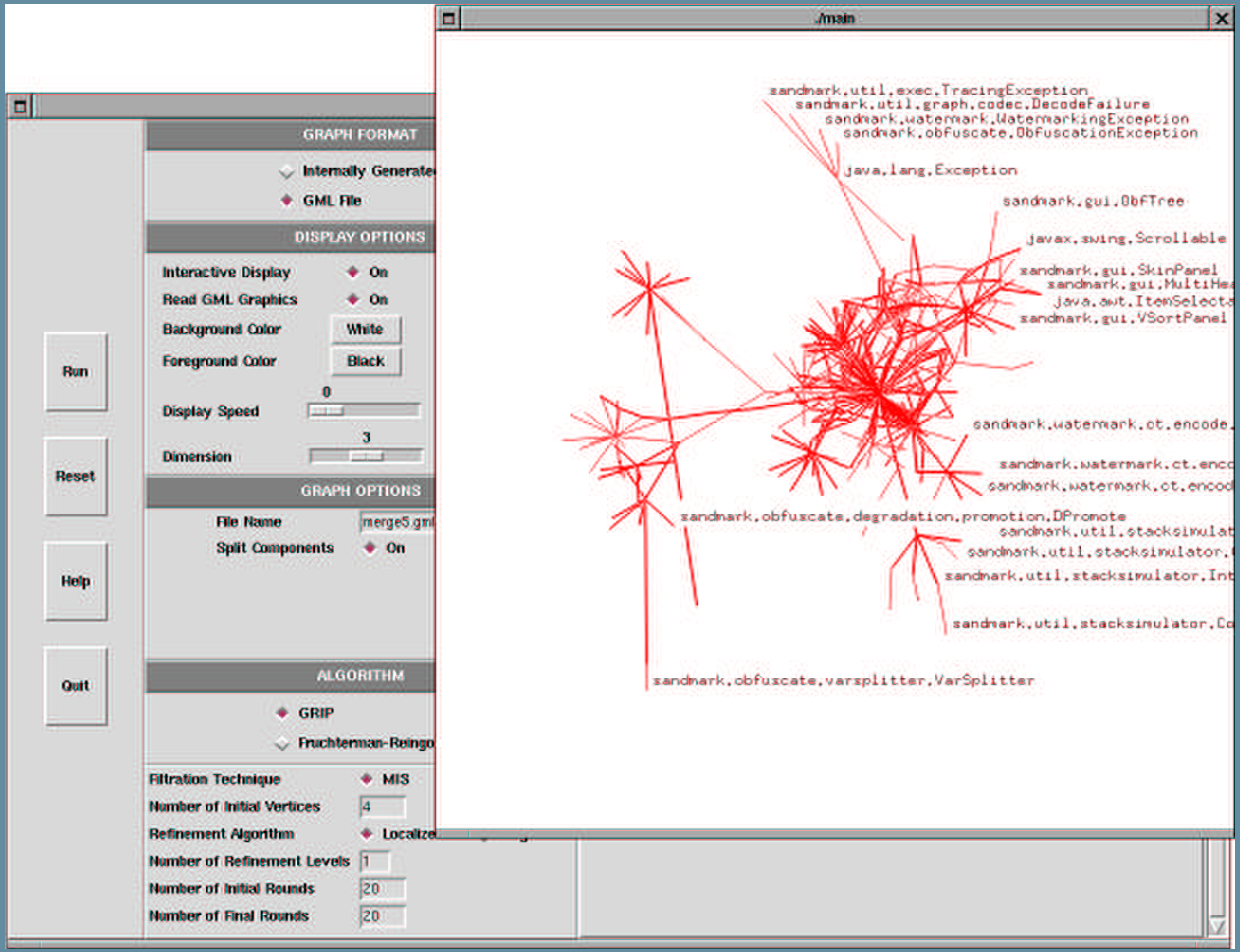

SandMark 3.1.1
File Edit Help
Recognize
Log of execution
Welcome to Sandmark!
TRA...
jar f...
Classp...
Ma...
Argume...
Trace f...
Browse
Default
Default ⌘1
Default ⌘2
Default ⌘3
Default ⌘4
> java -cp sandmark.jar:BCEL.jar:bloat-1.0.jar sandmark.Console
Welcome to Sandmark!
NO GRAPHS!
WRONG VERSION!

SandMark 3.1.1
File
Edit
Help
NO GRAPHS!
WRONG VERSION!
3.3
Recognize
TRA...
jar f...
Classp...
Ma...
Argume...
Trace f...
Browse
Default
Default ⌘1
Default ⌘2
Default ⌘3
Default ⌘4
> java -cp sandmark.jar:BCEL.jar:bloat-1.0.jar sandmark.Co
Welcome to Sandmark!

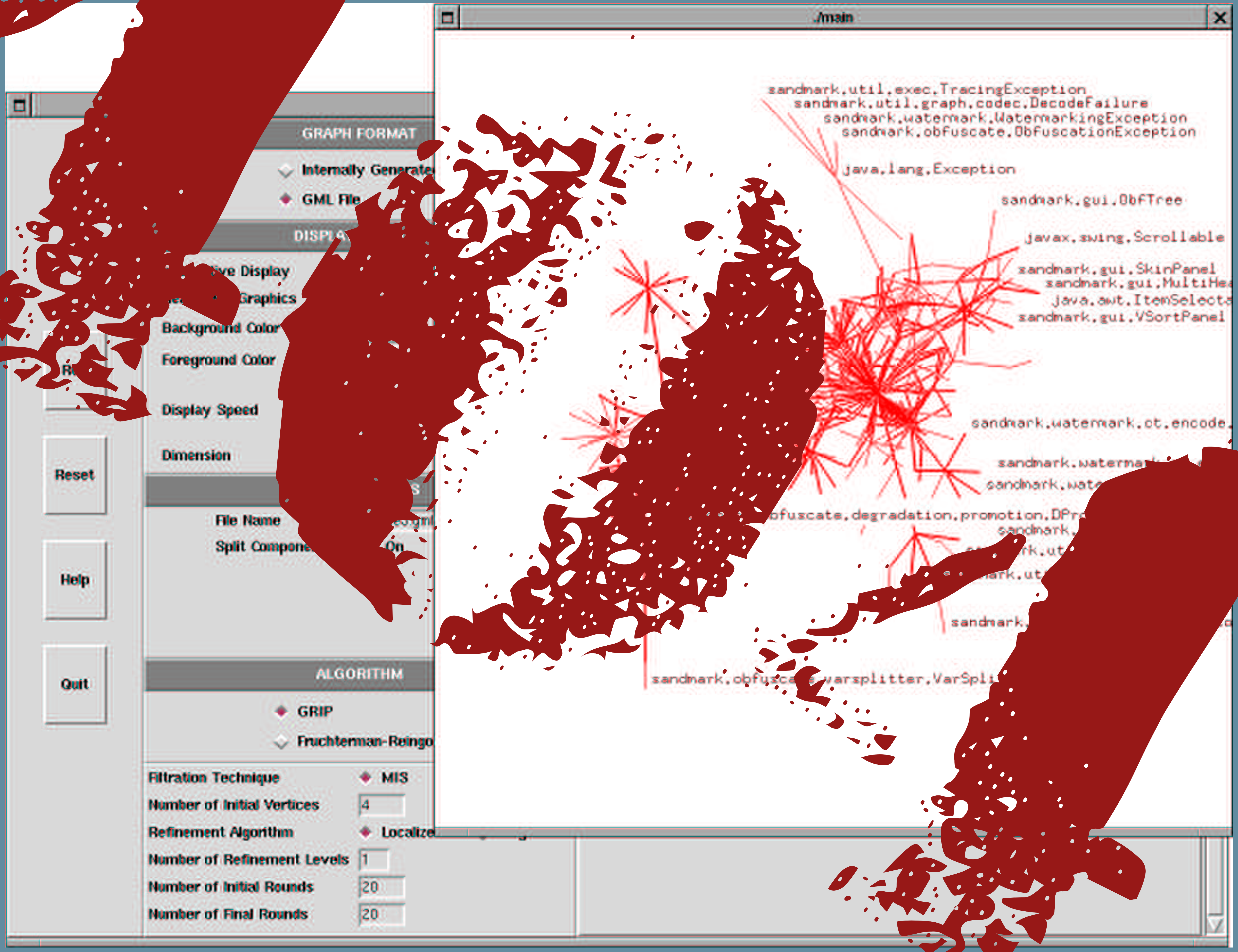

View JarFile
Class Sort
Method Sort
hello
HelloWorld
<init>()V
main([Ljava/lang/String;)V
Metrics
Harrison Measure
Opcodes
nop
Graph zoom
Instructions
CFG
Interference Graph
Statistics
block 0 source
block 2
0: getstatic[178](3) 2
block 3
3: ldc[18](2) 3
block 4
NODE INFORMATION
This is very slow and may use up your RAM


View JarFile
Method Sort
Instructions
CFG
Interference Graph
Statistics
block 0 source
block 2
getstatic[178](3) 2
block 4
Metrics
Harrison Measure
Opcodes
nop
Graph zoom
This is very slow and may use up your RAM

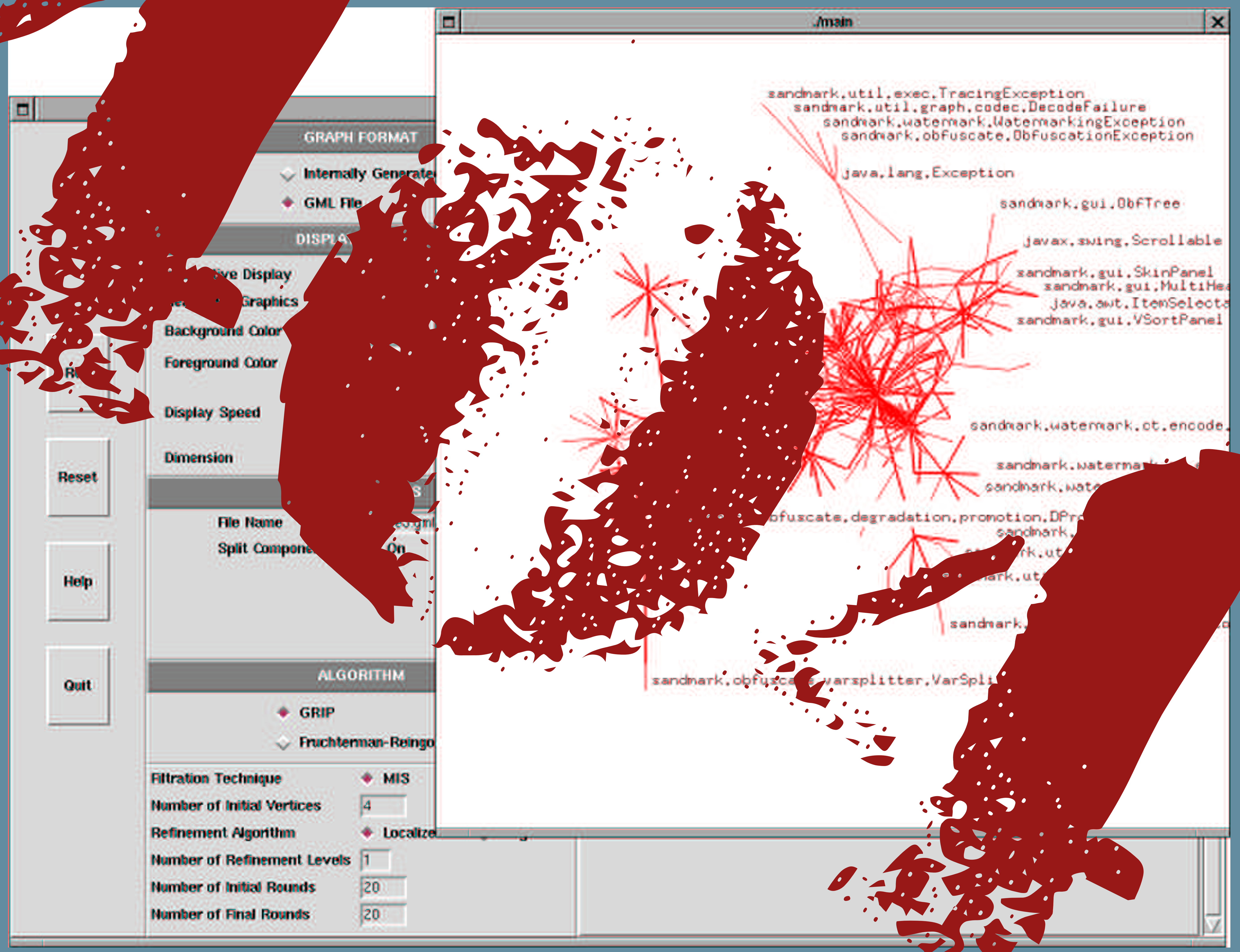
tgrip.cs.arizona.edu



tgrip.cs.arizona.edu



tgrip.cs.arizona.edu



graphael.cs.arizona.edu


version 6

Our current test case is the SandMark system which consists of approximately 90,000 lines of code developed over 200 days.

Paper submission: Dec. 16, 2002

Our current test case is the SandMark system which consists of approximately 90,000 lines of code developed over 200 days.

Default ⌥⌘2

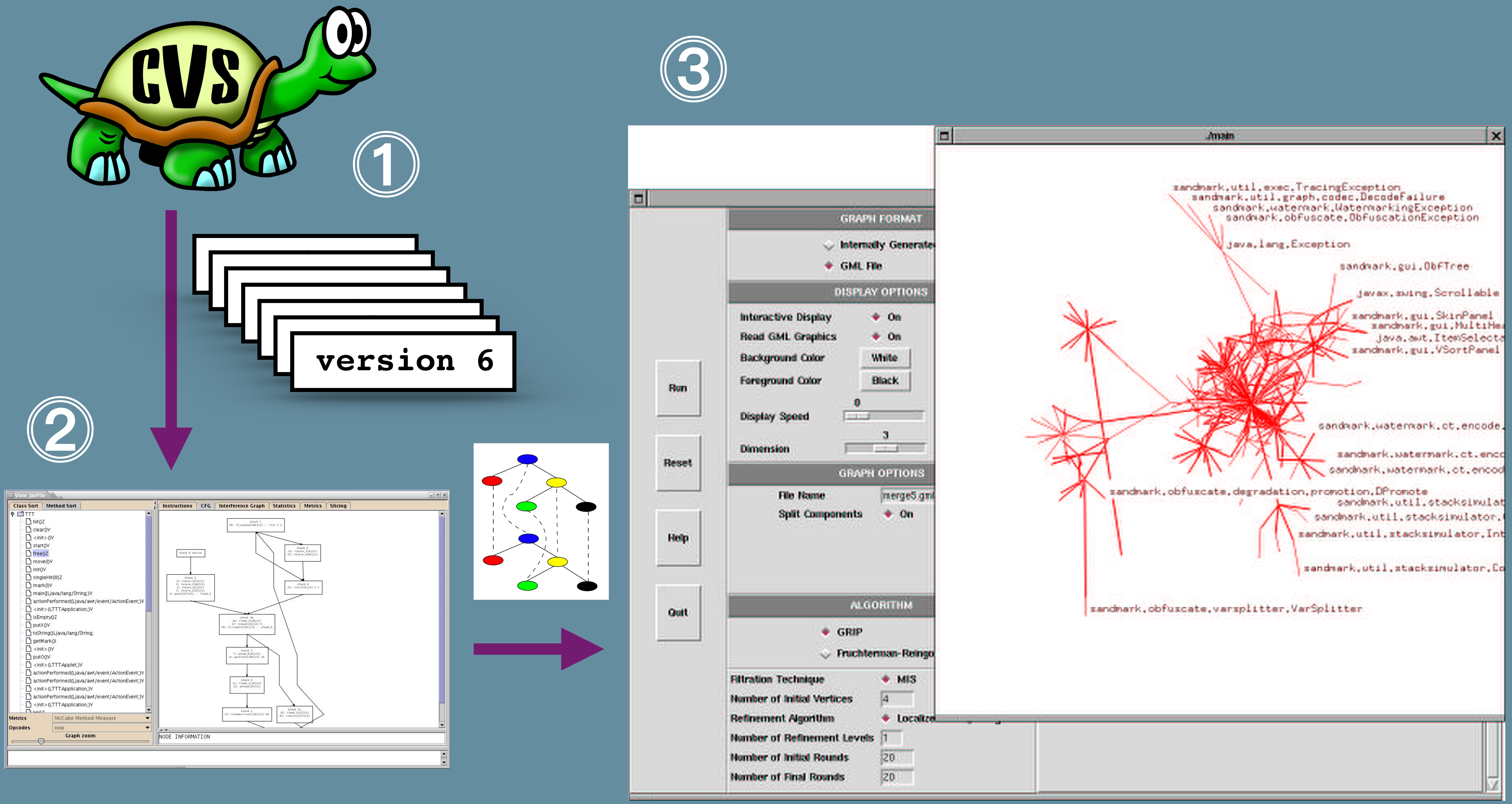
```
> cd /Users/collberg/wmark/smark3
> cvs log |& gawk '/date: 2002/{print $2}' | sort -u | wc -w
     244
```

Paper submission: Dec. 16, 2002

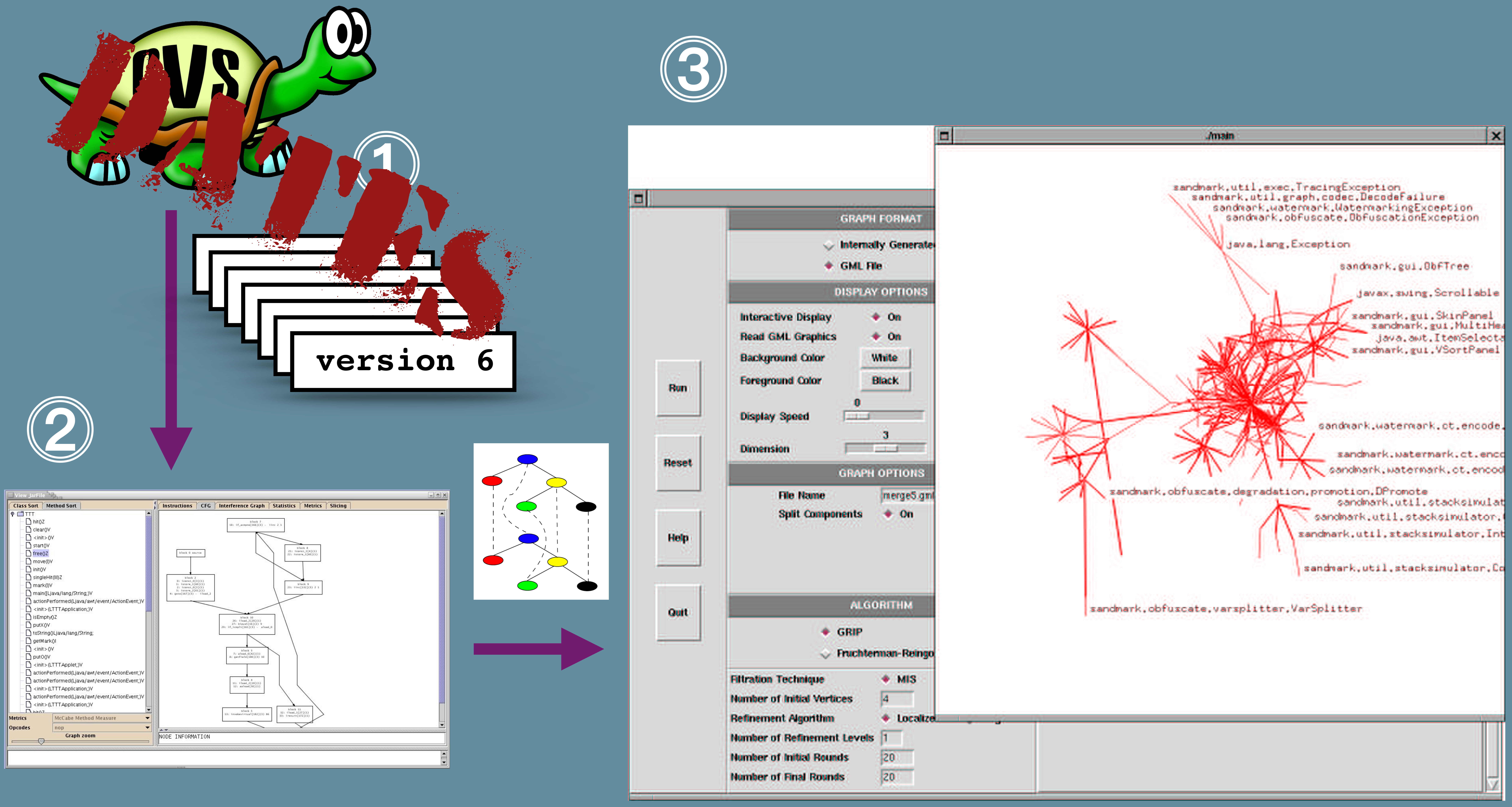
version 6

Our current test case is the SandMark system[illegible]which [illegible] of approximately 90,000 lines of code deve[illegible]pe[illegible] o[illegible] da[illegible]

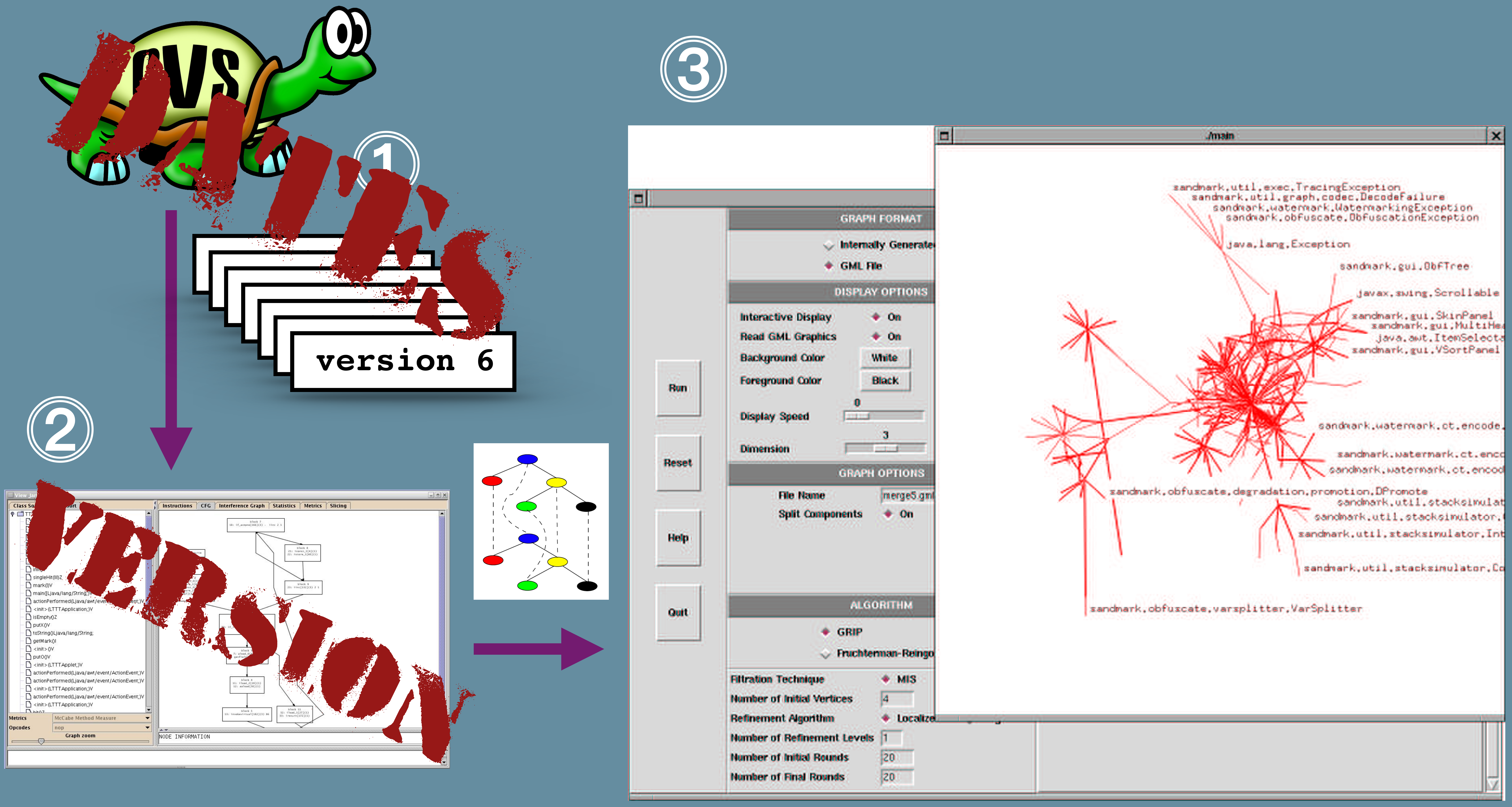
```
Default                                                    ⌥⌘2
> cd /Users/collberg/wmark/smark3
> cvs log | & gawk '/date: 2002/{print $2}' [illegible]rt -u
    244
```



CVS
①
version 6
②
③
GRAPH FORMAT
Internally Generated
GML File
DISPLAY OPTIONS
Interactive Display On
Read GML Graphics On
Background Color White
Foreground Color Black
Display Speed
Dimension
GRAPH OPTIONS
File Name
Split Components On
ALGORITHM
GRIP
Fruchterman-Reingold
Filtration Technique MIS
Number of Initial Vertices 4
Number of Refinement Levels 1
Number of Initial Rounds 20
Number of Final Rounds 20
Run
Reset
Help
Quit
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote
sandmark.obfuscate.varsplitter.VarSplitter



①
②
③
version 6
View JarFile
Class Sort
Method Sort
Instructions
CFG
Interference Graph
Statistics
Metrics
Slicing
Metrics
McCabe Method Measure
Opcodes
nop
Graph zoom
NODE INFORMATION
GRAPH FORMAT
Internally Generated
GML File
DISPLAY OPTIONS
Interactive Display
On
Read GML Graphics
On
Background Color
White
Foreground Color
Black
Display Speed
0
Dimension
3
GRAPH OPTIONS
File Name
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingold
Filtration Technique
MIS
Number of Initial Vertices
4
Refinement Algorithm
Number of Refinement Levels
1
Number of Initial Rounds
20
Number of Final Rounds
20
Run
Reset
Help
Quit
./main
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote
sandmark.obfuscate.varsplitter.VarSplitter

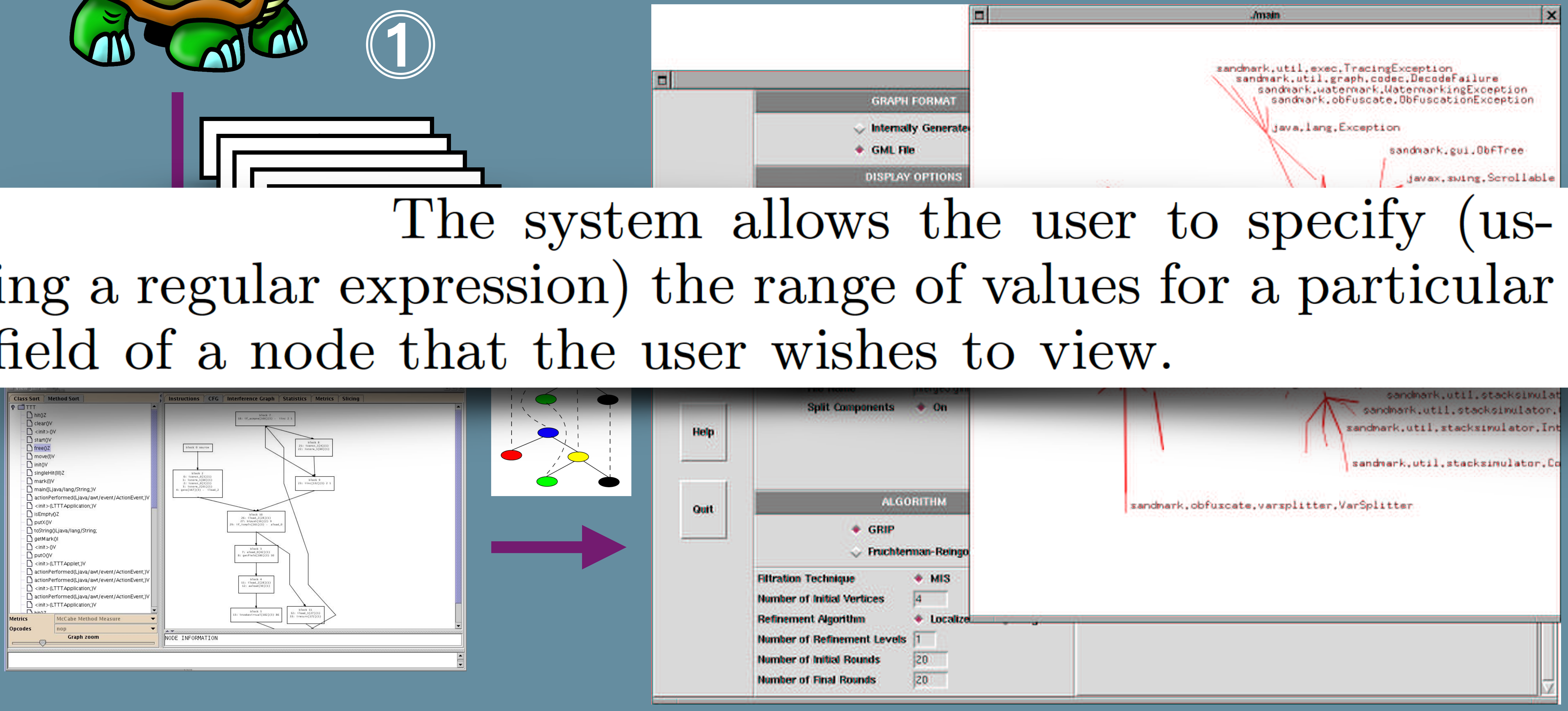


CVS
DIFFS
①
version 6
②
VERSION
③
./main
GRAPH FORMAT
Internally Generated
GML File
DISPLAY OPTIONS
Interactive Display
On
Read GML Graphics
On
Background Color
White
Foreground Color
Black
Display Speed
Dimension
GRAPH OPTIONS
File Name
merge5.gml
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingold
Filtration Technique
MIS
Number of Initial Vertices
4
Refinement Algorithm
Number of Refinement Levels
1
Number of Initial Rounds
20
Number of Final Rounds
20
Run
Reset
Help
Quit
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote
sandmark.obfuscate.varsplitter.VarSplitter
Instructions
CFG
Interference Graph
Statistics
Metrics
Slicing
NODE INFORMATION
Graph zoom

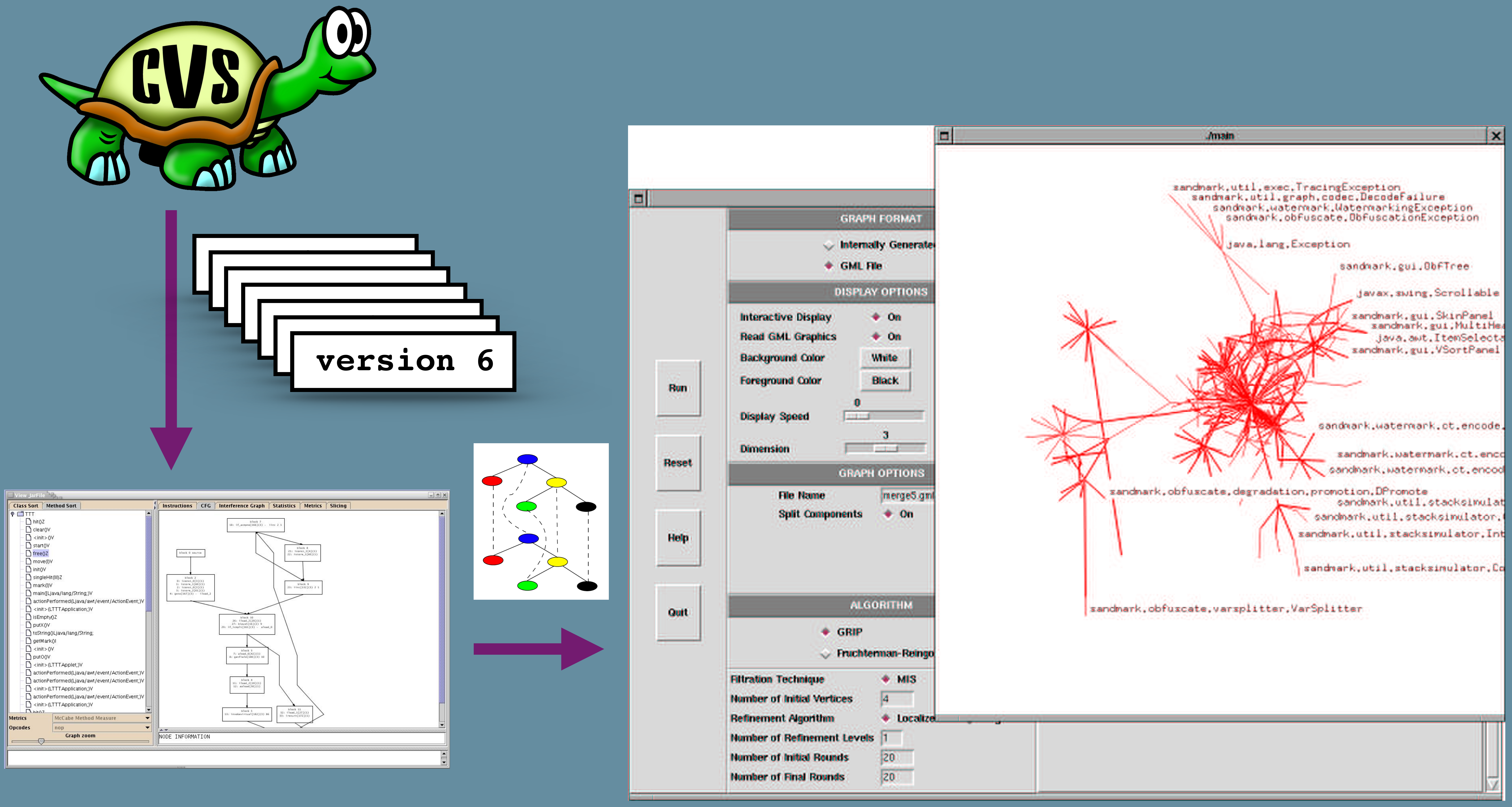

③
①
②
CVS
version 6
VERSION
./main
GRAPH OPTIONS
File Name
merge5.gml
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingo
Filtration Technique
MIS
Number of Initial Vertices
4
Refinement Algorithm
Number of Refinement Levels
1
Number of Initial Rounds
20
Number of Final Rounds
20
Interactive Display
Read GML Graphics
Background Color
Foreground Color
Display Speed
Dimension
Run
Reset
Help
Quit
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
NODE INFORMATION


④ How were experiments carried out?
①
②
③
CVS
version 6
VERSION
GRAPH FORMAT
DISPLAY
Interactive Display
Read GML Graphics
Background Color
Foreground Color
Display Speed
Dimension
GRAPH OPTIONS
File Name
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingold
Filtration Technique
MIS
Number of Initial Vertices
Refinement Algorithm
Number of Refinement Levels
Number of Initial Rounds
Number of Final Rounds
Run
Reset
Help
Quit
./main
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable

④ How were experiments carried out?

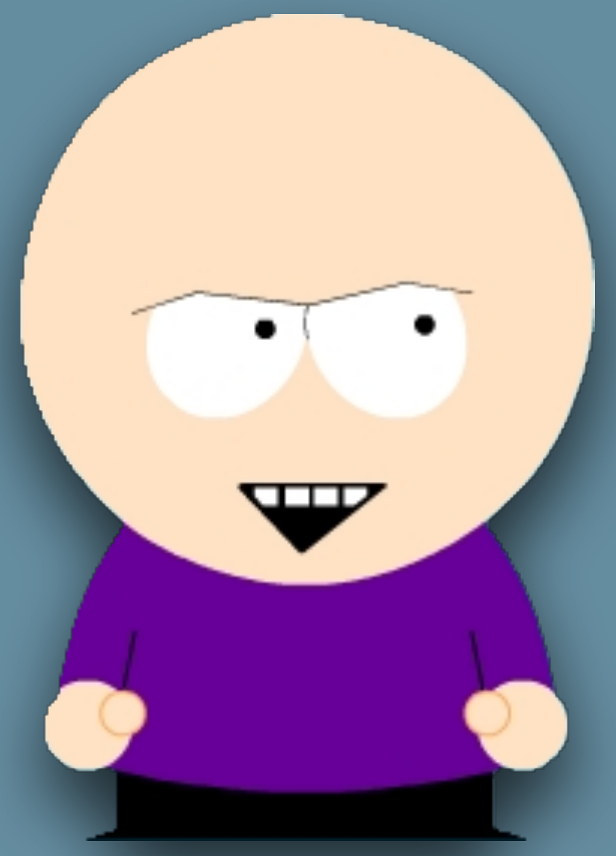


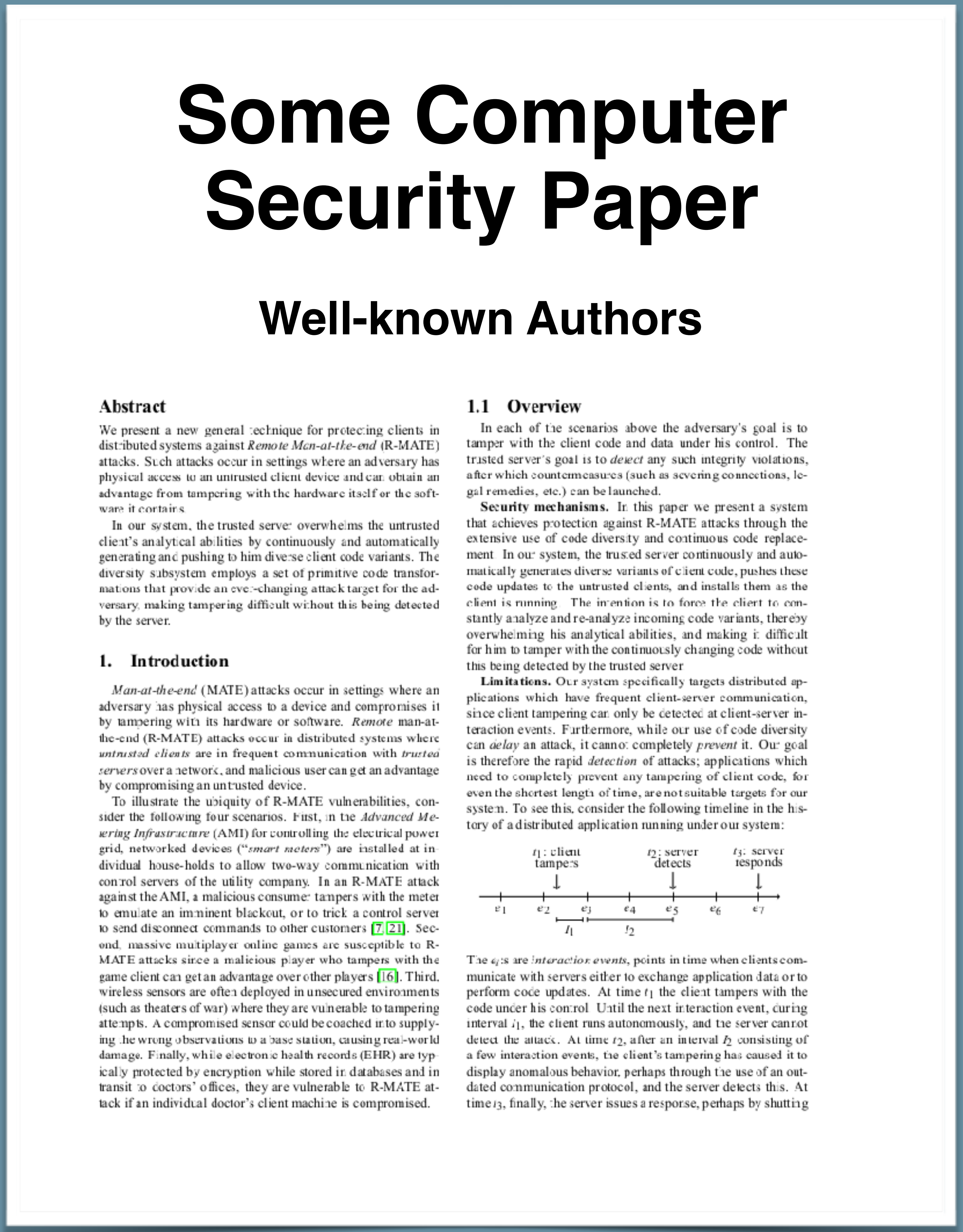
The system allows the user to specify (using a regular expression) the range of values for a particular field of a node that the user wishes to view.


CVS
version 6
GRAPH FORMAT
Internally Generated
GML File
DISPLAY OPTIONS
Interactive Display
On
Read GML Graphics
On
Background Color
White
Foreground Color
Black
Display Speed
Dimension
GRAPH OPTIONS
File Name
Split Components
On
ALGORITHM
GRIP
Fruchterman-Reingo
Filtration Technique
MIS
Number of Initial Vertices
4
Refinement Algorithm
Localize
Number of Refinement Levels
1
Number of Initial Rounds
20
Number of Final Rounds
20
Run
Reset
Help
Quit
./main
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
java.awt.ItemSelectable
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote
sandmark.obfuscate.varsplitter.VarSplitter


CVS
CANNOT BE REPLICATED
View JarFile
Class Sort
Method Sort
Instructions
CFG
Interference Graph
Statistics
Metrics
Slicing
Graph zoom
NODE INFORMATION
./main
GML File
Run
Quit
White
Black
Dimension
GRAPH OPTIONS
Filtration Technique
MIS
Number of Initial Vertices
4
Refinement Algorithm
Number of Refinement Levels
1
Number of Initial Rounds
20
Number of Final Rounds
20
sandmark.util.exec.TracingException
sandmark.util.graph.codec.DecodeFailure
sandmark.watermark.WatermarkingException
sandmark.obfuscate.ObfuscationException
java.lang.Exception
sandmark.gui.ObfTree
javax.swing.Scrollable
sandmark.gui.SkinPanel
sandmark.gui.VSortPanel
sandmark.obfuscate.degradation.promotion.DPromote

Why Should We Care?





Some Computer Security Paper

Well-known Authors

Abstract

We present a new general technique for protecting clients in distributed systems against *Remote Man-at-the-end* (R-MATE) attacks. Such attacks occur in settings where an adversary has physical access to an untrusted client device and can obtain an advantage from tampering with the hardware itself or the software it contains.

In our system, the trusted server overwhelms the untrusted client's analytical abilities by continuously and automatically generating and pushing to him diverse client code variants. The diversity subsystem employs a set of primitive code transformations that provide an ever-changing attack target for the adversary, making tampering difficult without this being detected by the server.

1. Introduction

Man-at-the-end (MATE) attacks occur in settings where an adversary has physical access to a device and compromises it by tampering with its hardware or software. *Remote* man-at-the-end (R-MATE) attacks occur in distributed systems where *untrusted clients* are in frequent communication with *trusted servers* over a network, and malicious user can get an advantage by compromising an untrusted device.

To illustrate the ubiquity of R-MATE vulnerabilities, consider the following four scenarios. First, in the *Advanced Metering Infrastructure* (AMI) for controlling the electrical power grid, networked devices ("*smart meters*") are installed at individual house-holds to allow two-way communication with control servers of the utility company. In an R-MATE attack against the AMI, a malicious consumer tampers with the meter to emulate an imminent blackout, or to trick a control server to send disconnect commands to other customers [7, 21]. Second, massive multiplayer online games are susceptible to R-MATE attacks since a malicious player who tampers with the game client can get an advantage over other players [16]. Third, wireless sensors are often deployed in unsecured environments (such as theaters of war) where they are vulnerable to tampering attempts. A compromised sensor could be coached into supplying the wrong observations to a base station, causing real-world damage. Finally, while electronic health records (EHR) are typically protected by encryption while stored in databases and in transit to doctors' offices, they are vulnerable to R-MATE attack if an individual doctor's client machine is compromised.

1.1 Overview

In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

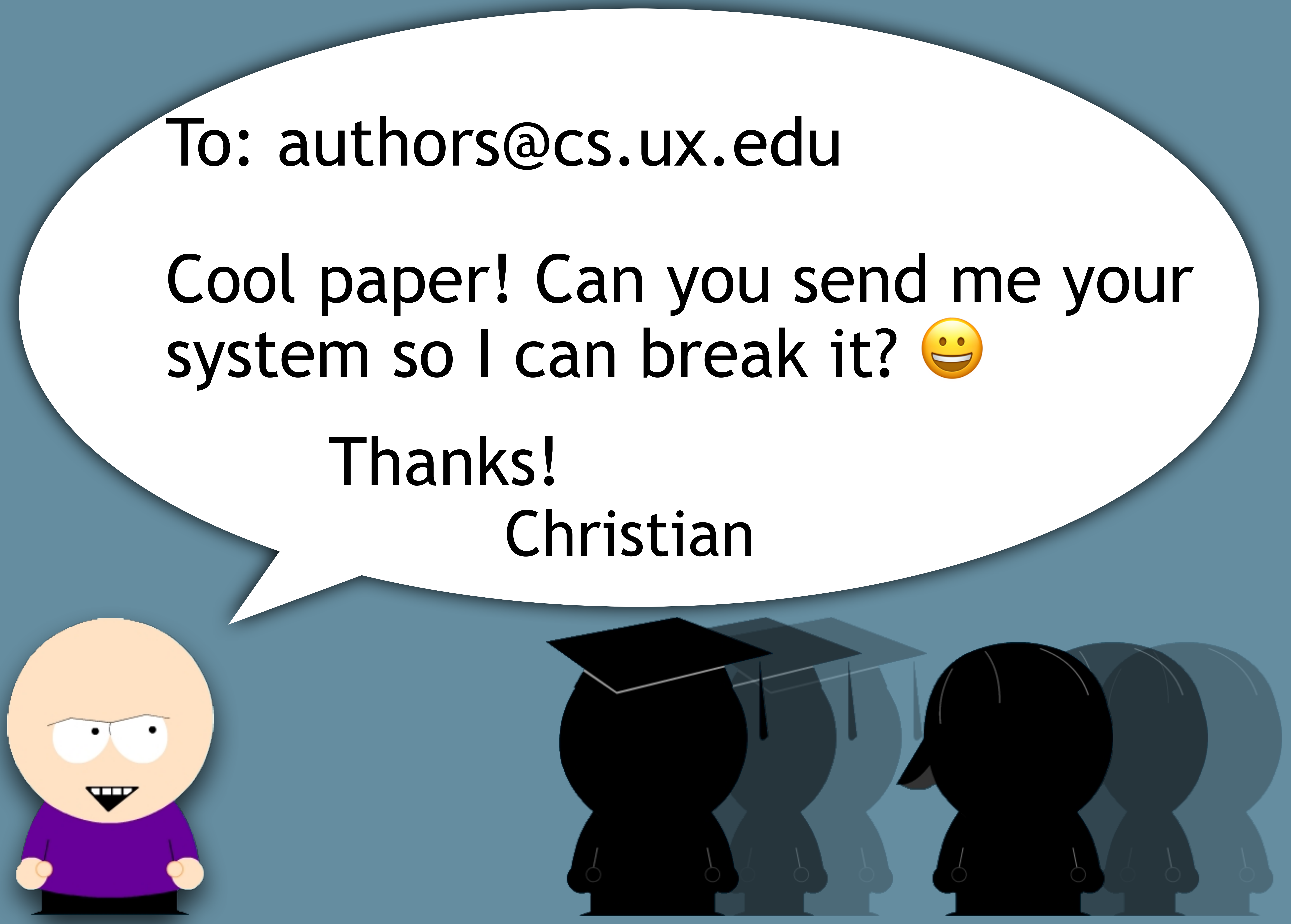
Security mechanisms. In this paper we present a system that achieves protection against R-MATE attacks through the extensive use of code diversity and continuous code replacement. In our system, the trusted server continuously and automatically generates diverse variants of client code, pushes these code updates to the untrusted clients, and installs them as the client is running. The intention is to force the client to constantly analyze and re-analyze incoming code variants, thereby overwhelming his analytical abilities, and making it difficult for him to tamper with the continuously changing code without this being detected by the trusted server.

Limitations. Our system specifically targets distributed applications which have frequent client-server communication, since client tampering can only be detected at client-server interaction events. Furthermore, while our use of code diversity can *delay* an attack, it cannot completely *prevent* it. Our goal is therefore the rapid *detection* of attacks; applications which need to completely prevent any tampering of client code, for even the shortest length of time, are not suitable targets for our system. To see this, consider the following timeline in the history of a distributed application running under our system:

t_1: client tampers
t_2: server detects
t_3: server responds
e_1 e_2 e_3 e_4 e_5 e_6 e_7
I_1 I_2

The e_i:s are *interaction events*, points in time when clients communicate with servers either to exchange application data or to perform code updates. At time t_1 the client tampers with the code under his control. Until the next interaction event, during interval I_1, the client runs autonomously, and the server cannot detect the attack. At time t_2, after an interval I_2 consisting of a few interaction events, the client's tampering has caused it to display anomalous behavior, perhaps through the use of an outdated communication protocol, and the server detects this. At time t_3, finally, the server issues a response, perhaps by shutting


tigress.cs.arizona.edu
Some Computer
Abstract
1. Introduction

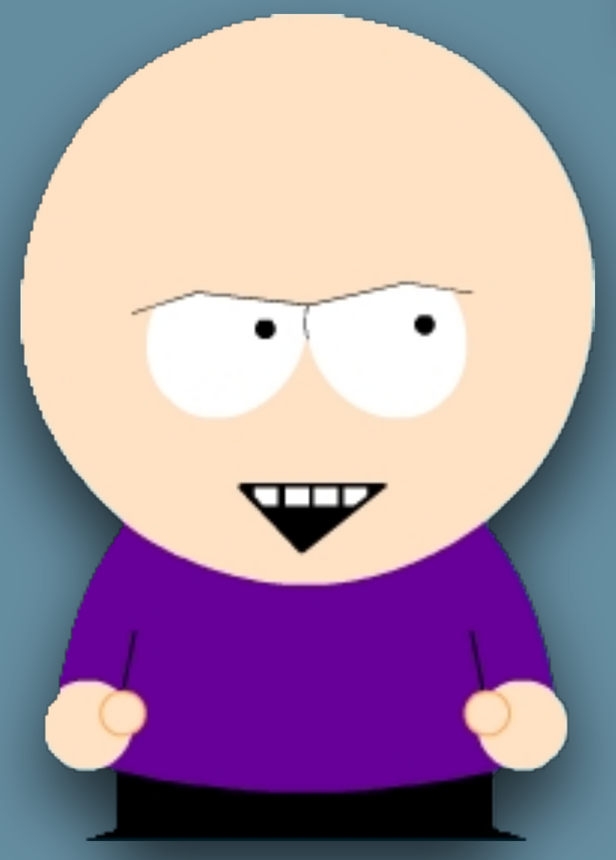

To: authors@cs.ux.edu
Cool paper! Can you send me your system so I can break it? 😀
Thanks!
Christian

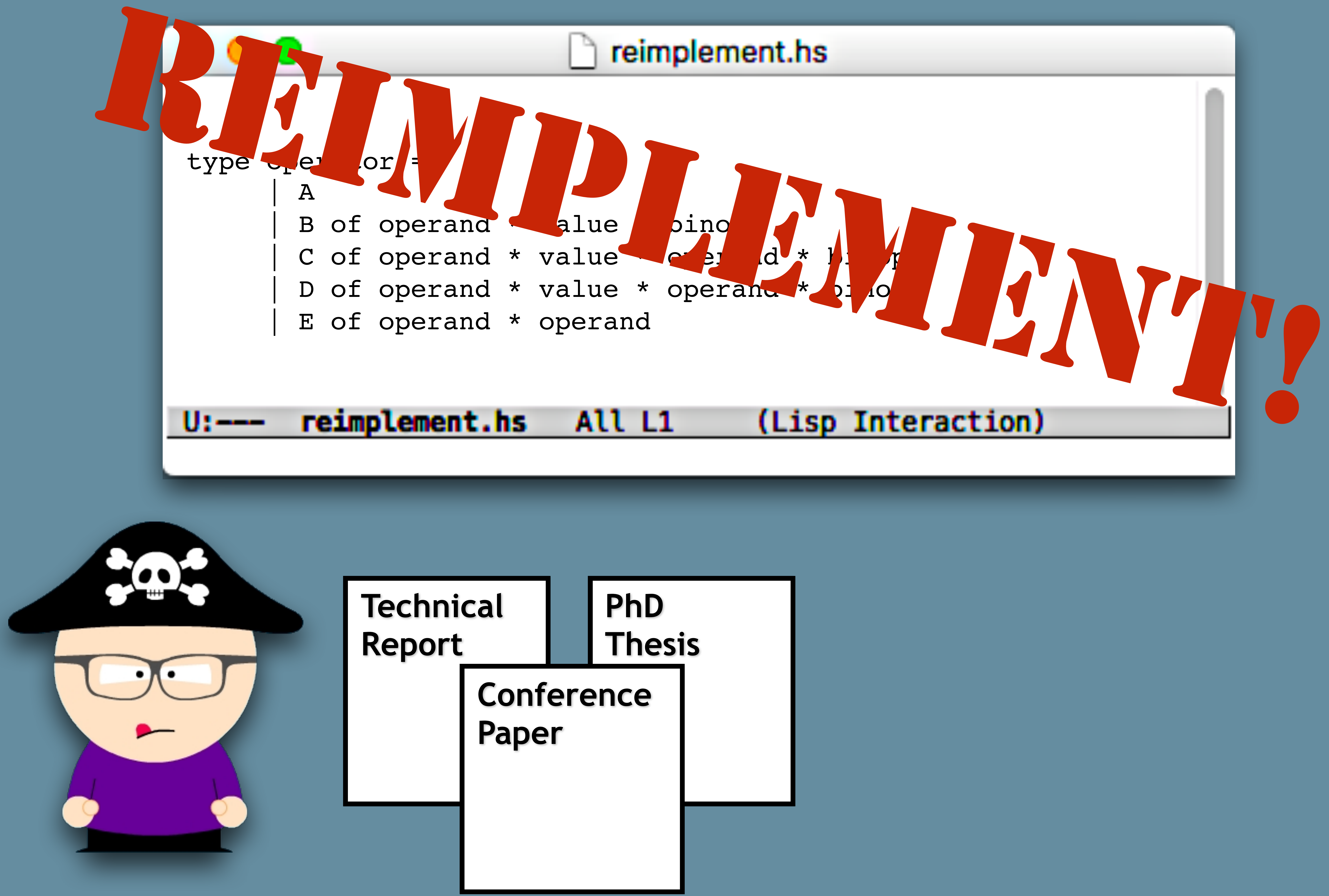
To: authors@cs.ux.edu

Cool paper! Can you send me your system so I can break it? 😀

Thanks!

Christian




reimplement.hs
REIMPLEMENT!
type
| A
| B of operand
| C of operand * value
| D of operand * value * operand *
| E of operand * operand
U:--- reimplement.hs All L1 (Lisp Interaction)
Technical Report
PhD Thesis
Conference Paper

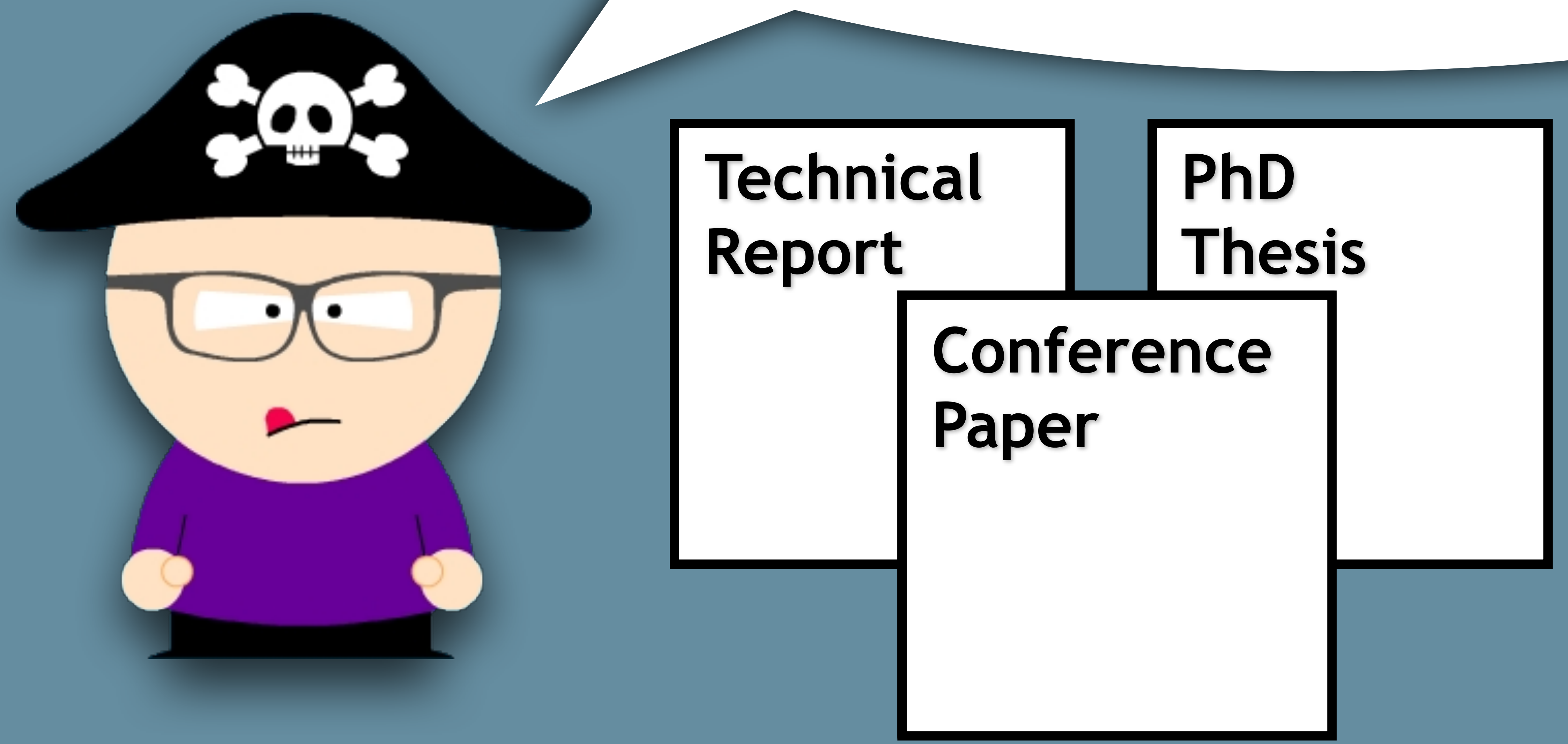
To: authors@cs.ux.edu

f: never used!

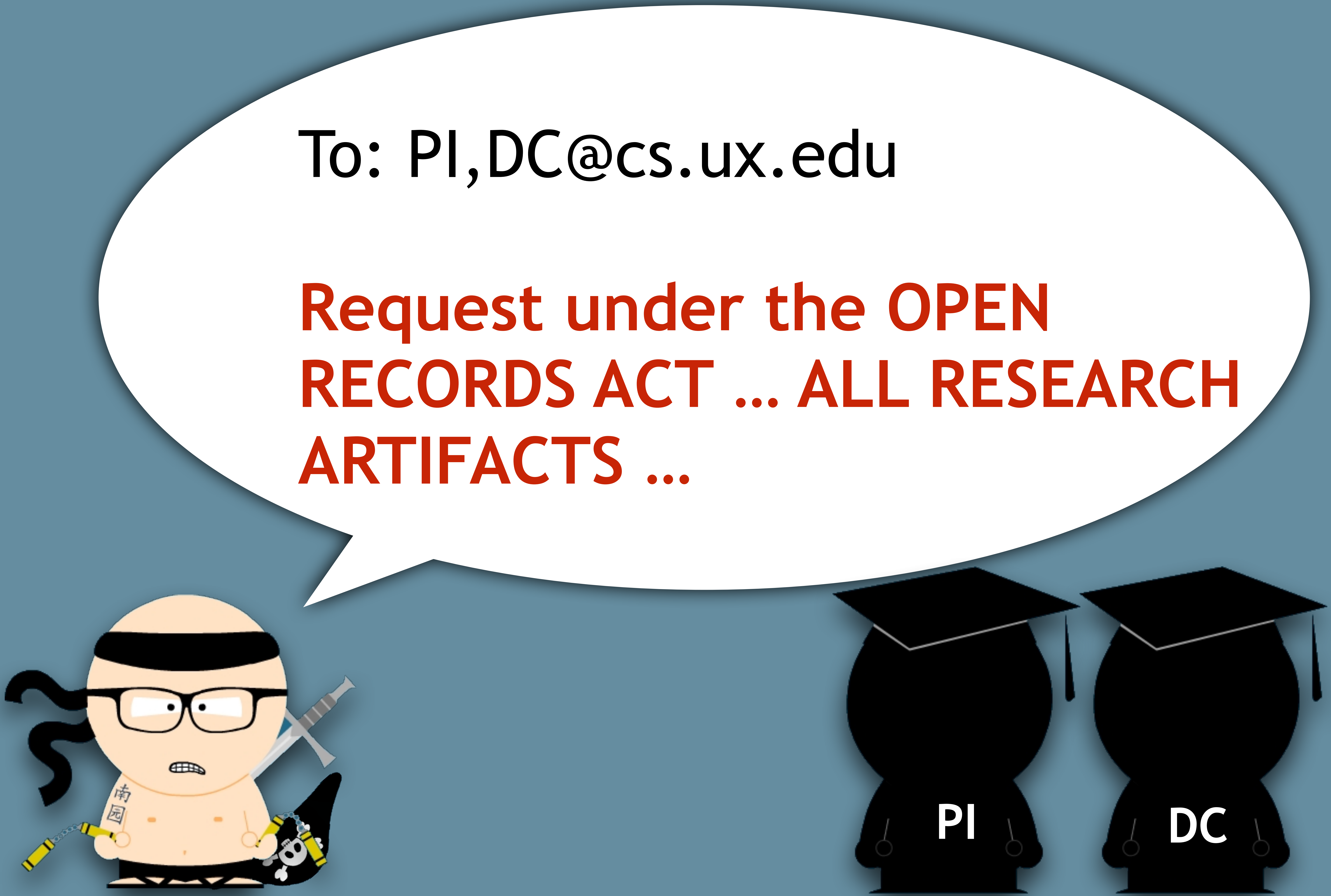
g: not defined!

h: doesn’t type check!

i: different in TR and paper!

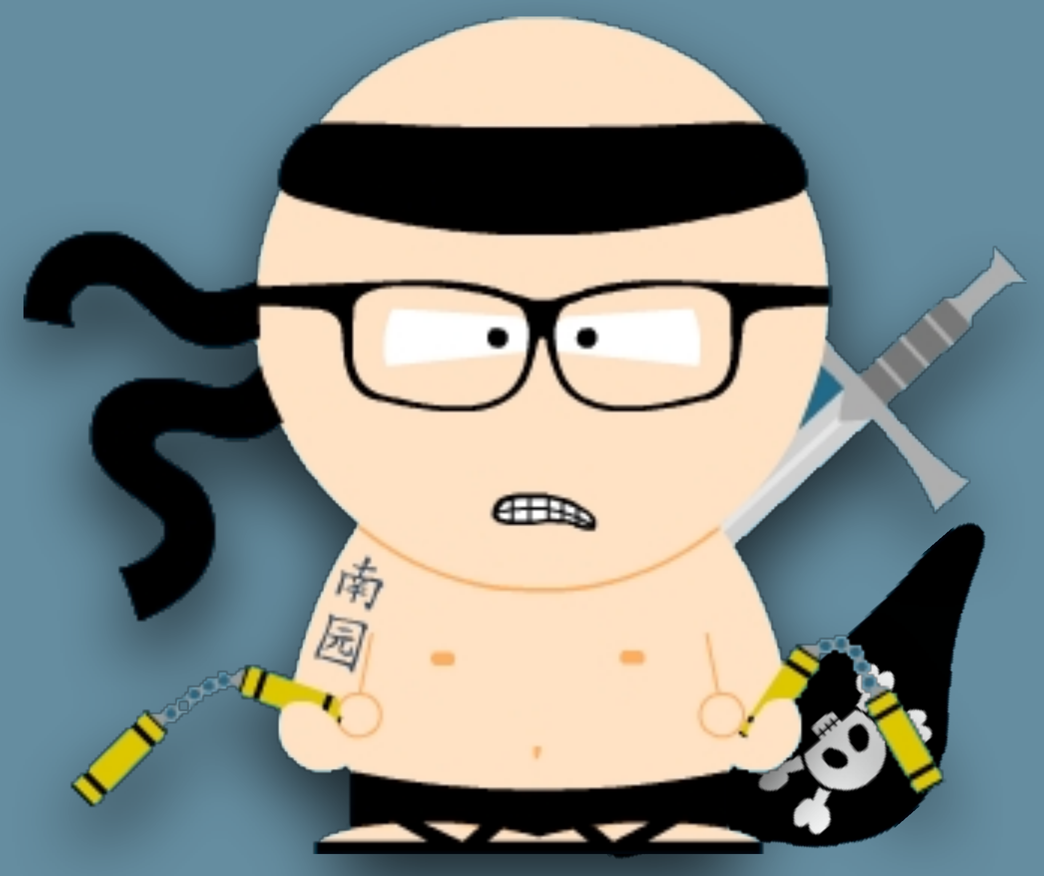


I … have few recollections of the work. [It was] like seeing a new paper for the first time.
Technical Report
PhD Thesis
Conference Paper
3rd


To: PI,DC@cs.ux.edu
Request under the OPEN RECORDS ACT … ALL RESEARCH ARTIFACTS …
PI
DC

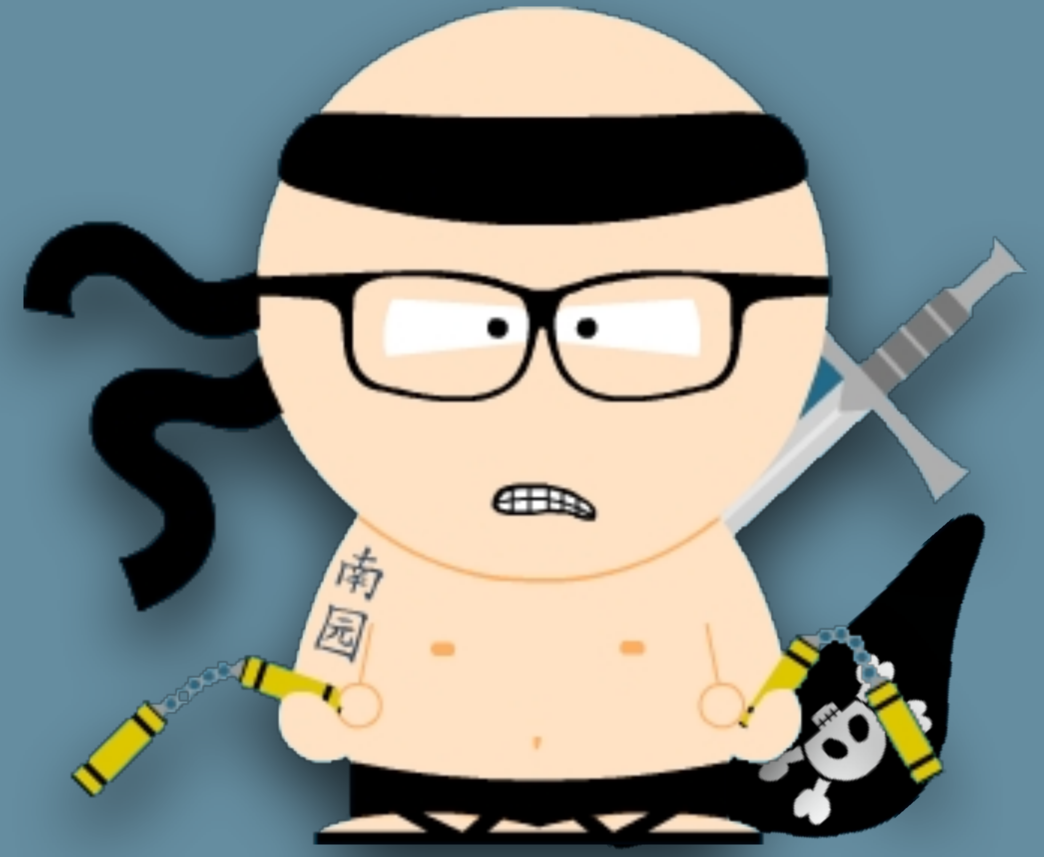
From: legal@cs.ux.edu

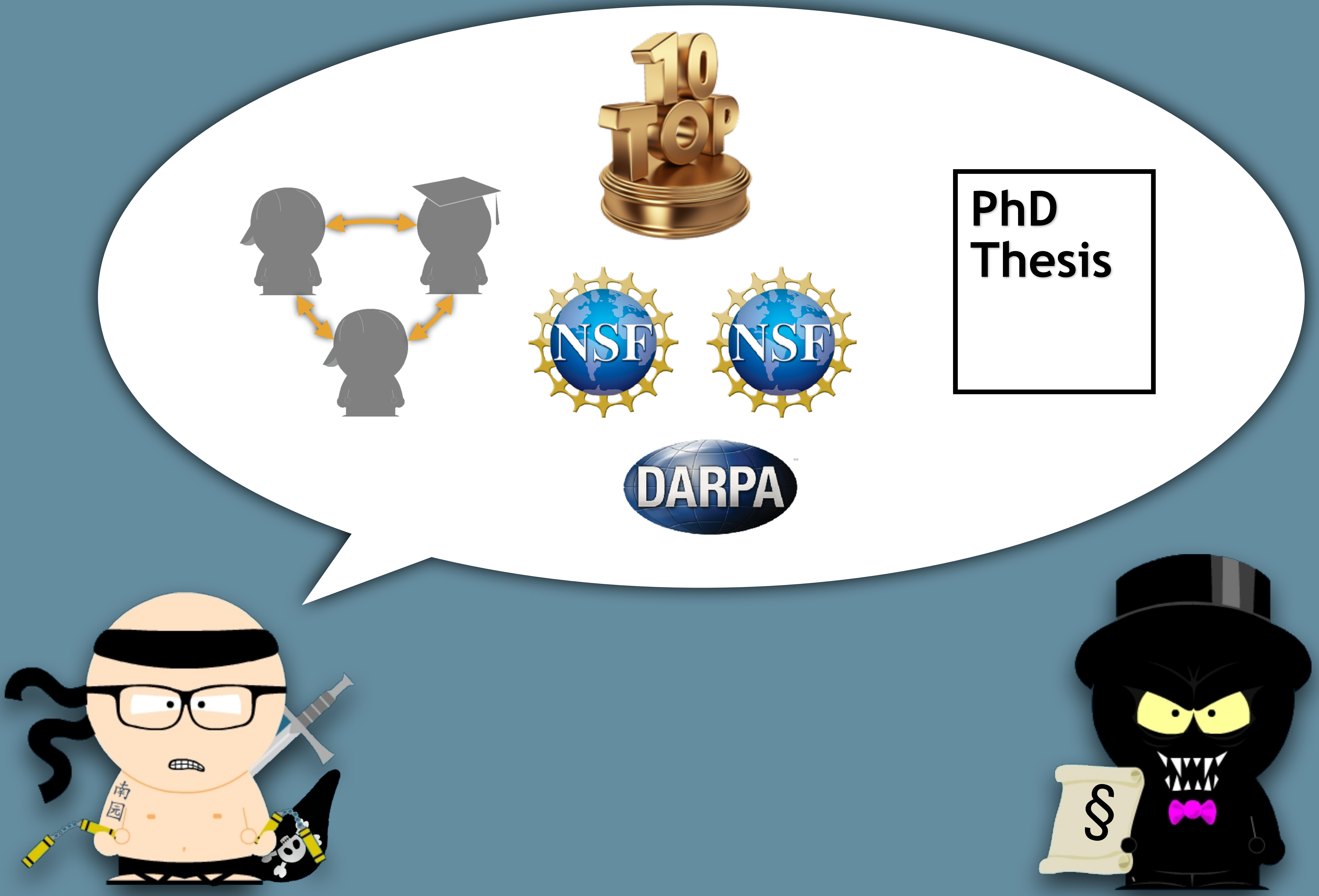
… to the extent such records may exist, **they will not be produced pursuant to ORA**.



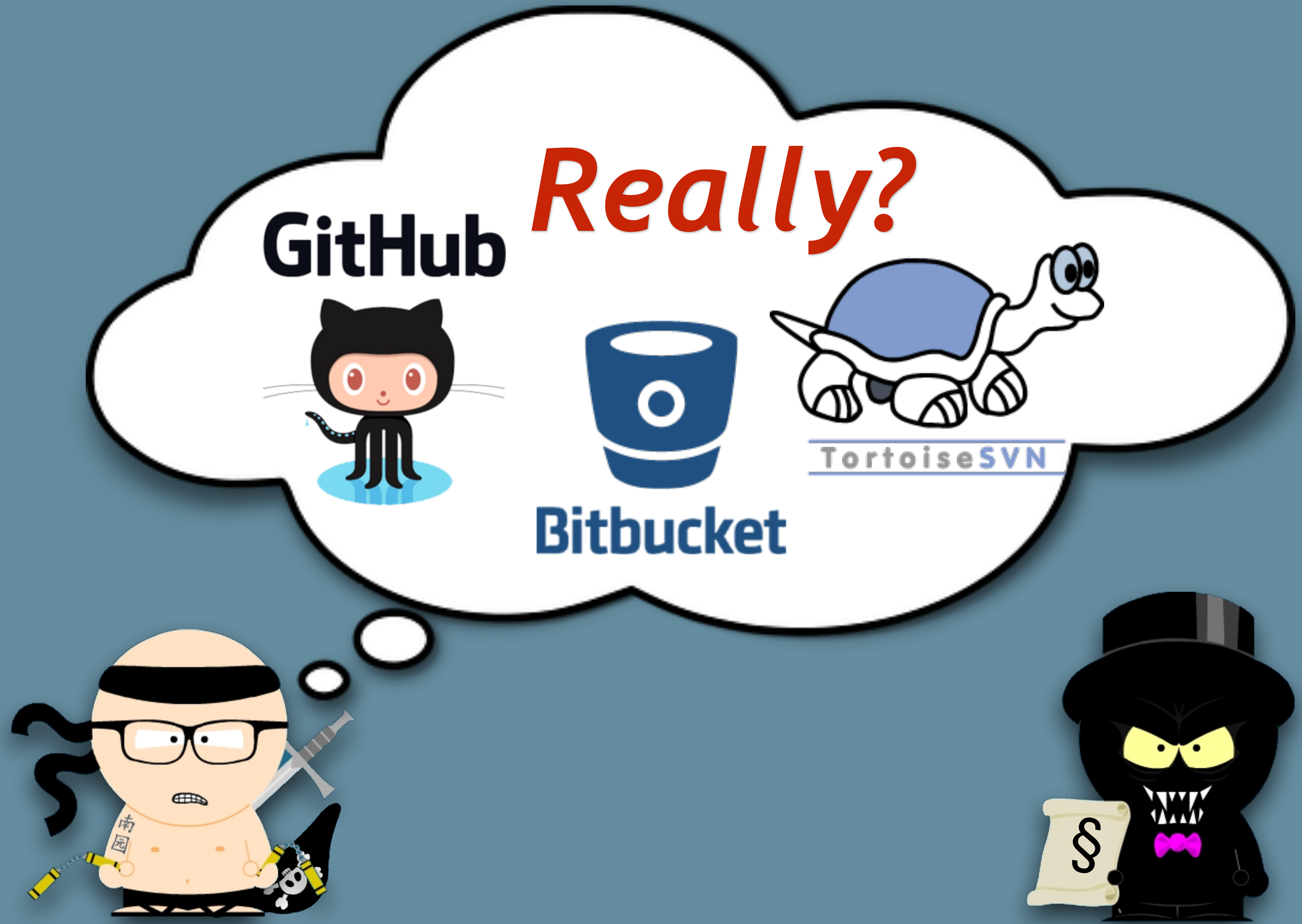
From: legal@cs.ux.edu

… and no, they don't exist…





10
TOP
NSF
NSF
DARPA
PhD
Thesis
§

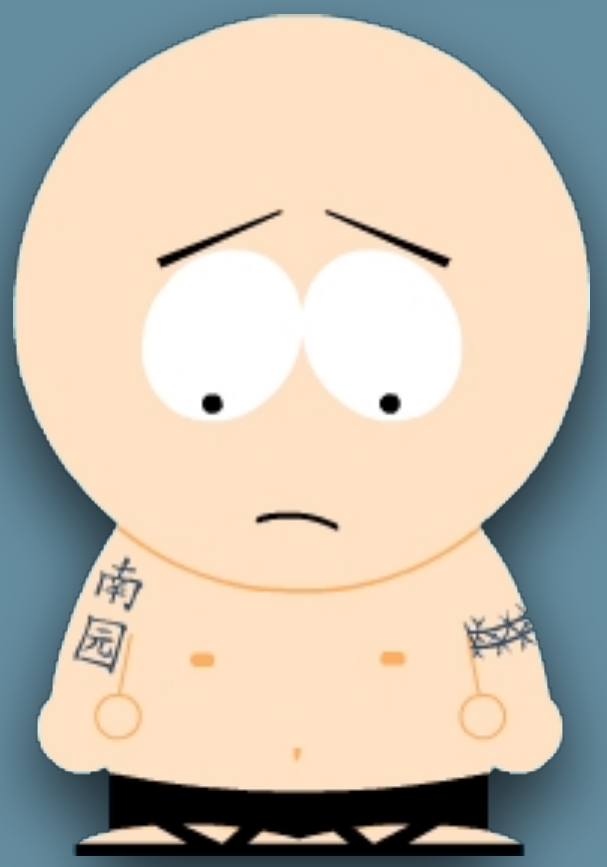

Really?
GitHub
Bitbucket
TortoiseSVN
§


Pursuant to ORA, I request copies of all electronic mail…
§


… a total cost of $2,263.66 to search for, retrieve, redact and produce such records.
§

Grant application #: [redacted]

We will also make our data and software available to the research community **when appropriate**.

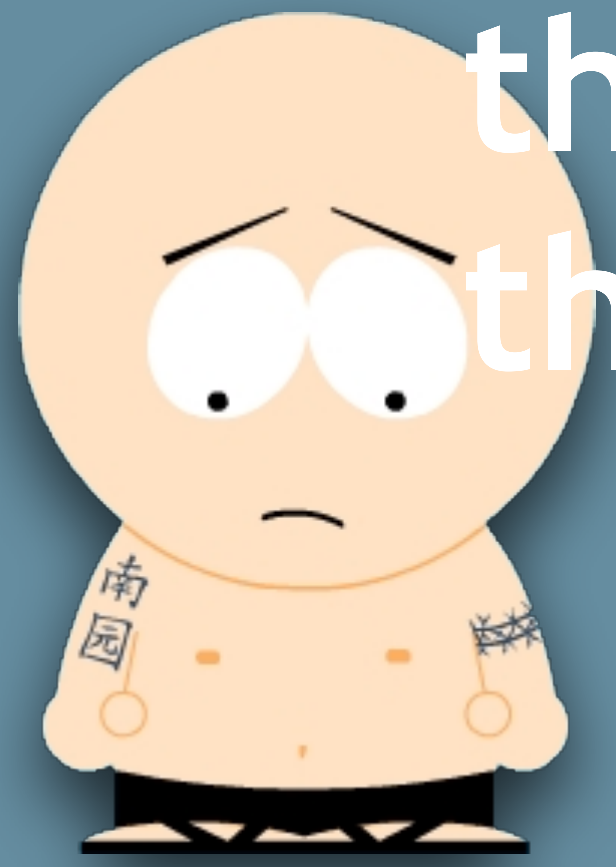


Consequences

By

- not sharing their artifacts,
- (perhaps unintentionally) leaving holes in their publications, and
- not responding to questions,

the authors have effectively guaranteed that their claims can never be refuted.



Consequences

By

- not sharing their artifacts,
- (perhaps unintentionally) leaving holes in their publications, and
- not responding to questions,

the authors have effectively guaranteed that their claims can never be refuted.

8th Law of Artifact Sharing

(Pretschner's Law)

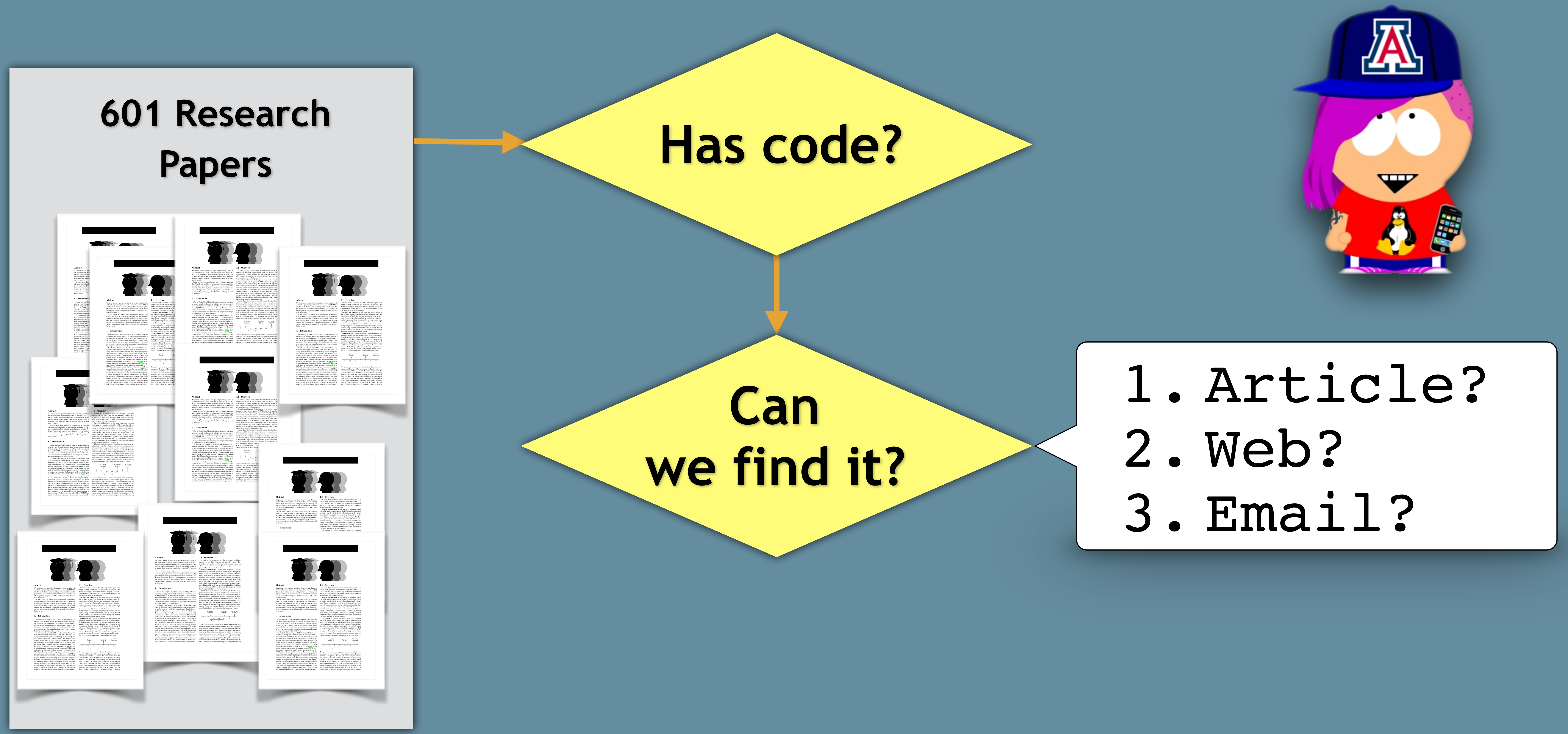
The probability of getting code out of someone is inversely proportional to the outrageousness of the claims in the paper.

The Deception Study

601 Research Papers

601 Research Papers

Has code?


601 Research Papers
Has code?
Can we find it?
1. Article?
2. Web?
3. Email?

601 Research Papers

Has code?

Can we find it?

1. Article?
2. Web?
3. Email?

Does it "work"?

1. ≤30 mins?
2. >30 mins?
3. Author?

601 Research Papers

Has code?

Can we find it?

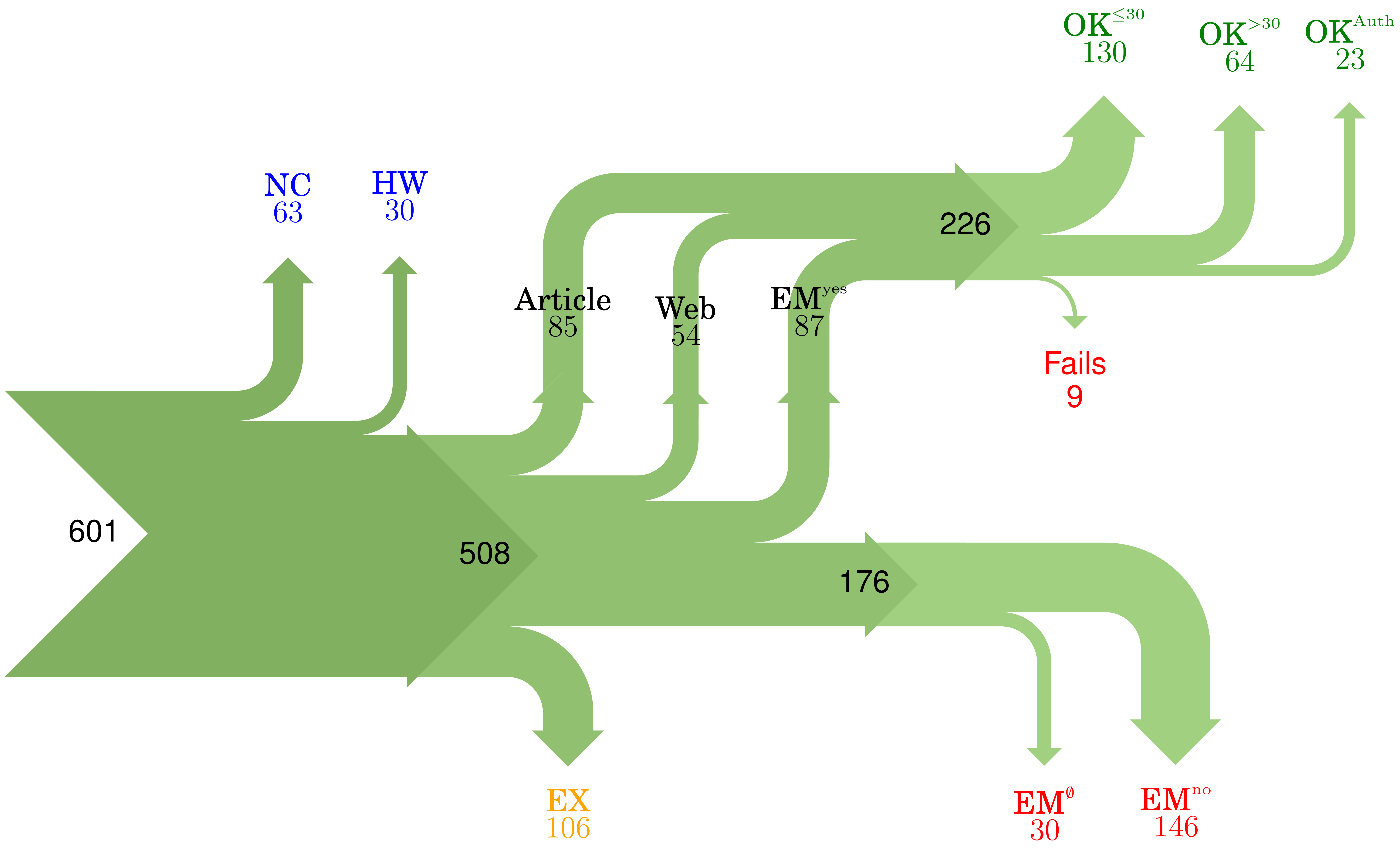
1. Article?
2. Web?
3. Email?

Does it "work"?

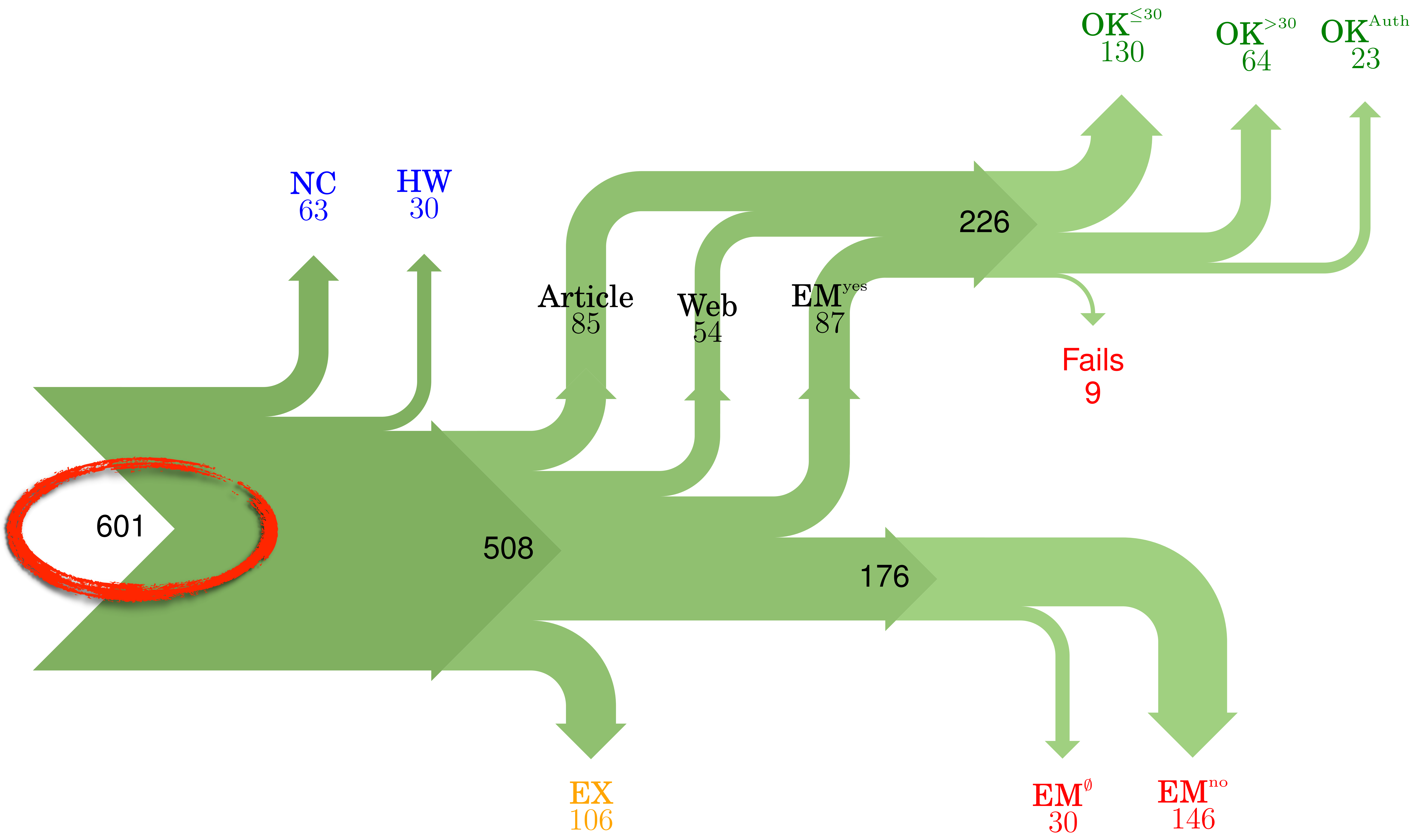
1. ≤30 mins?
2. >30 mins?
3. Author?

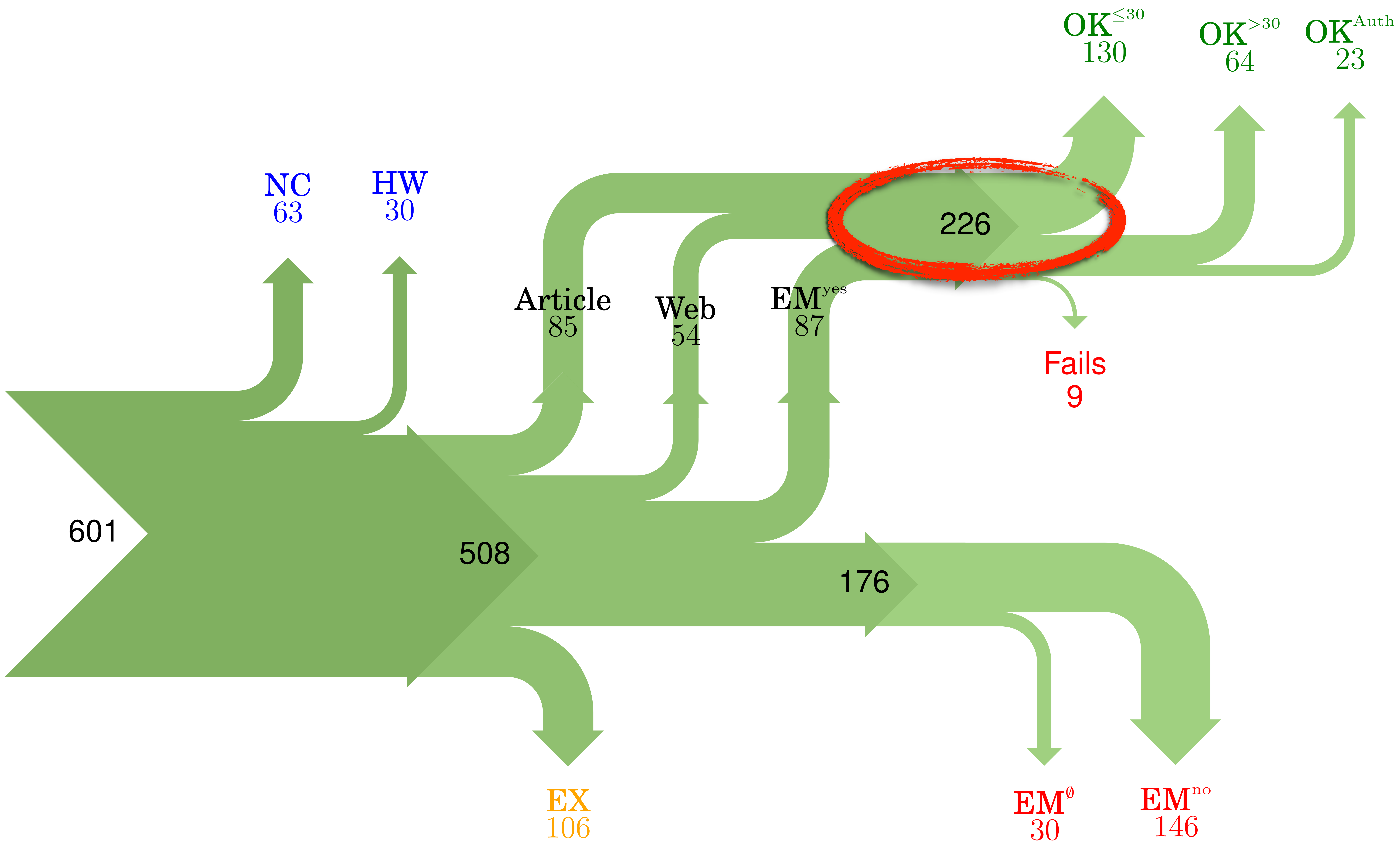
Weakly Repeatable

Authors share their code, and it builds.

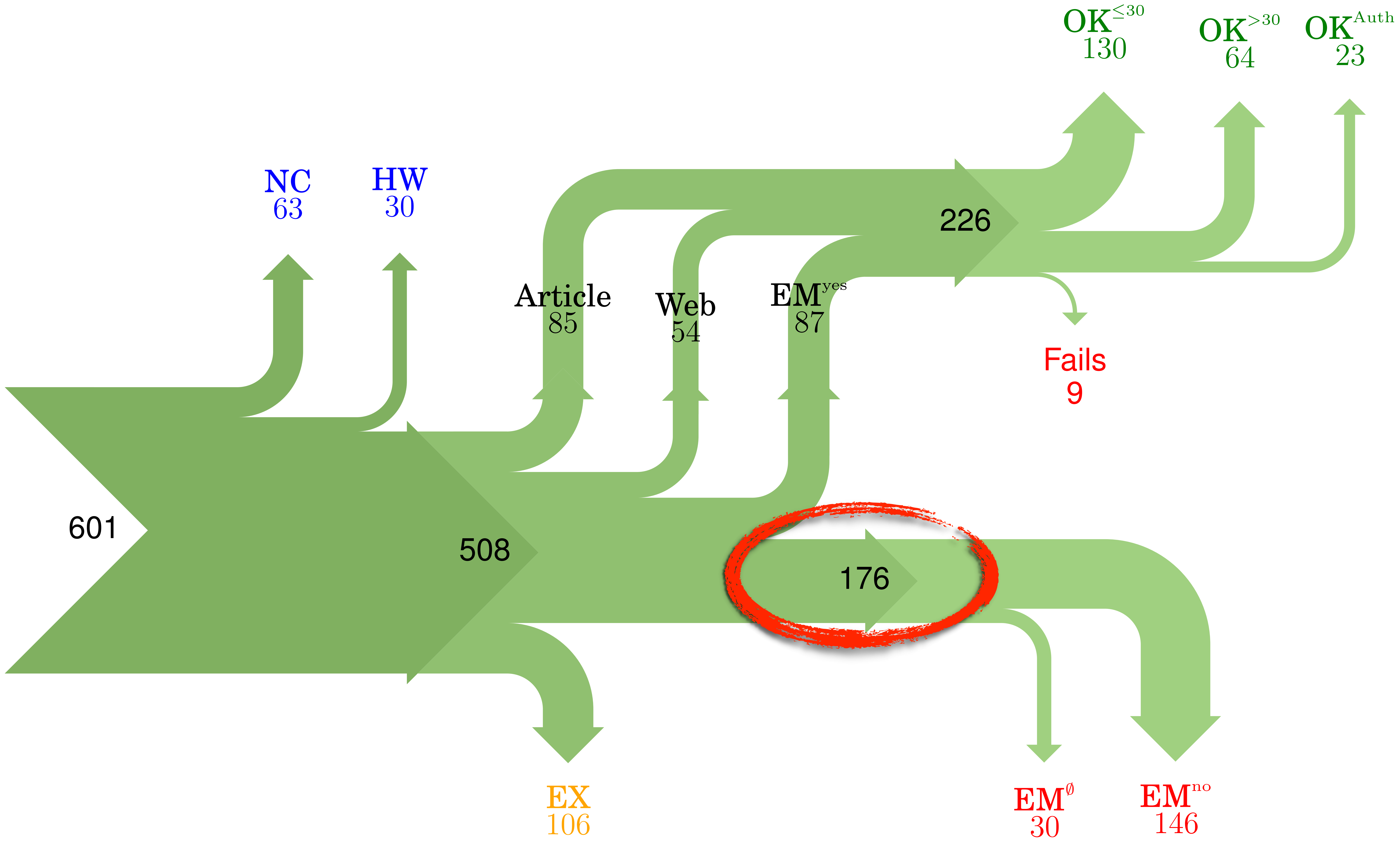


OK$^{\leq 30}$ 130
OK$^{>30}$ 64
OK^{Auth} 23
NC 63
HW 30
226
Article 85
Web 54
EM^{yes} 87
Fails 9
601
508
176
EX 106
EM$^{\emptyset}$ 30
EM^{no} 146


601
NC 63
HW 30
508
Article 85
Web 54
EM^yes 87
226
OK^≤30 130
OK^>30 64
OK^Auth 23
Fails 9
176
EX 106
EM^∅ 30
EM^no 146



$OK^{\leq 30}$
130
$OK^{>30}$
64
OK^{Auth}
23
NC
63
HW
30
226
Article
85
Web
54
EM^{yes}
87
Fails
9
601
508
176
EX
106
$EM^{\emptyset}$
30
EM^{no}
146



NC
63
HW
30
601
508
Article
85
Web
54
EM^yes
87
226
OK^≤30
130
OK^>30
64
OK^Auth
23
Fails
9
176
EX
106
EM^∅
30
EM^no
146

The good news … I was able to find some code. I am just **hoping** that it … **matches the implementation** we … used for the paper.

Versioning

Unfortunately the **current system is not mature** … We are actively working on a number of extensions … Soon …

Available Soon

[Our] prototype … included many moving pieces that only [student] knew how to operate … **he left**.

Personnel Issues

[Our] prototype … included many moving pieces that only [student] knew how to operate … **he left**.

Personnel Issues

… the server in which my implementation was stored had a **disk crash** … three disks crashed … Sorry for that.

Lost Code

… the server in which my implementation was stored had a **disk crash** … three disks crashed … Sorry for that.

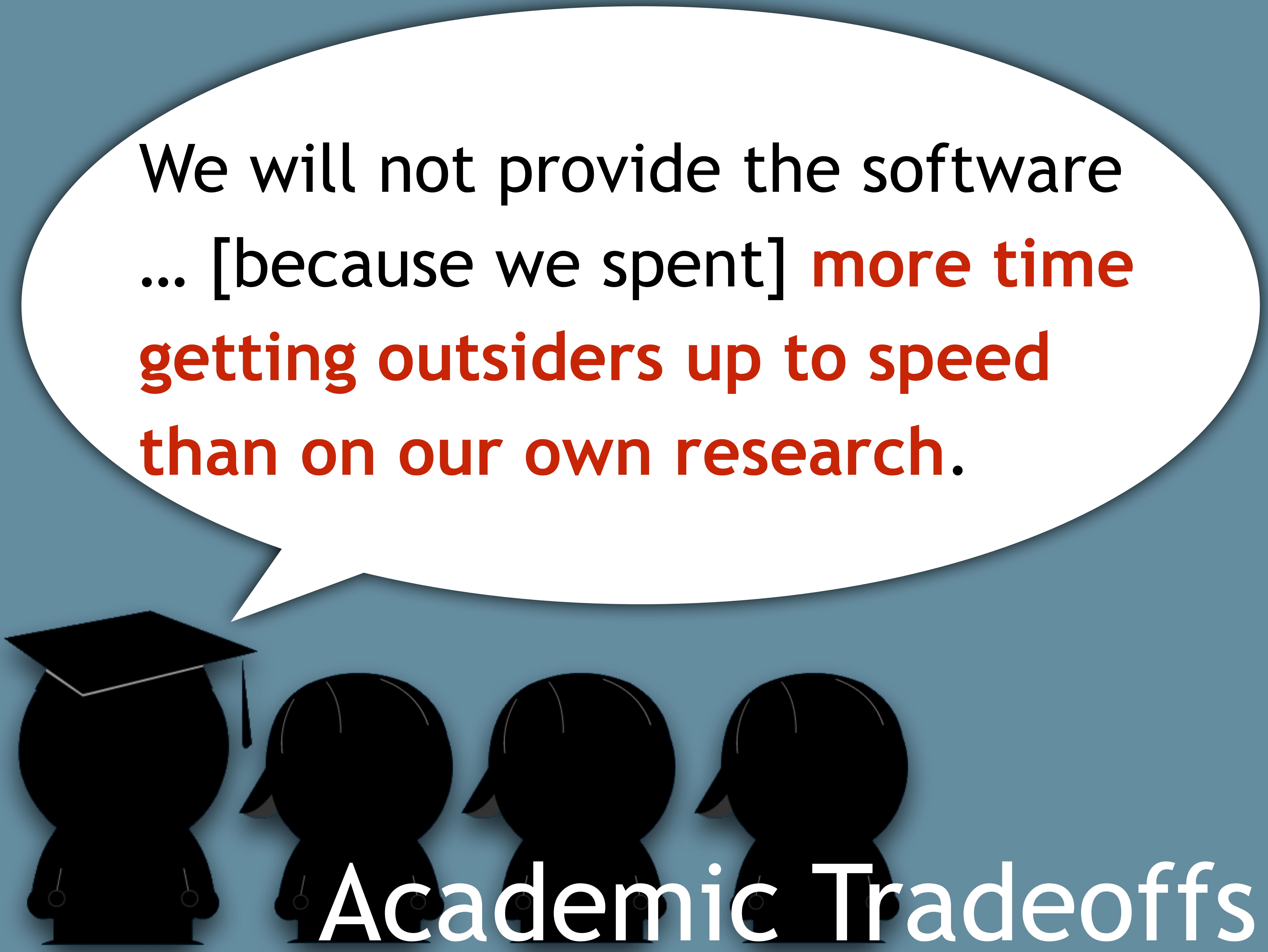
Lost Code

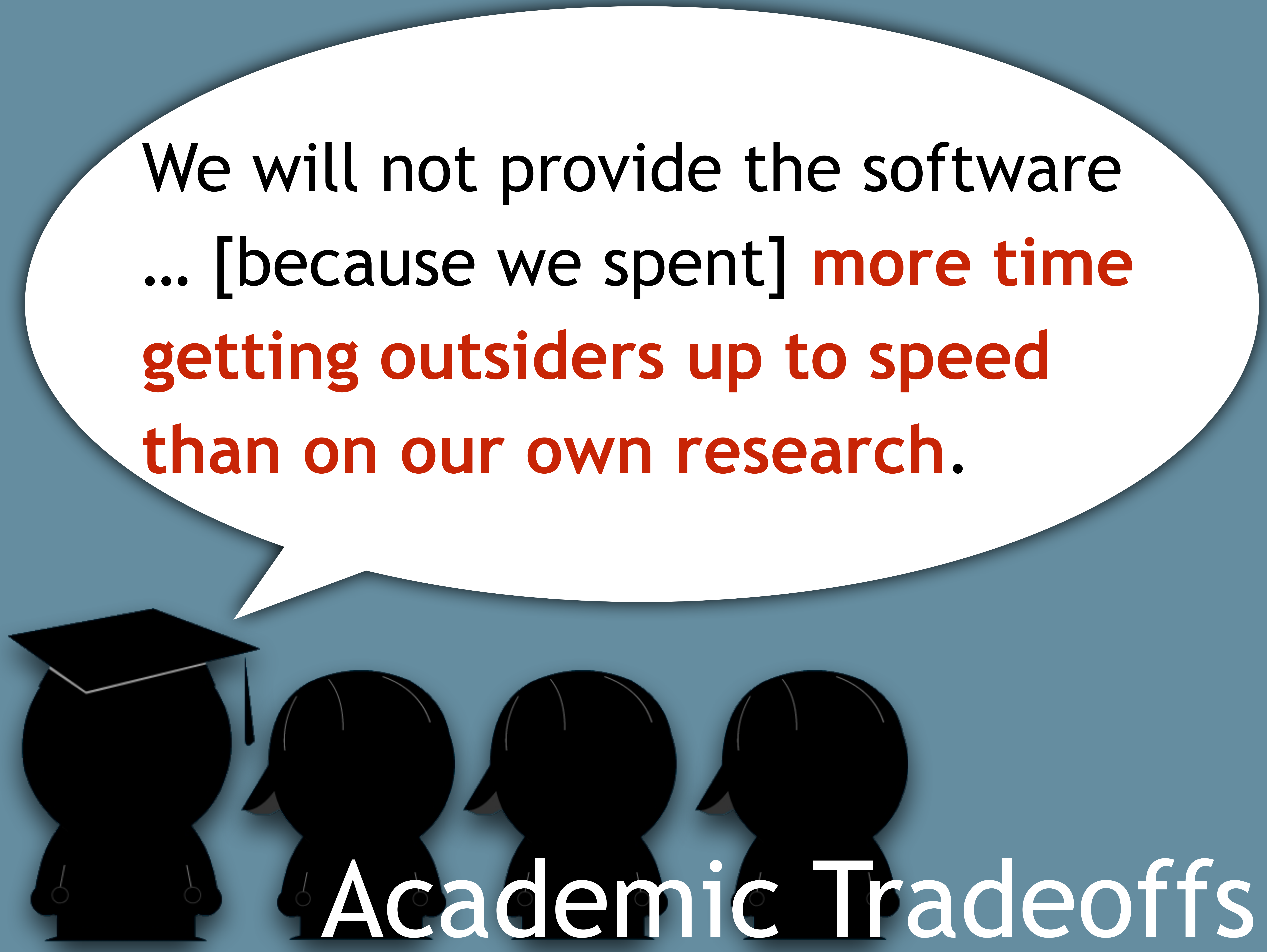
The code … is … **hardly usable by anyone** other than the authors … due to our decision to use [obscure variant of obscure language]

Design Issues

7th Law of Artifact Sharing
(Prepare to Share)

Unless a project starts with the express goal of post-publication artifact sharing, getting the right code, in a timely fashion, out of the project is virtually impossible.


We will not provide the software … [because we spent] **more time getting outsiders up to speed than on our own research**.
Academic Tradeoffs


We will not provide the software … [because we spent] more time getting outsiders up to speed than on our own research.
Academic Tradeoffs

… **we can't share what did for this paper**. … this is not in the academic tradition, but this is a hazard in an industrial lab.

Industrial Lab Tradeoffs

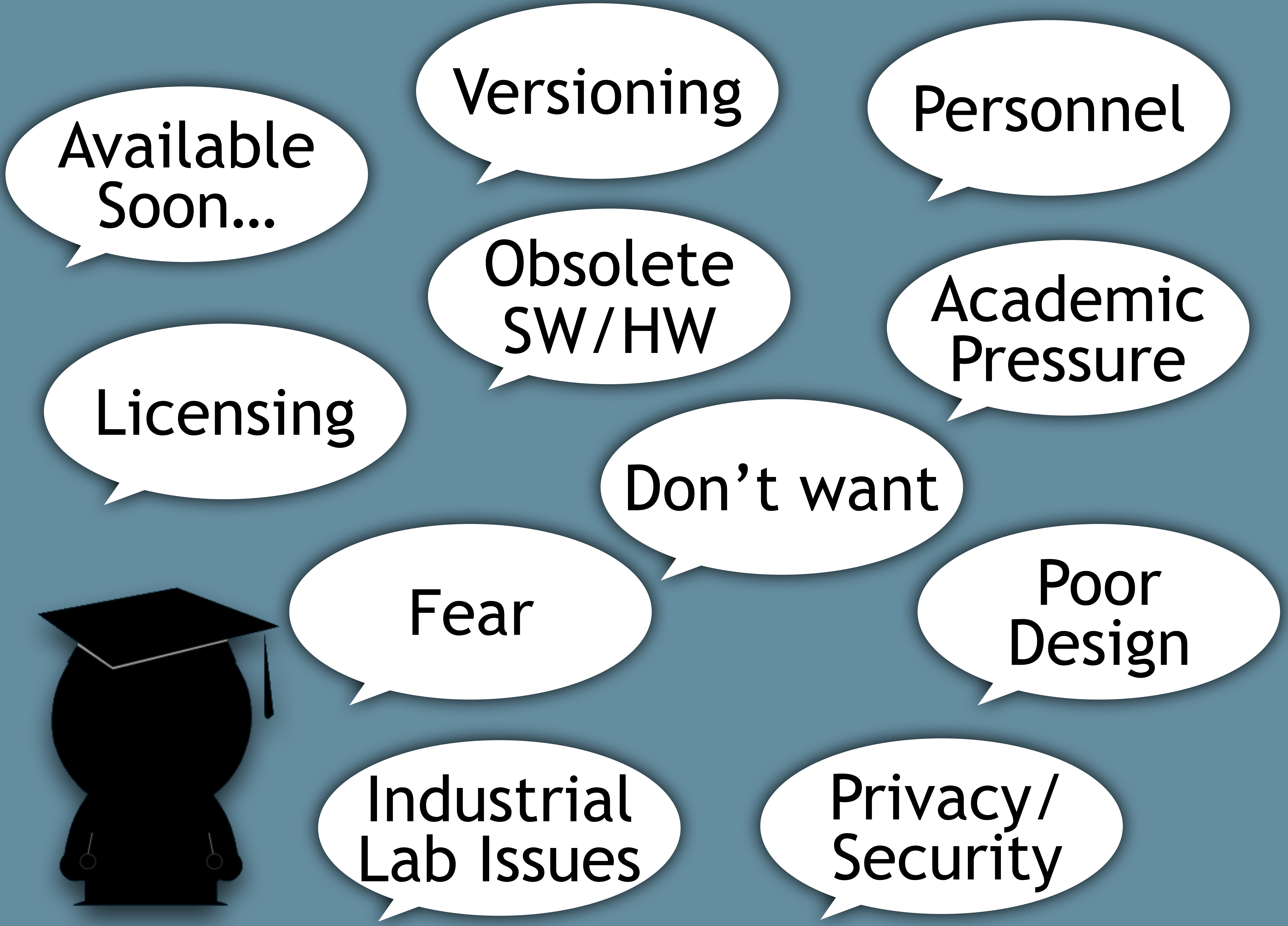
We have no plans to make the scheduler's source code publicly available … because [ancient OS] as such **does not exist anymore**.

Obsolete SW/HW

We have an agreement with the [business], and we cannot release the code because of the potential **privacy risks** …

Privacy/Security


Fear


Versioning
Personnel
Available Soon…
Obsolete SW/HW
Academic Pressure
Licensing
Don’t want
Fear
Poor Design
Industrial Lab Issues
Privacy/ Security

Sharing Proposal — #1 —

Artifact Curation

Sharing Proposal

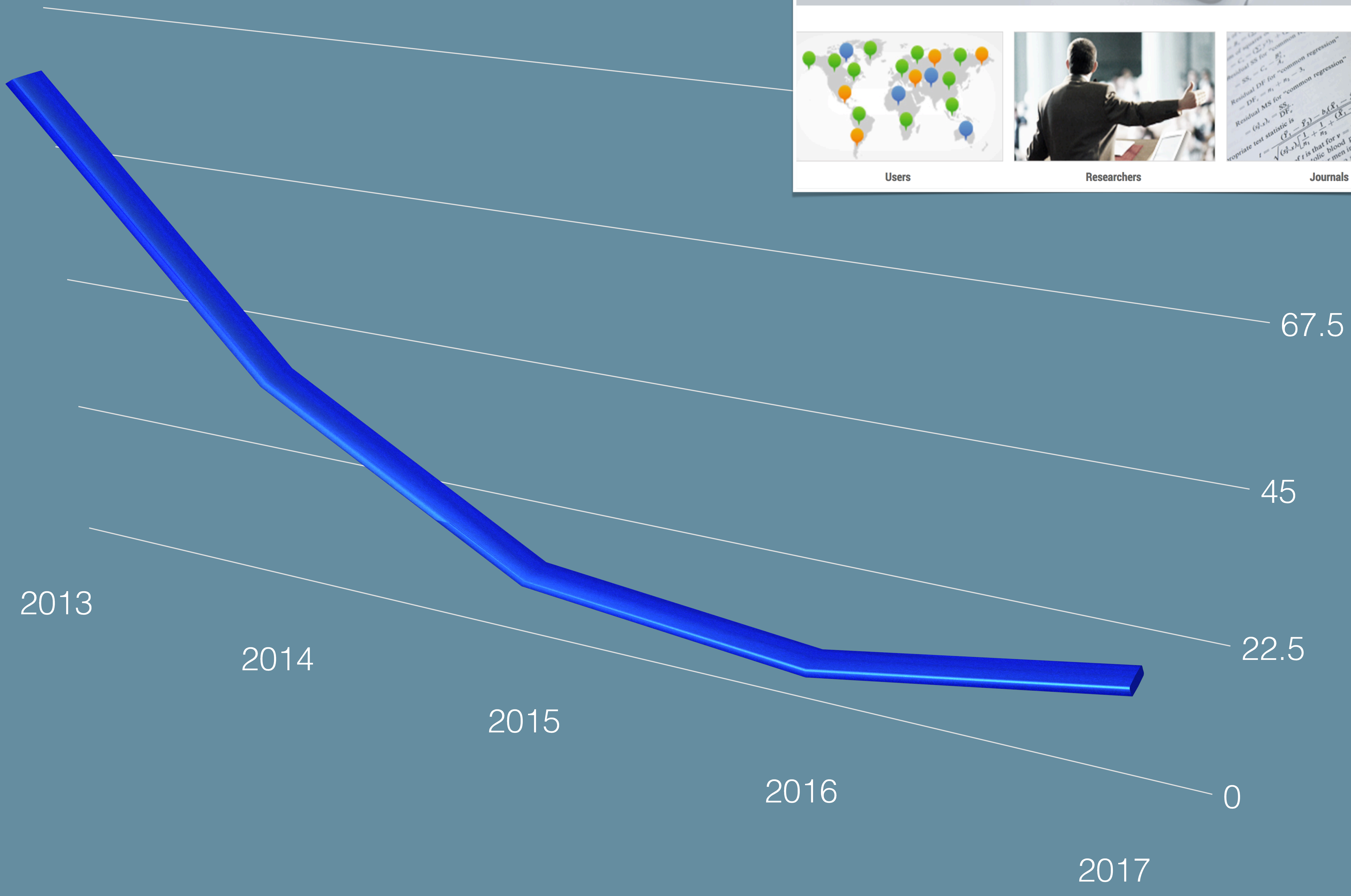
— #1 —

Artifact Curation

TECHNO-FIX


Search
Register | Sign In
runmycode
Home
Our goals
Researchers
Journal Editors
FAQ
Our partners
News
RunMyCode enables scientists to openly share the code and data that underlie their research publications
This service is based on the innovative concept of a companion website associated with a scientific publication.
Share
Your email
Create a password
SIGN UP FOR RUNMYCODE
Users
Researchers
Journals


Search
Register | Sign In
runmycode
Home
Our goals
Researchers
Journal Editors
FAQ
Our partners
News
RunMyCode enables scientists to openly share the code and data that underlie their research publications
This service is based on the innovative concept of a companion website associated with a scientific publication.
Share
Your email
Create a password
SIGN UP FOR RUNMYCODE
Users
Researchers
Journals


67.5
45
22.5
0
2013
2014
2015
2016
2017

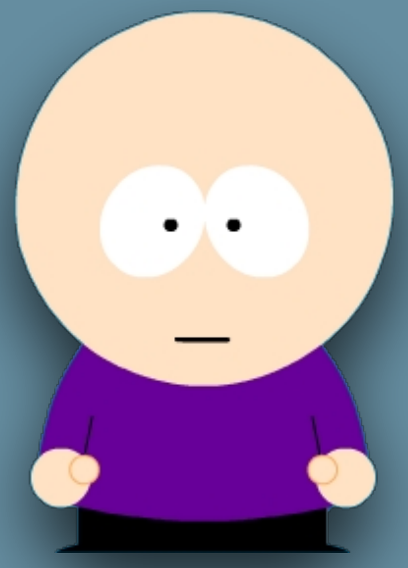
FindResearch.org

search.org FAQ Privacy policy Contact

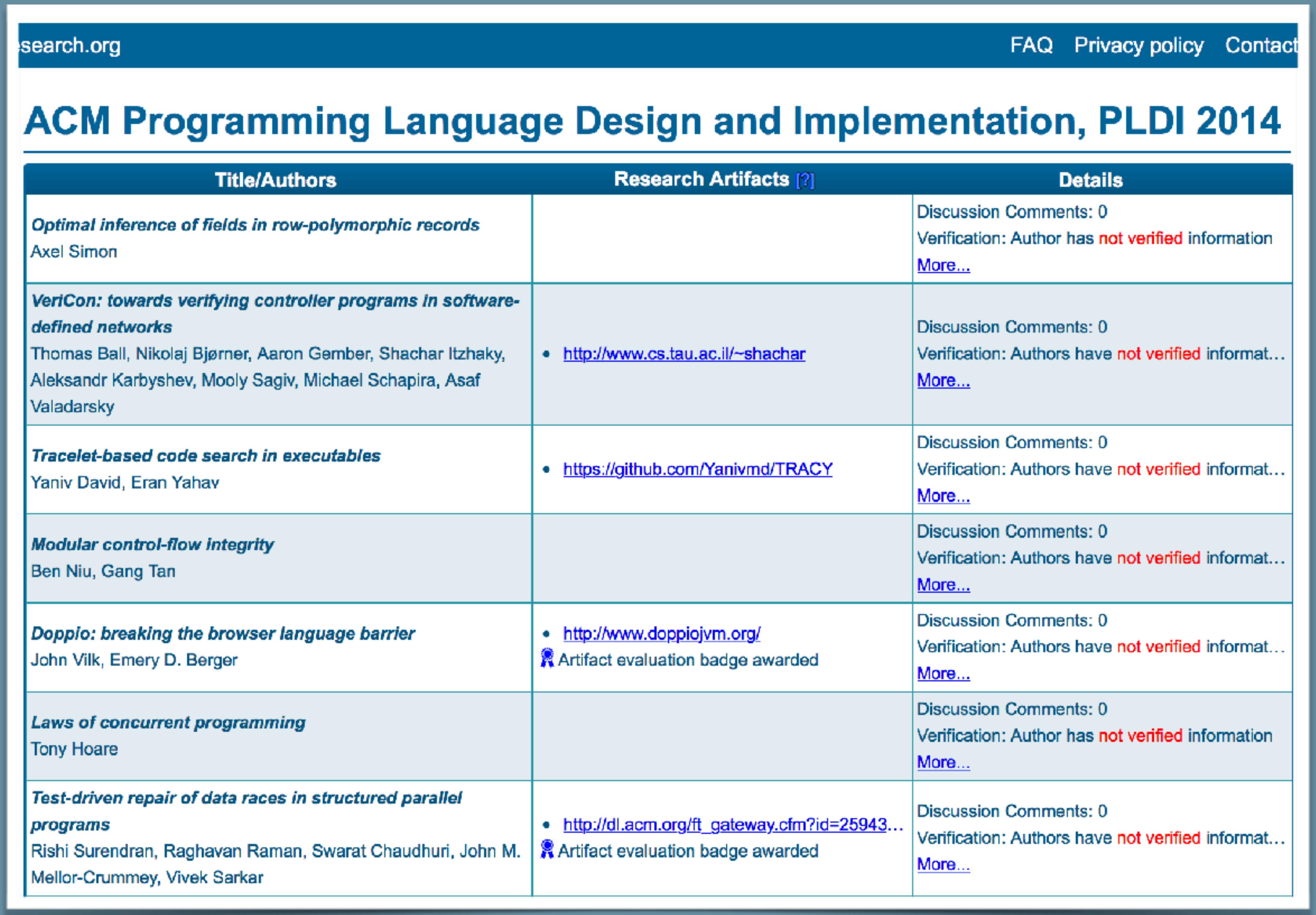
ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

1. Help the public find artifacts
2. Motivate researchers to share



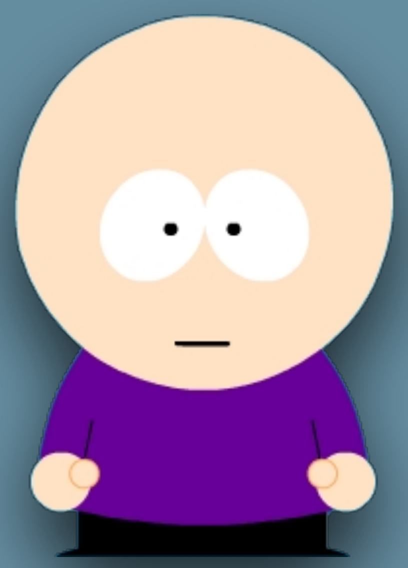
FindResearch.org



search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...



Find Artifacts
Emails, Grants

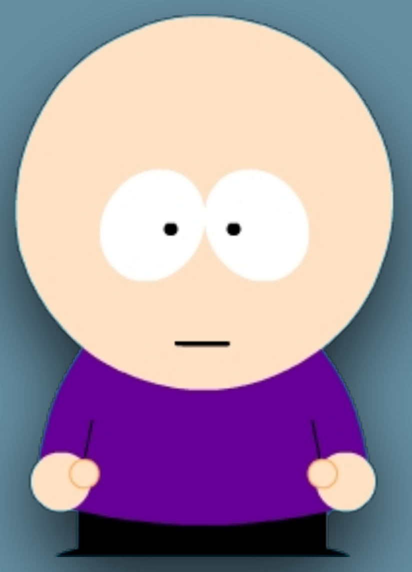
FindResearch.org

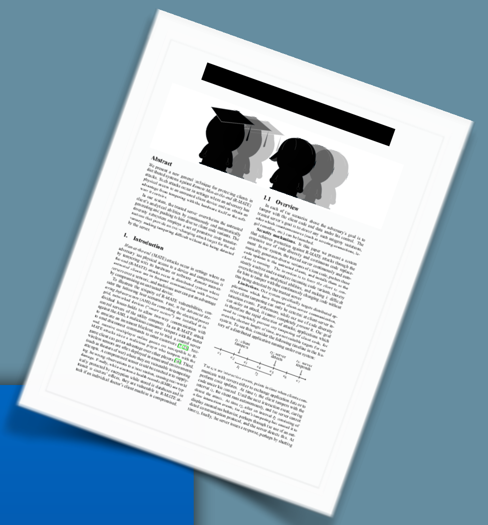
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

FindResearch.org





Find Artifacts
Emails, Grants

Verify
Information

search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

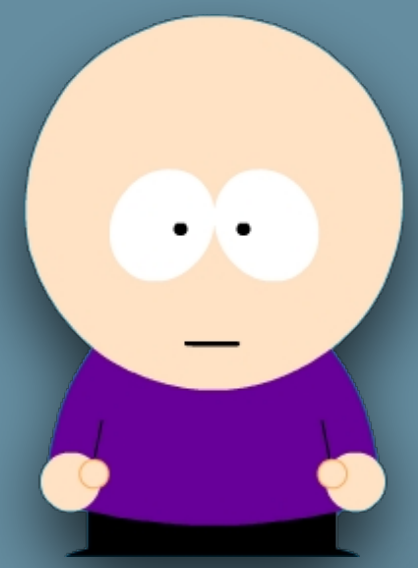
FindResearch.org

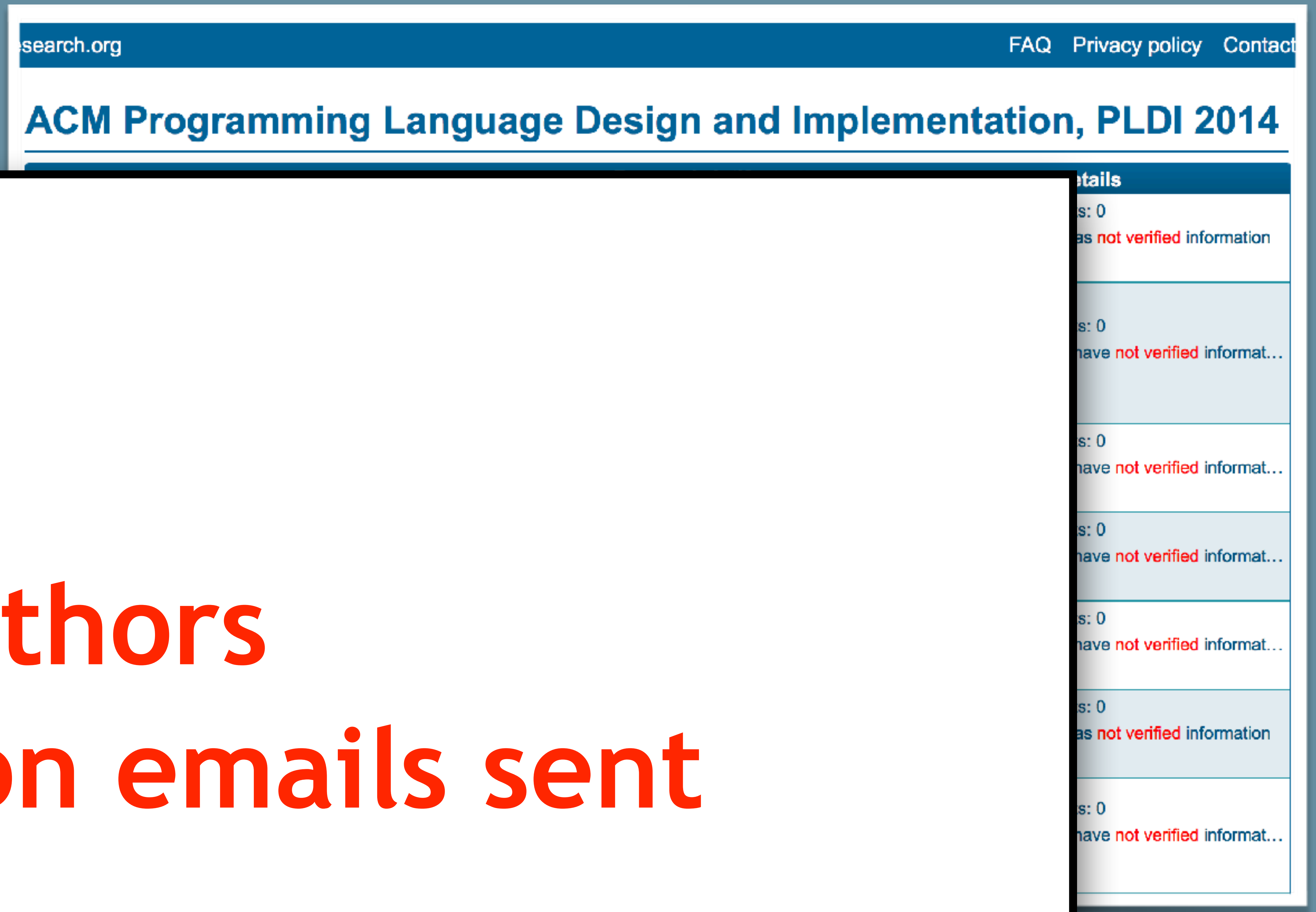
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

FindResearch.org





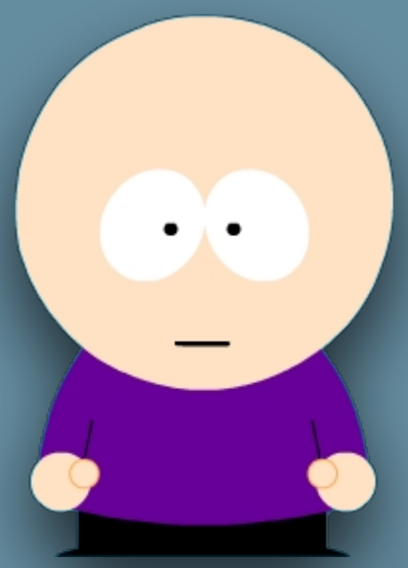
- 237 conferences
- 20,000 articles
- 39,000 unique authors
- 67,000 verification emails sent

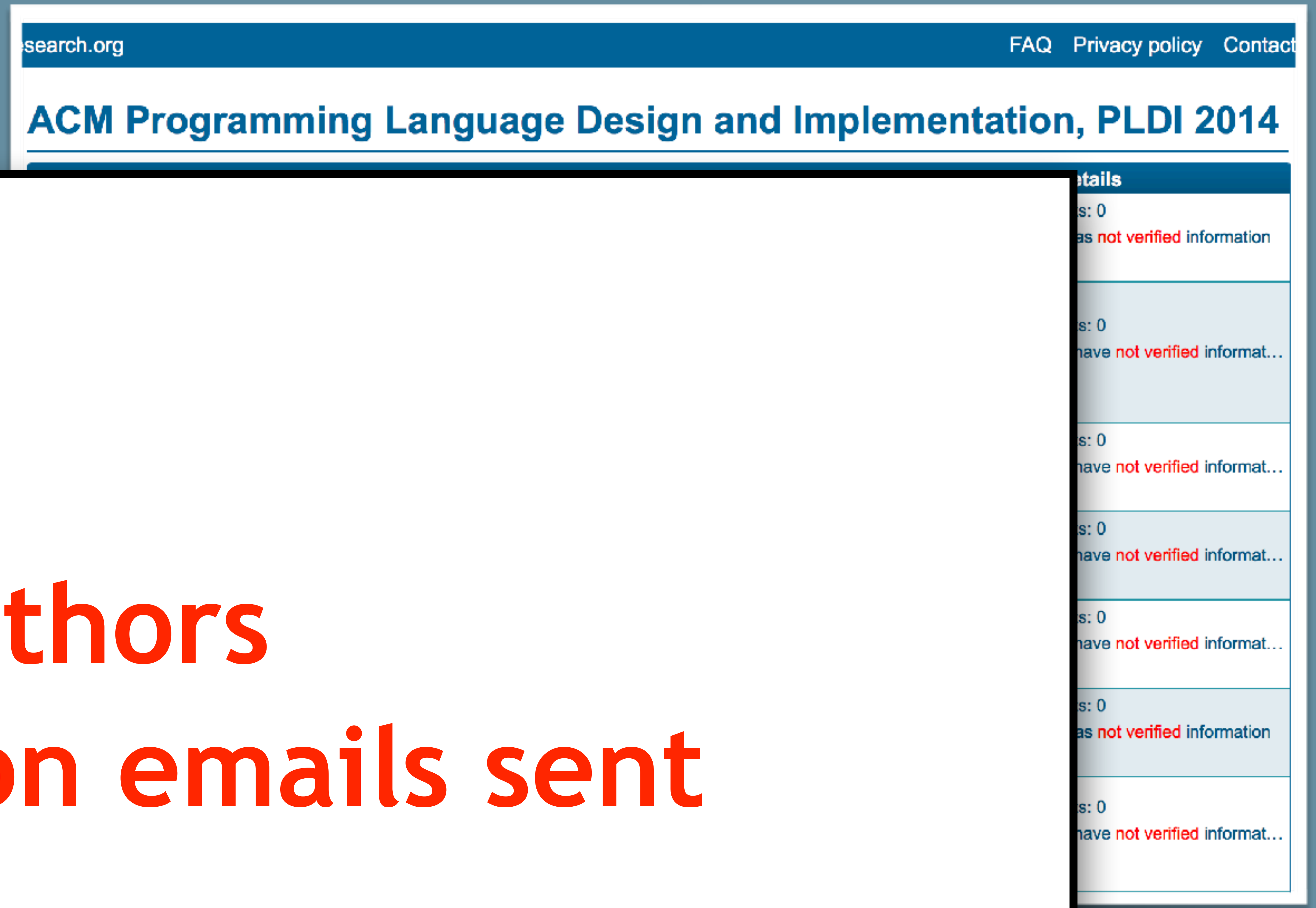
Fi
En

Ve
Int

tifacts
o share

FindResearch.org





- 237 conferences
- 20,000 articles
- 39,000 unique authors
- 67,000 verification emails sent

- 10% of articles are verified
- 6% of articles have shared artifacts


dblp
org
olicy Contact
2014
Ve
Int
artifacts
tifacts
o share
not verified informat...
STARBUCKS CARD
$5

6th Law of Artifact Sharing

(Inverse Costner's Law)

Even if you built it,
they still wouldn't come.

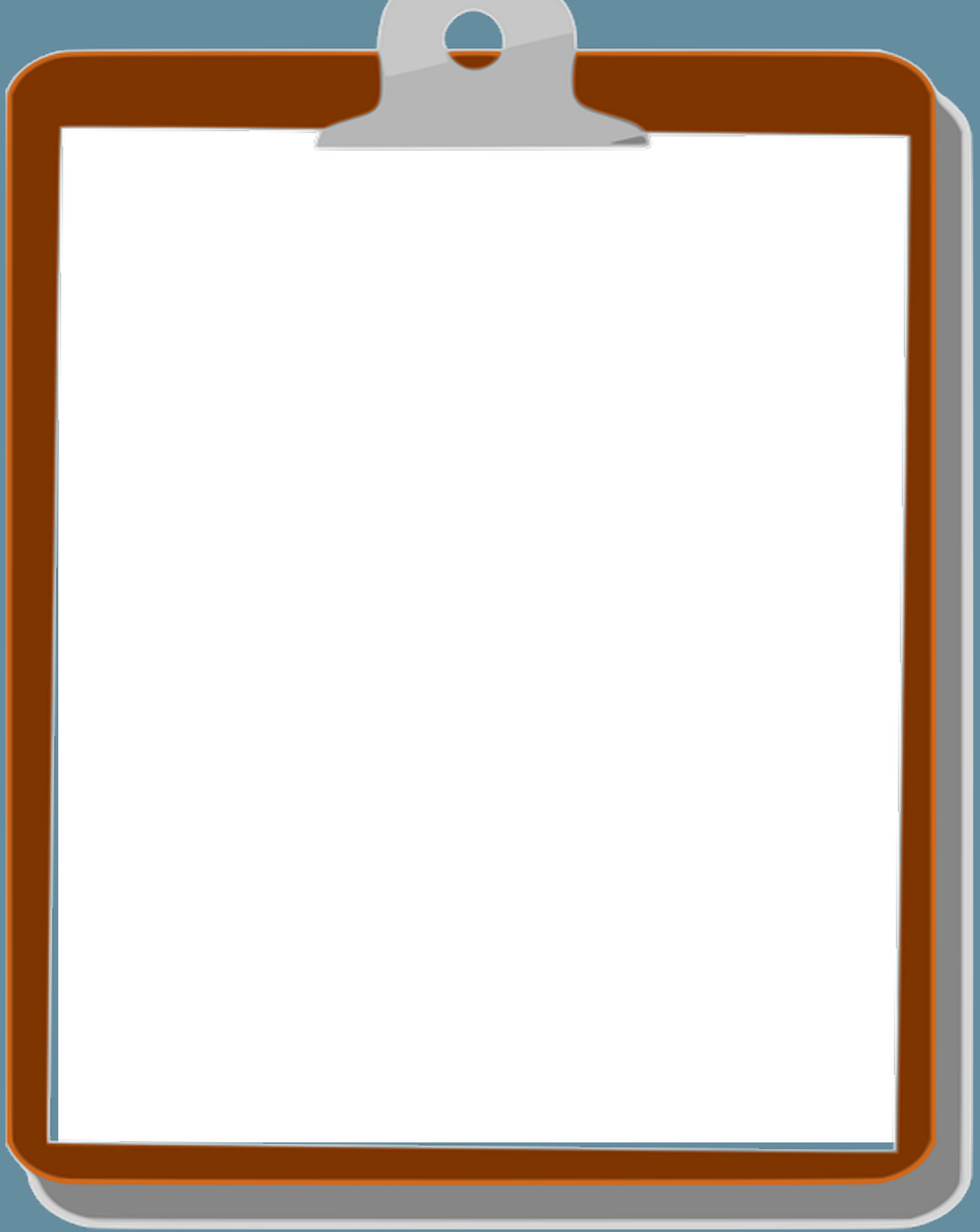
6th Law of Artifact Sha[illegible]ng

(Inverse Costner's Law)

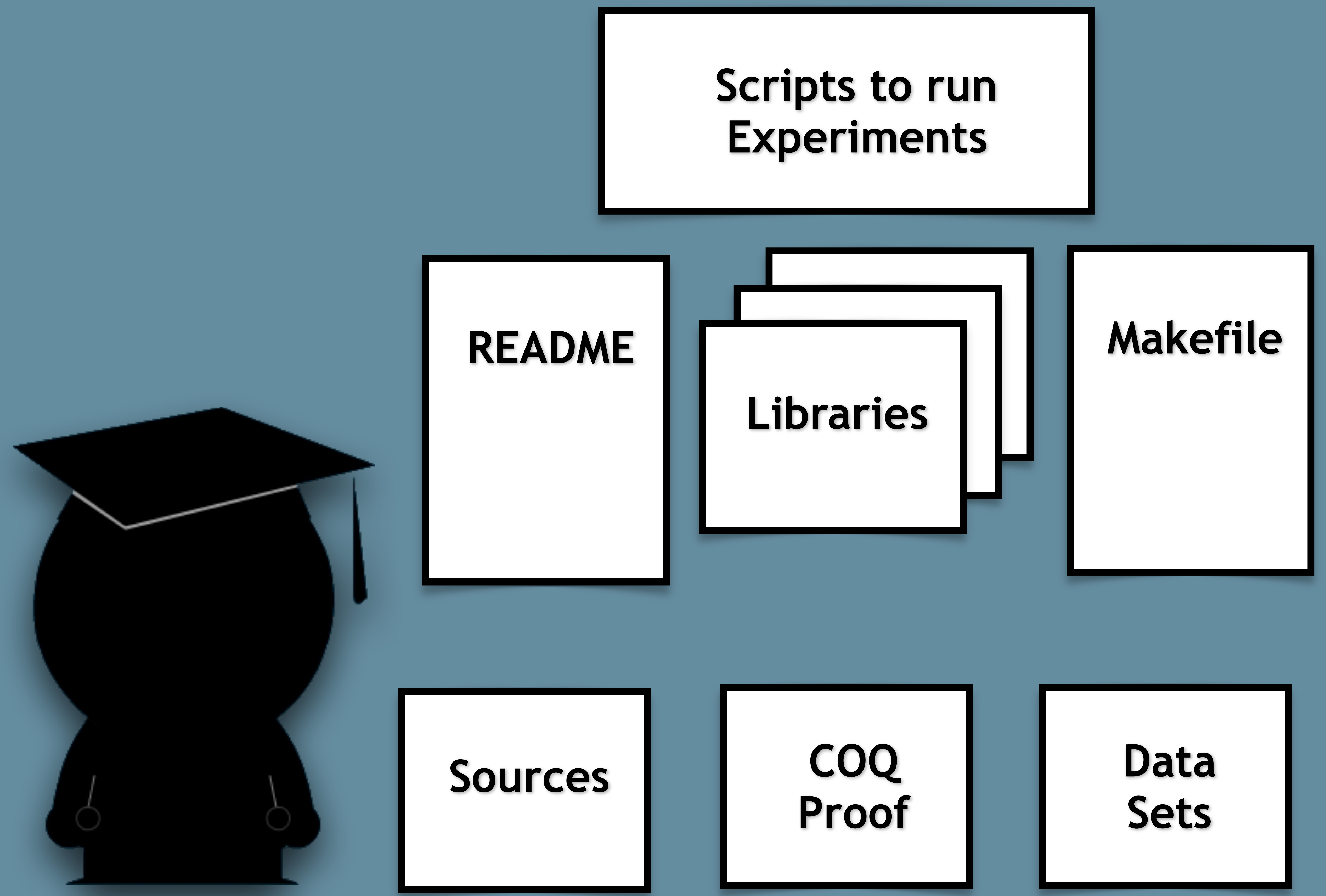
Even if you built it,
they still wouldn't come.

Sharing Proposal
— #2 —

Checklists



- ☑ Clearly link paper to artifact



- [x] Clearly link paper to artifact
- [x] Share everything

- ☑ Clearly link paper to artifact
- ☑ Share everything
- ☑ Include external code

- ☑ Clearly link paper to artifact
- ☑ Share everything
- ☑ Include external code
- ☑ Document software you can't include

FindResearch.org

4.7% of verified papers with shared artifacts have broken links

- ☑ Clearly link paper to artifact
- ☑ Share everything
- ☑ Include external code
- ☑ Document software you can't include
- ☑ Ensure availability

FindResearch.org

- 9% of emails bounced
- 14% of articles without any email address

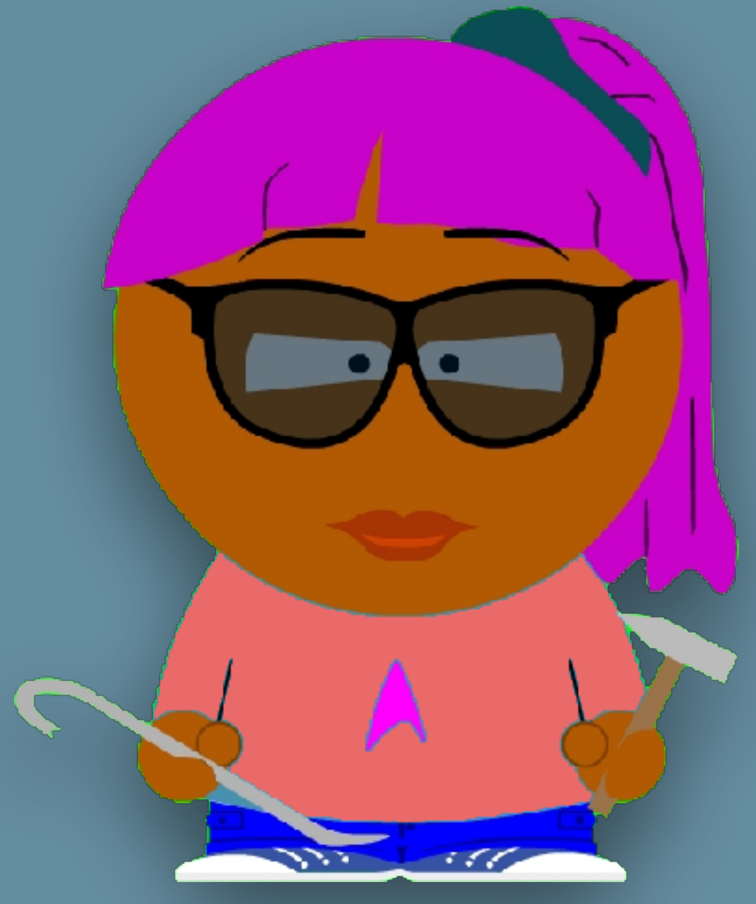
- ☑ Clearly link paper to artifact
- ☑ Share everything
- ☑ Include external code
- ☑ Document software you can't include
- ☑ Ensure availability
- ☑ Use permanent email addresses

5th Law of Artifact Sharing

To ensure repeatability of your results by others, you must

1. share everything
2. assume nothing
3. remain reachable

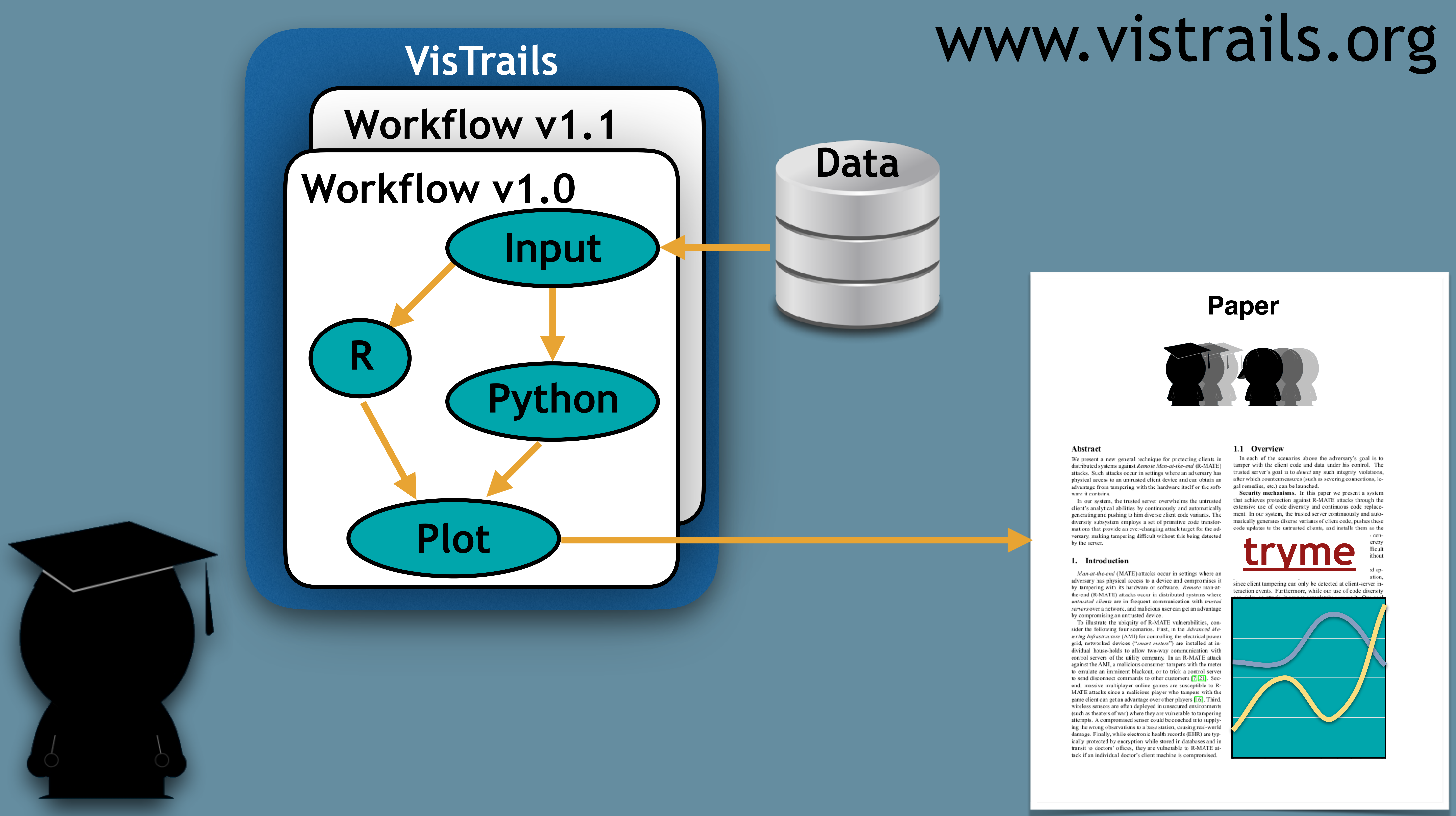
Sharing Proposal — #3 —



Tool Support

www.vistrails.org
VisTrails
Workflow v1.0
Data
Paper
Abstract
1.1 Overview
1. Introduction
tryme

www.vistrails.org
VisTrails
Workflow v1.0
Input
R
Python
Plot
Data
Paper
Abstract
1.1 Overview
1. Introduction
tryme


www.vistrails.org
VisTrails
Workflow v1.1
Workflow v1.0
Input
R
Python
Plot
Data
Paper
Abstract
1.1 Overview
1. Introduction
tryme

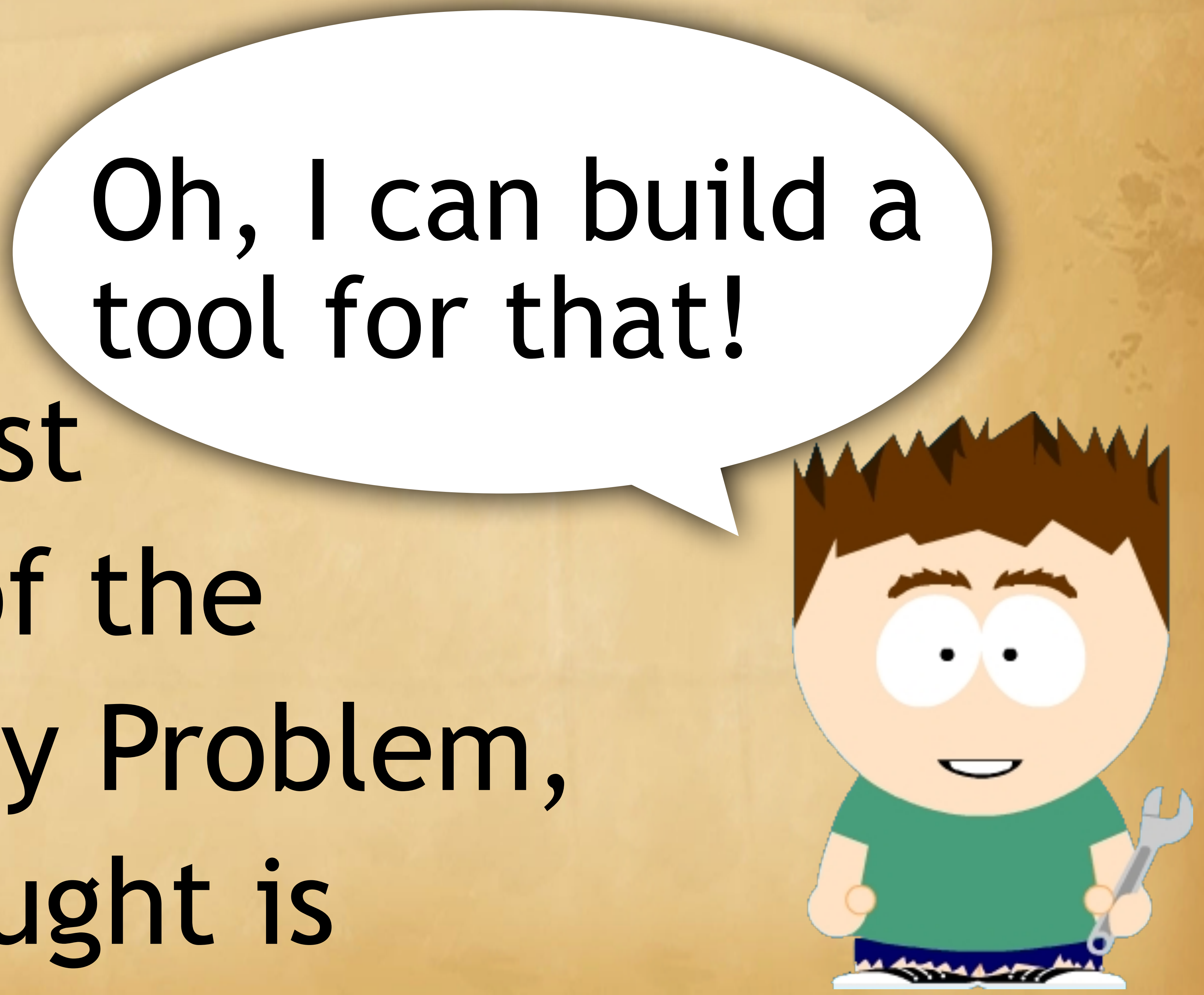

VisTrails - Vistrail Builder
File View Help
Send to Spreadsheet
corie5.xml - Time Step 1
corie5.xml
vtkElcircUnstructuredGridReader
vtkLookupTable
vtkTextProperty
vtkDataSetMapper
vtkProperty
vtkScalarBarActor
vtkTextProperty
vtkActor
vtkTextActor
vtkRenderer
vtkRenderWindow
Visualization Name: Time Step 1
Change
Search:
VTK Classes
Module Methods
Method
Signature
vtkElcircUnstructuredGridReader
SetVerticalScale void(double)
SetTimeStep void(int)
SetFromLayer void(int)
SetToLayer void(int)
vtkDataReader
vtkAlgorithm
vtkObject
vtkObjectBase
Update
Update and Send
SetFileName
const char * (unnamed) c:/data/corie/2_salt.63
SetTimeStep
int (unnamed) 1
VisTrails - Spreadsheet
File
Salinity
60.0
45.0
30.0
15.0
0.000
Time Step 1
Time Step 30
Time Step 60
Time Step 90
Rows 2
Cols 2

4th Law of Artifact Sharing

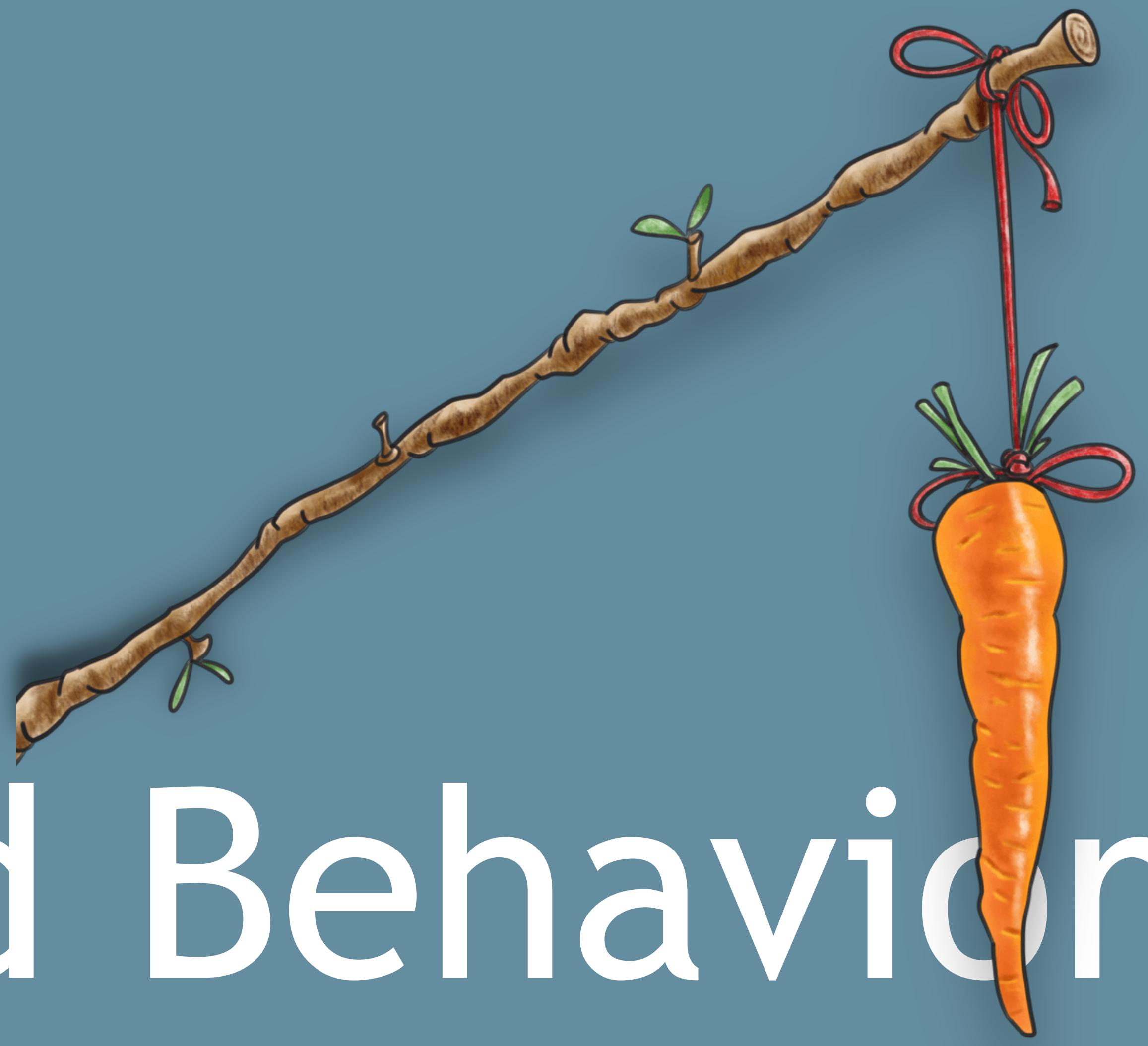
When a
Computer
Scientist is first
made aware of the
Reproducibility Problem,
their first thought is

4th Law of Artifact Sharing

When a Computer Scientist is first made aware of the Reproducibility Problem, their first thought is



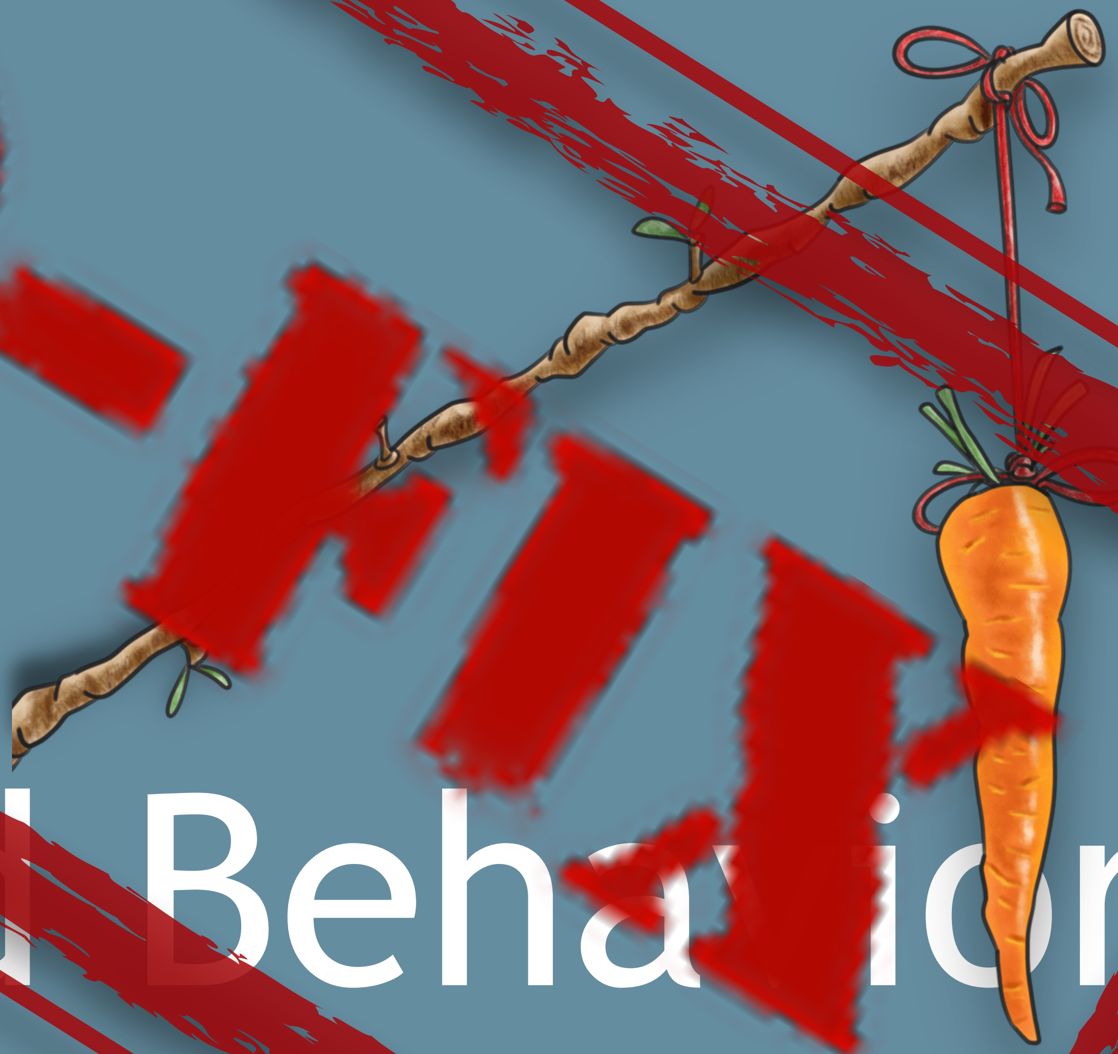
Sharing Proposal
— #4 —



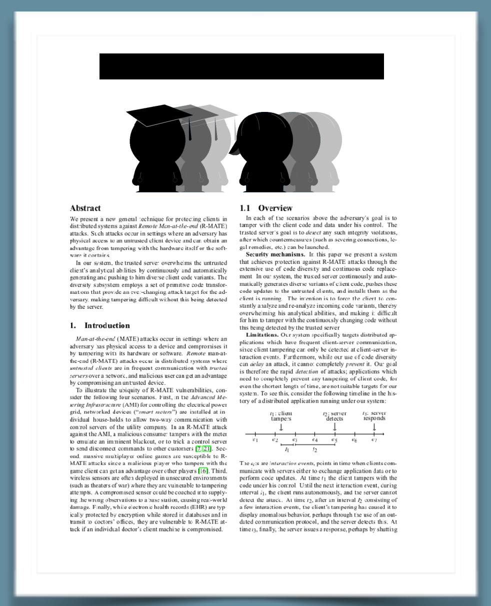
Rewarding Good Behavior

Sharing Proposal
— #4 —

SOCIO-FIX

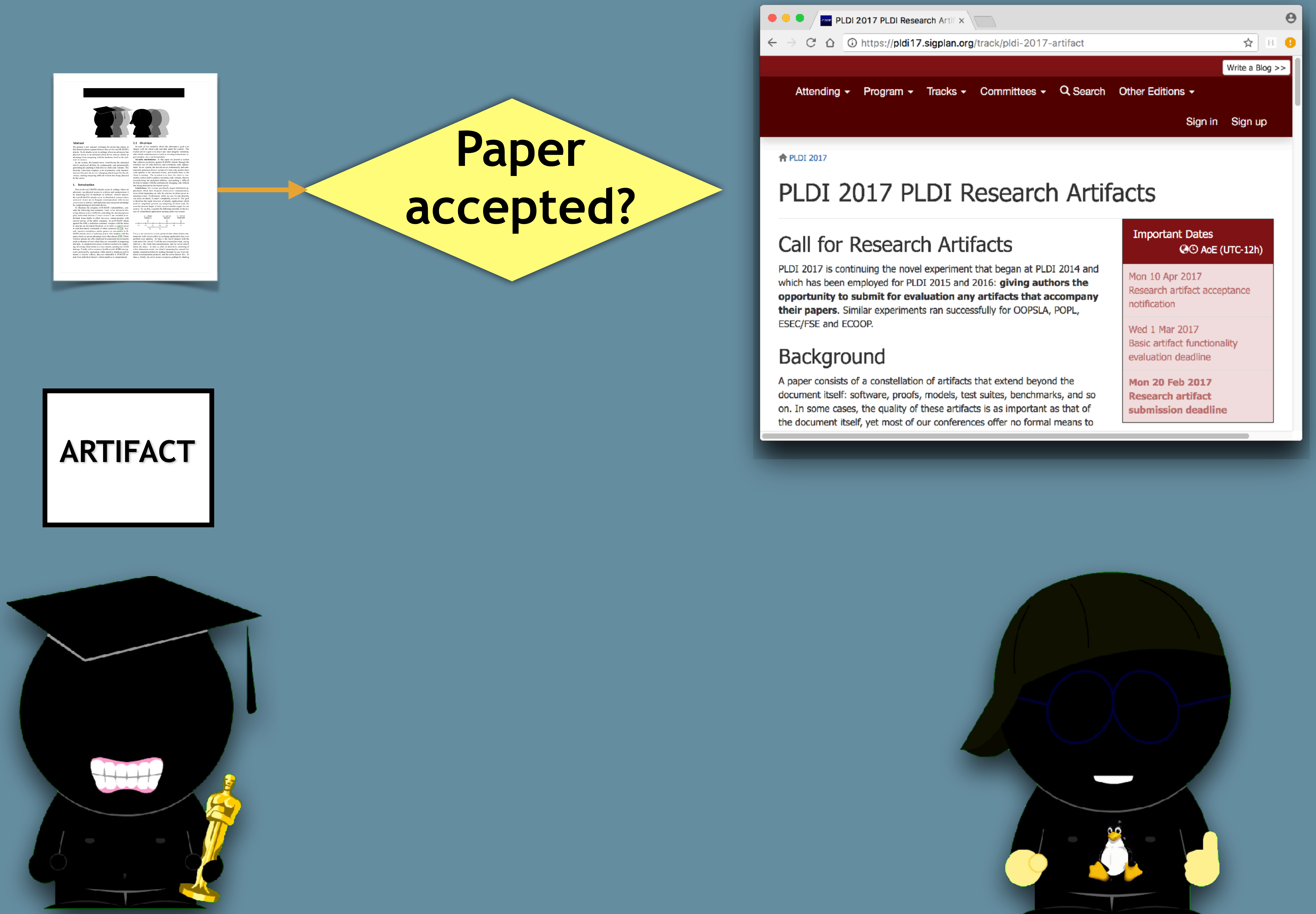


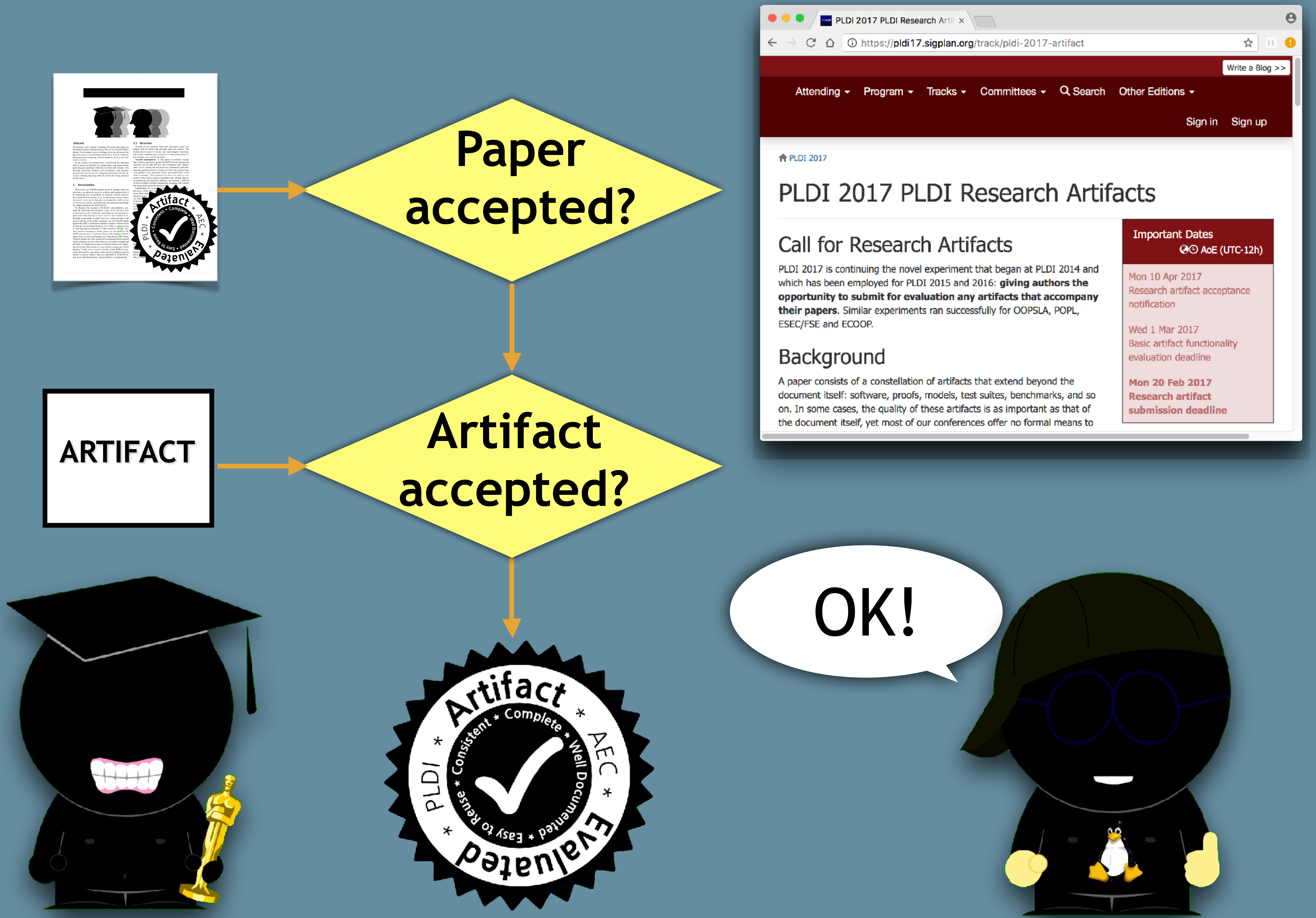
Rewarding Good Behavior



ARTIFACT

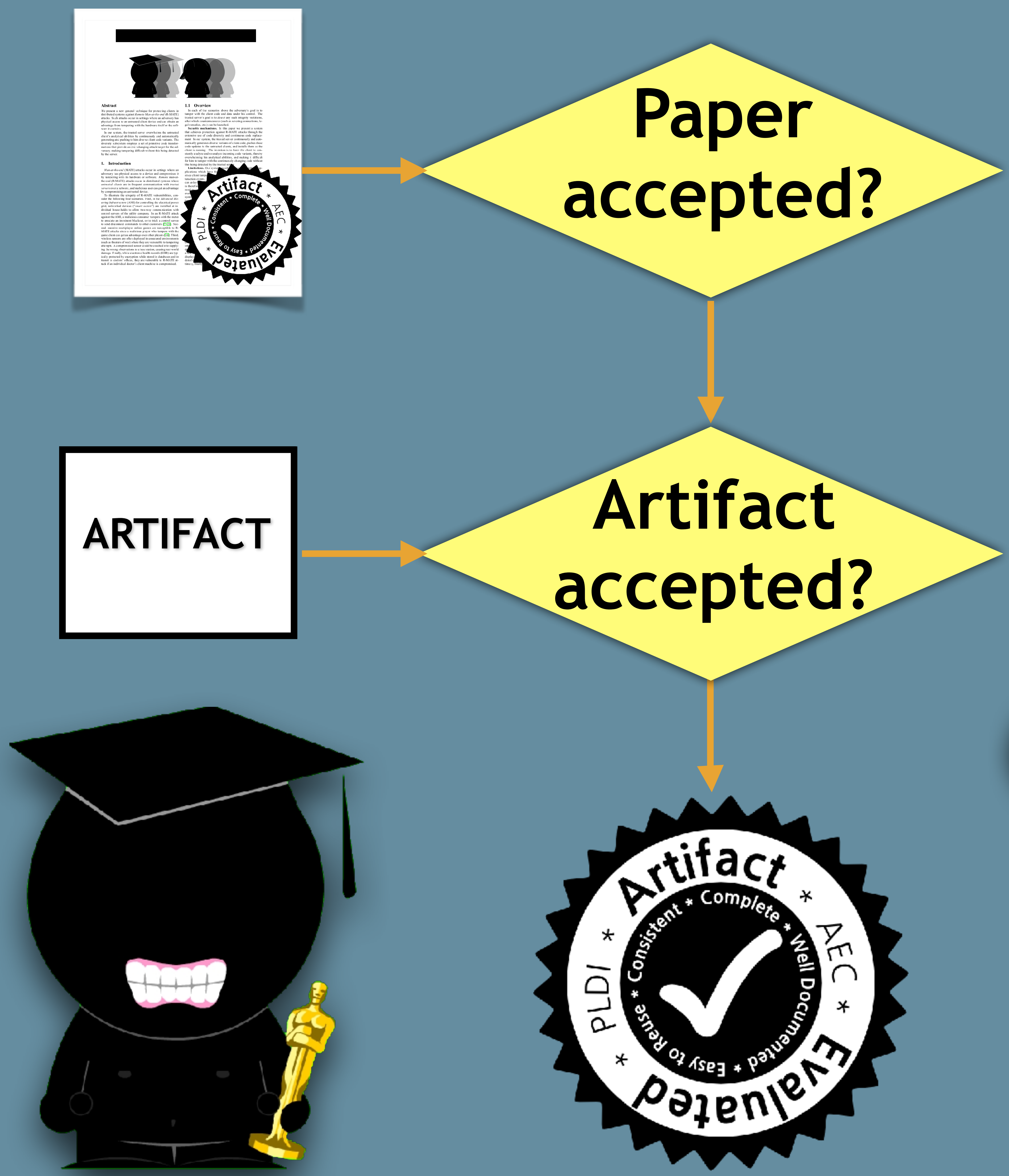

PLDI 2017 PLDI Research Arti
https://pldi17.sigplan.org/track/pldi-2017-artifact
Write a Blog >>
Attending
Program
Tracks
Committees
Search
Other Editions
Sign in
Sign up
PLDI 2017
PLDI 2017 PLDI Research Artifacts
Call for Research Artifacts
PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.
Background
A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to
Important Dates
AoE (UTC-12h)
Mon 10 Apr 2017
Research artifact acceptance notification
Wed 1 Mar 2017
Basic artifact functionality evaluation deadline
Mon 20 Feb 2017
Research artifact submission deadline


Paper accepted?
ARTIFACT
PLDI 2017 PLDI Research Arti
https://pldi17.sigplan.org/track/pldi-2017-artifact
Write a Blog >>
Attending
Program
Tracks
Committees
Search
Other Editions
Sign in
Sign up
PLDI 2017
PLDI 2017 PLDI Research Artifacts
Call for Research Artifacts
PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.
Background
A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to
Important Dates
AoE (UTC-12h)
Mon 10 Apr 2017
Research artifact acceptance notification
Wed 1 Mar 2017
Basic artifact functionality evaluation deadline
Mon 20 Feb 2017
Research artifact submission deadline



Paper accepted?
ARTIFACT
Artifact accepted?
PLDI 2017 PLDI Research Arti
https://pldi17.sigplan.org/track/pldi-2017-artifact
Write a Blog >>
Attending Program Tracks Committees Search Other Editions
Sign in Sign up
PLDI 2017
PLDI 2017 PLDI Research Artifacts
Call for Research Artifacts
PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.
Background
A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to
Important Dates
AoE (UTC-12h)
Mon 10 Apr 2017
Research artifact acceptance notification
Wed 1 Mar 2017
Basic artifact functionality evaluation deadline
Mon 20 Feb 2017
Research artifact submission deadline
Artifact
Consistent * Complete * Well Documented * Easy to Reuse *
PLDI
AEC
Evaluated
OK!

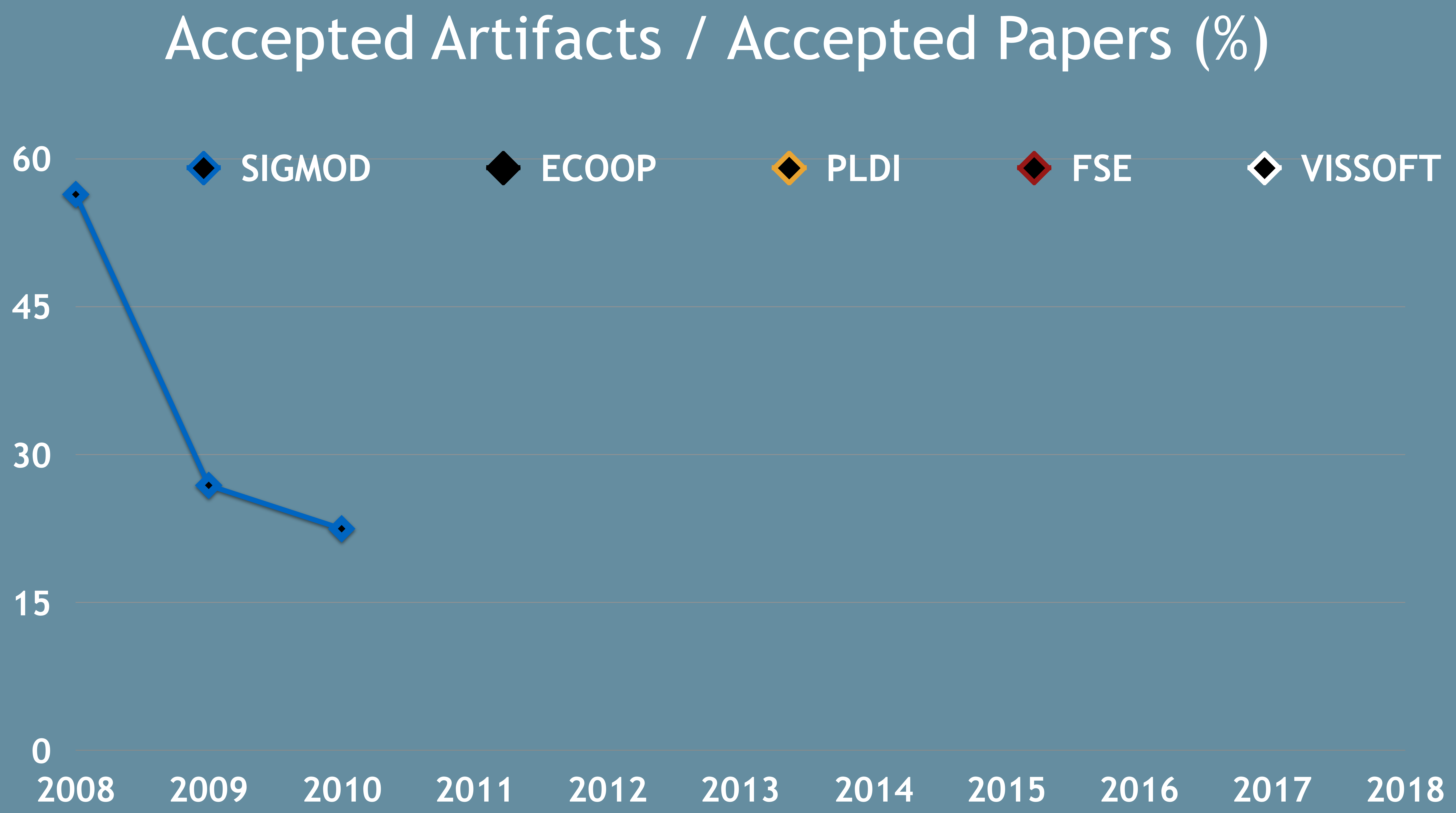
Paper accepted?
ARTIFACT
Artifact accepted?
Artifact * Complete * Well Documented * Easy to Reuse * Consistent * PLDI * AEC * Evaluated
PLDI 2017 PLDI Research Arti
https://pldi17.sigplan.org/track/pldi-2017-artifact
Write a Blog >>
Attending Program Tracks Committees Search Other Editions
Sign in Sign up
PLDI 2017
PLDI 2017 PLDI Research Artifacts
Call for Research Artifacts
PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: giving authors the opportunity to submit for evaluation any artifacts that accompany their papers. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.
Background
A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to
Important Dates
AoE (UTC-12h)
Mon 10 Apr 2017
Research artifact acceptance notification
Wed 1 Mar 2017
Basic artifact functionality evaluation deadline
Mon 20 Feb 2017
Research artifact submission deadline
OK!
Artifacts Available acm
Results Replicated acm
Artifacts Evaluated Functional acm
Results Reproduced acm
Artifacts Evaluated Reusable acm

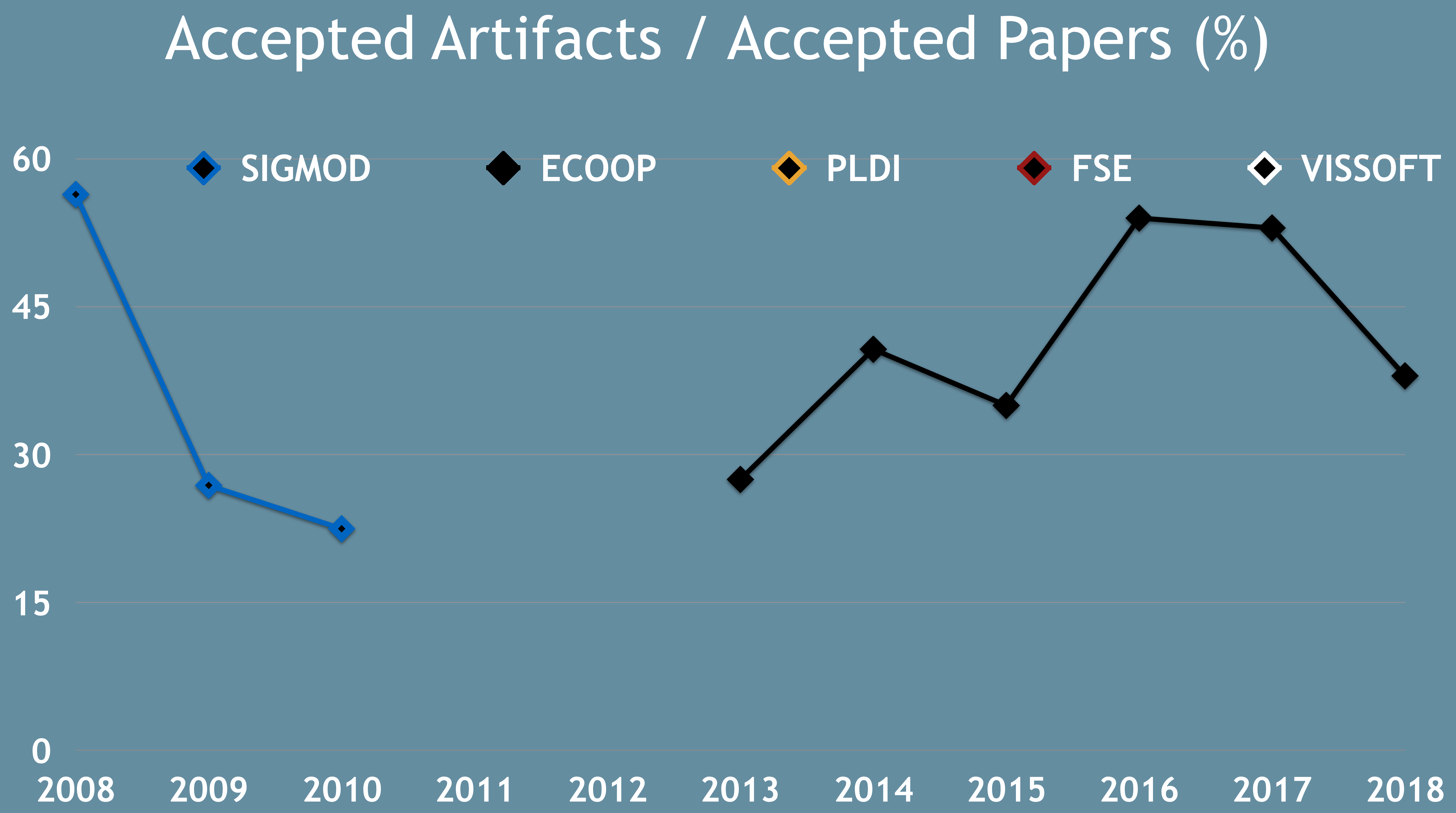


- Voluntary
- Does not affect accept/reject
- No expectation of sharing

Accepted Artifacts / Accepted Papers (%)

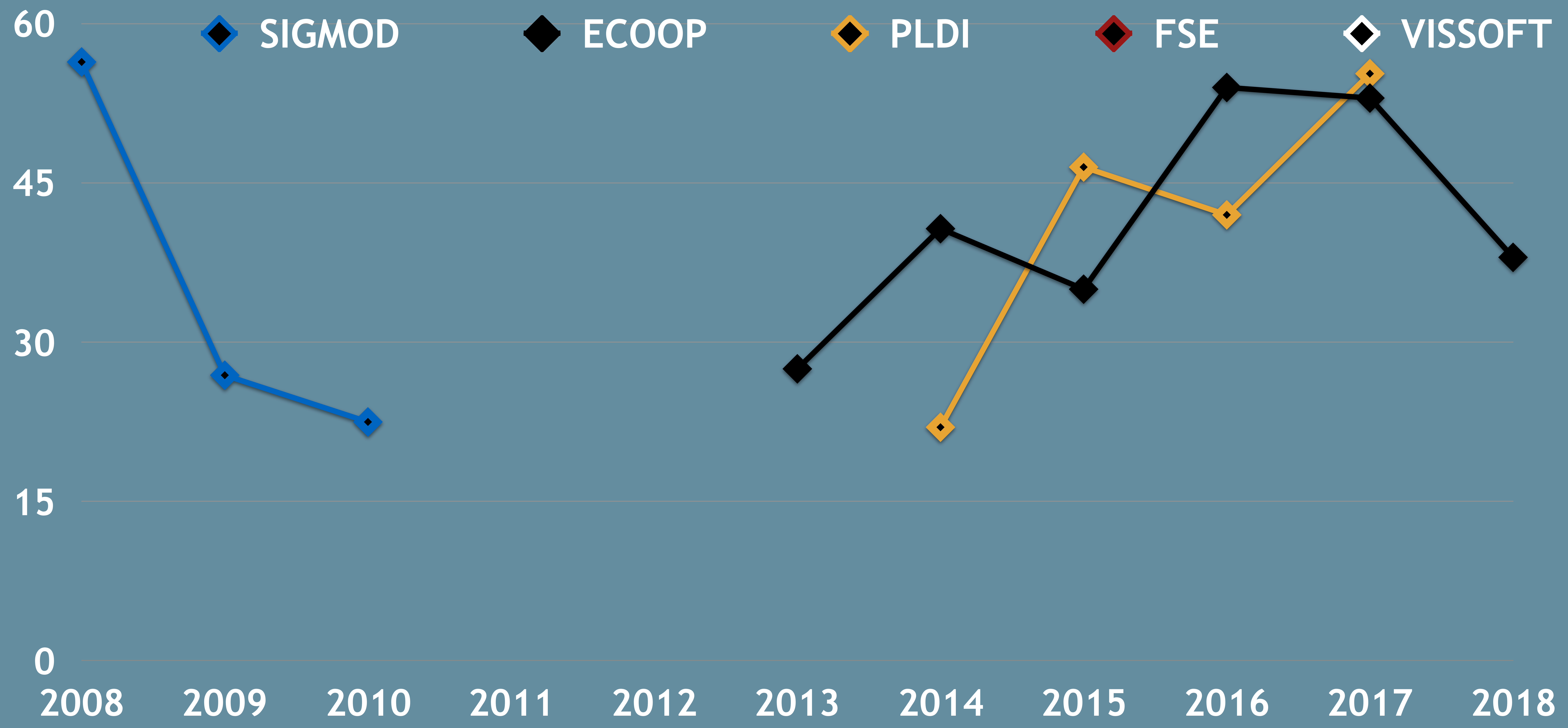
Accepted Artifacts / Accepted Papers (%)


Accepted Artifacts / Accepted Papers (%)
SIGMOD
ECOOP
PLDI
FSE
VISSOFT
60
45
30
15
0
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018

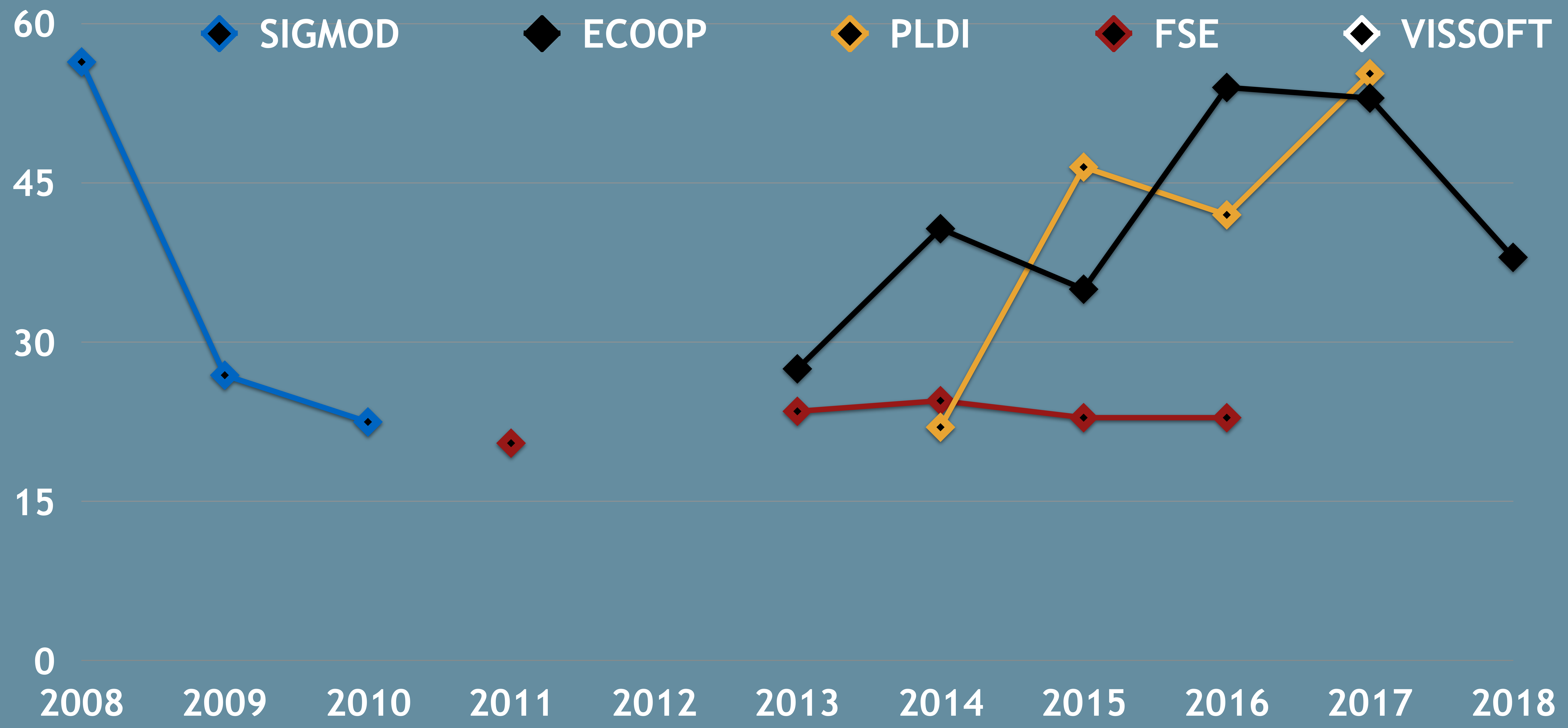


Accepted Artifacts / Accepted Papers (%)
SIGMOD
ECOOP
PLDI
FSE
VISSOFT
60
45
30
15
0
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018

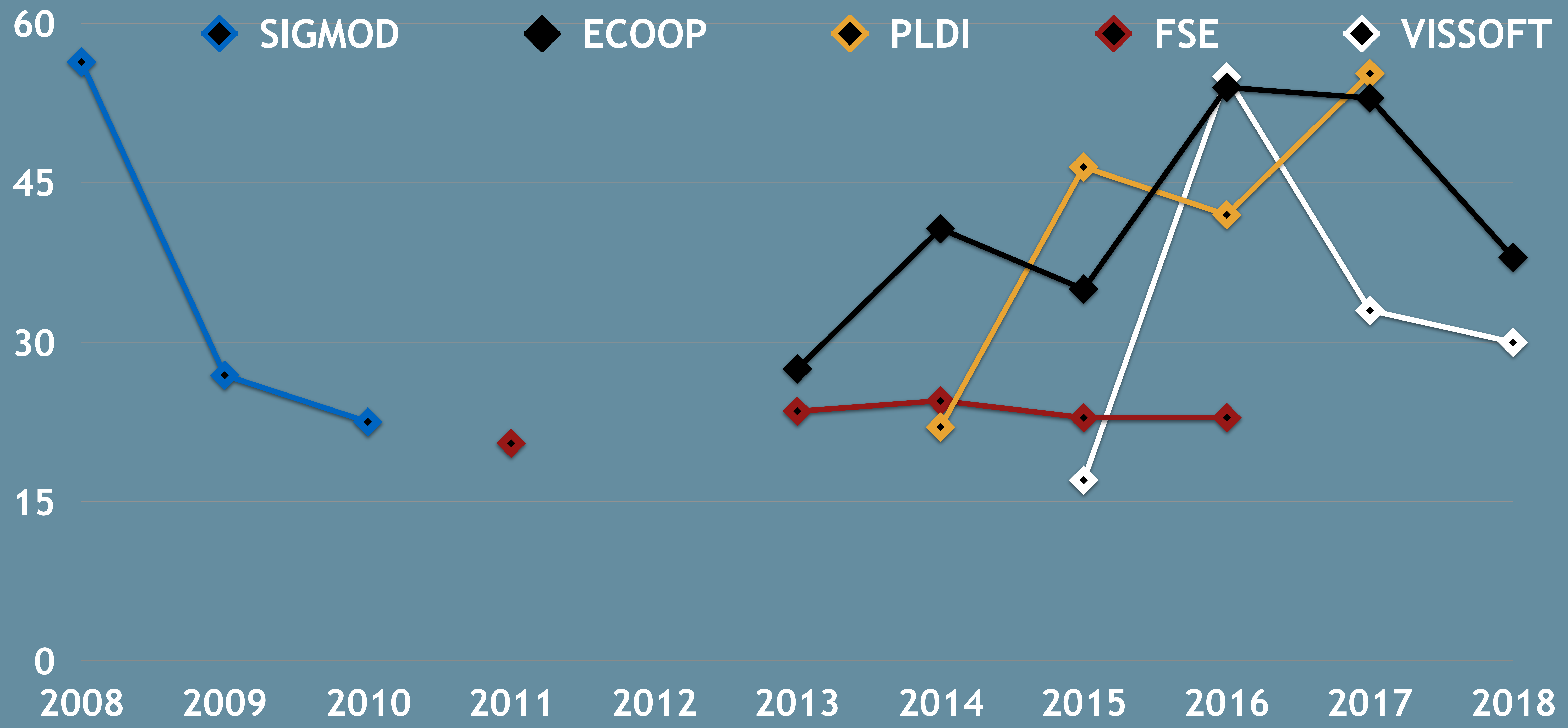
Accepted Artifacts / Accepted Papers (%)



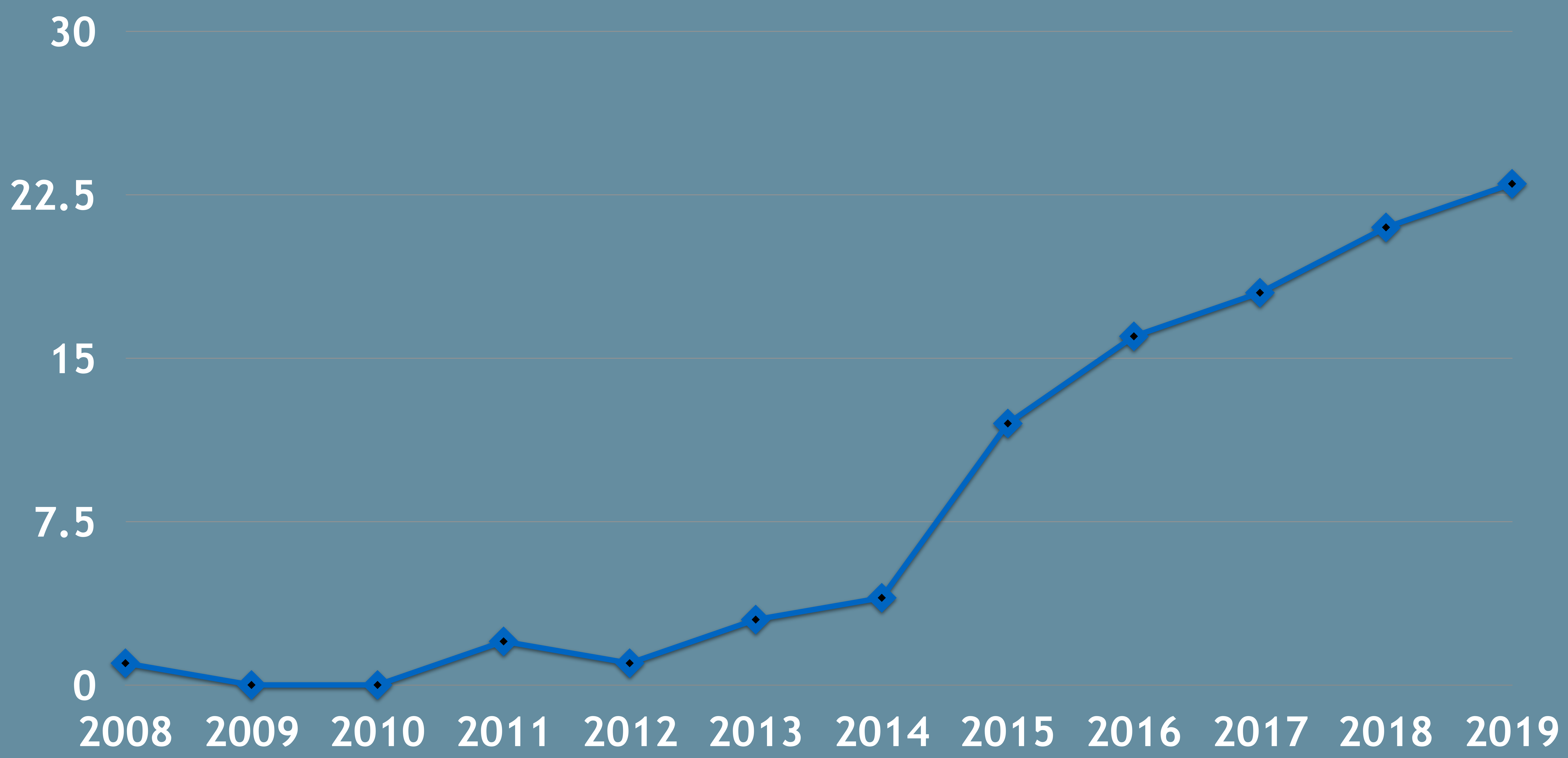
Accepted Artifacts / Accepted Papers (%)



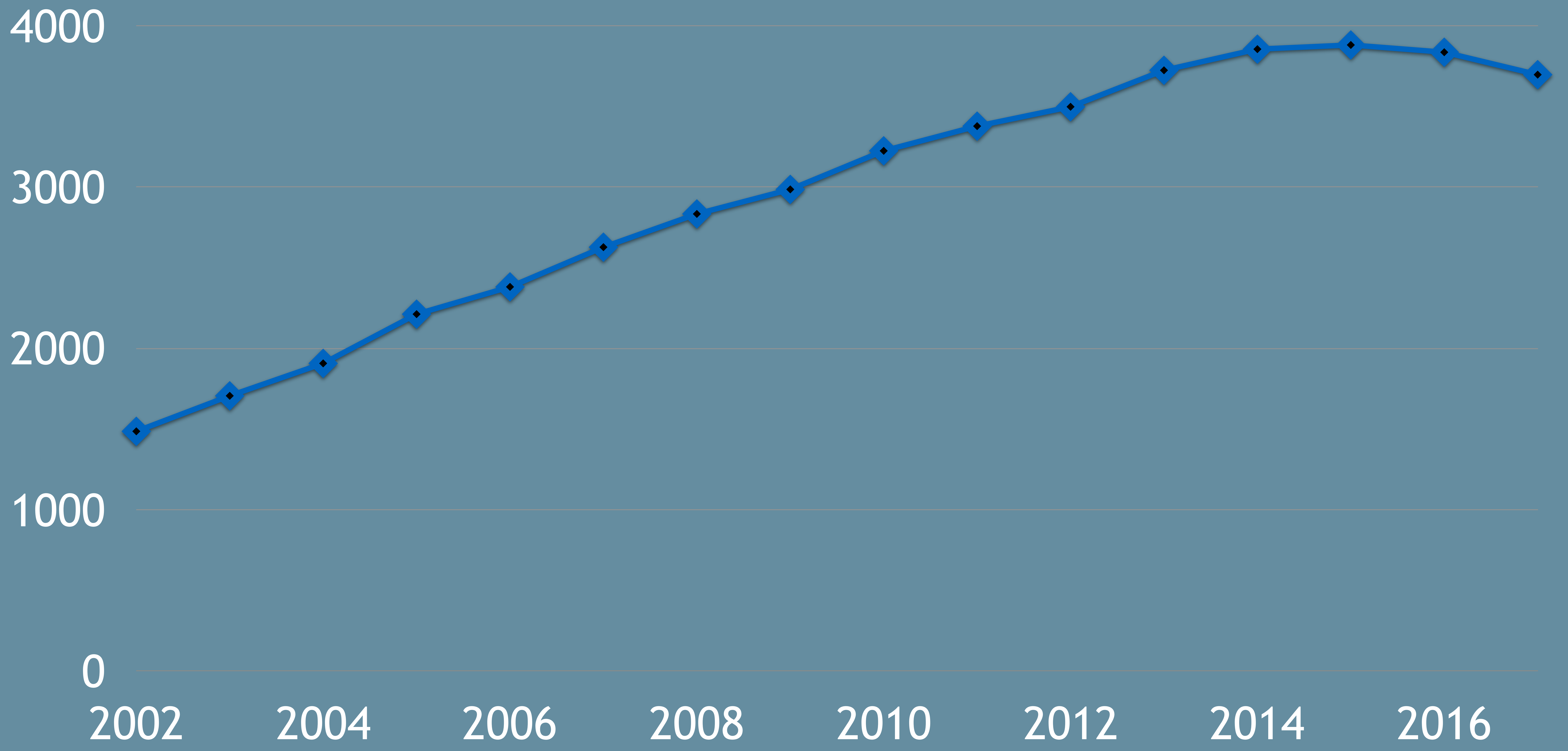
Accepted Artifacts / Accepted Papers (%)



Conferences with AE

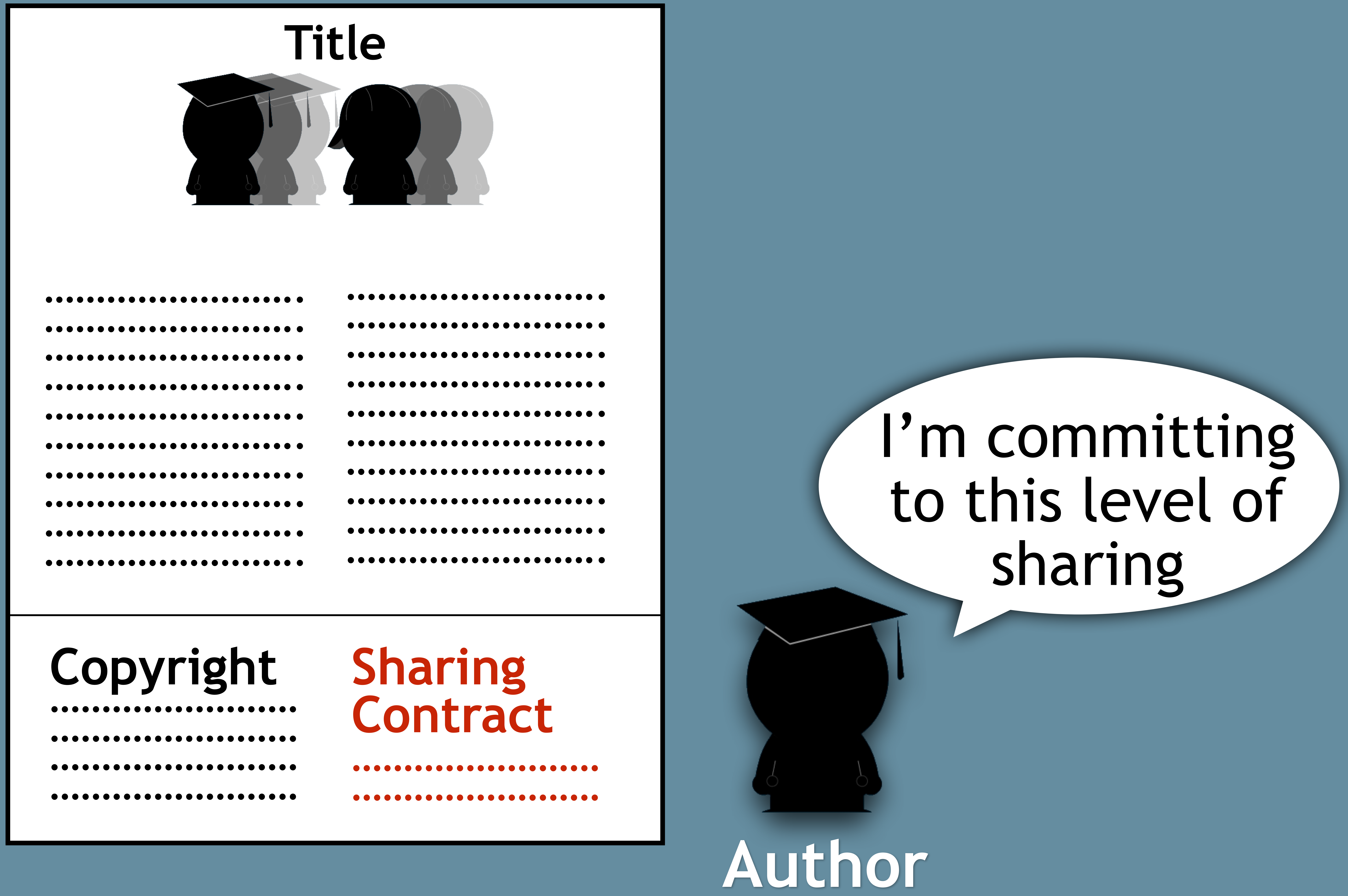


Publication Venues (Dblp)



Sharing Proposal
— #5 —

Punishing Bad Behavior


Title
Copyright
Sharing Contract
I'm committing to this level of sharing
Author


Title
Copyright
Sharing
Contract
Accept/
Reject?
Reviewer
I'm committing
to this level of
sharing
Author


Title
Copyright
Sharing Contract
You promised!
I’m committing to this level of sharing
Reader
Author


Title
Copyright
Sharing Contract
Sharing Contract
•License: ...
•Artifacts: source code, data, ...
•Where: ...
•Support: ...

Sharing Proposal
— #7 —

Sharing Proposal
— #7 —

CS Research Methods Courses?

CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics

CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics

Reproducibility???

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF MECHANICAL ENGINEERING

2.671 Measurement and Instrumentation

Keeping a complete and accurate record of experimental methods and data … **could someone else**, … use your notebook to **repeat your work**, and obtain the same results?

Reproducibility PI Manifesto

Lorena Barba

I pledge to

- ☑ teach grad students about reproducibility
- ☑ share artifacts at the time of submission
- ☑ add a *reproducibility statement* to papers

http://lorenabarba.com/gallery/reproducibility-pi-manifesto/

Sharing Proposal

— #8 —

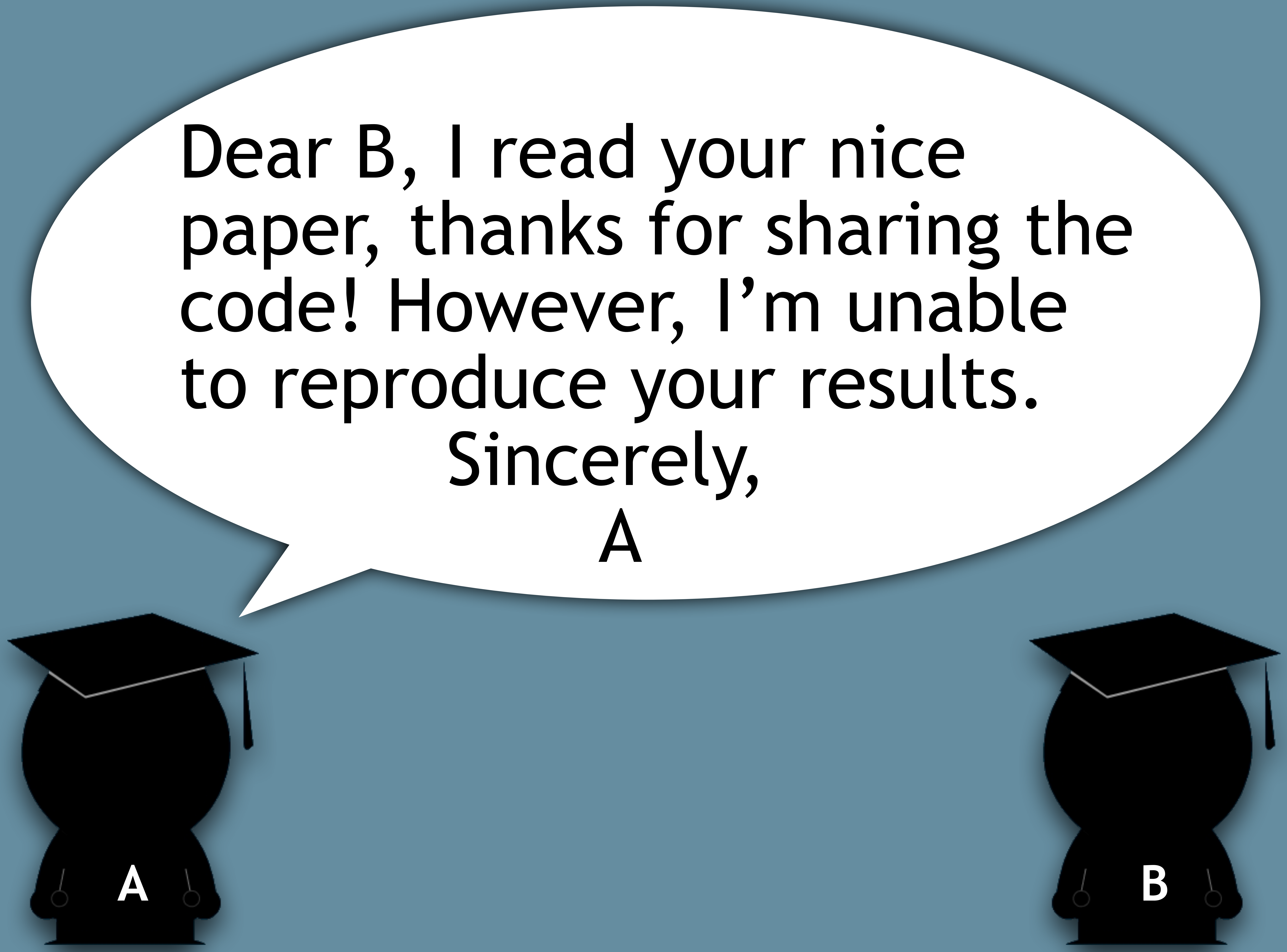
All I Really Need to Know I Learned in Kindergarten

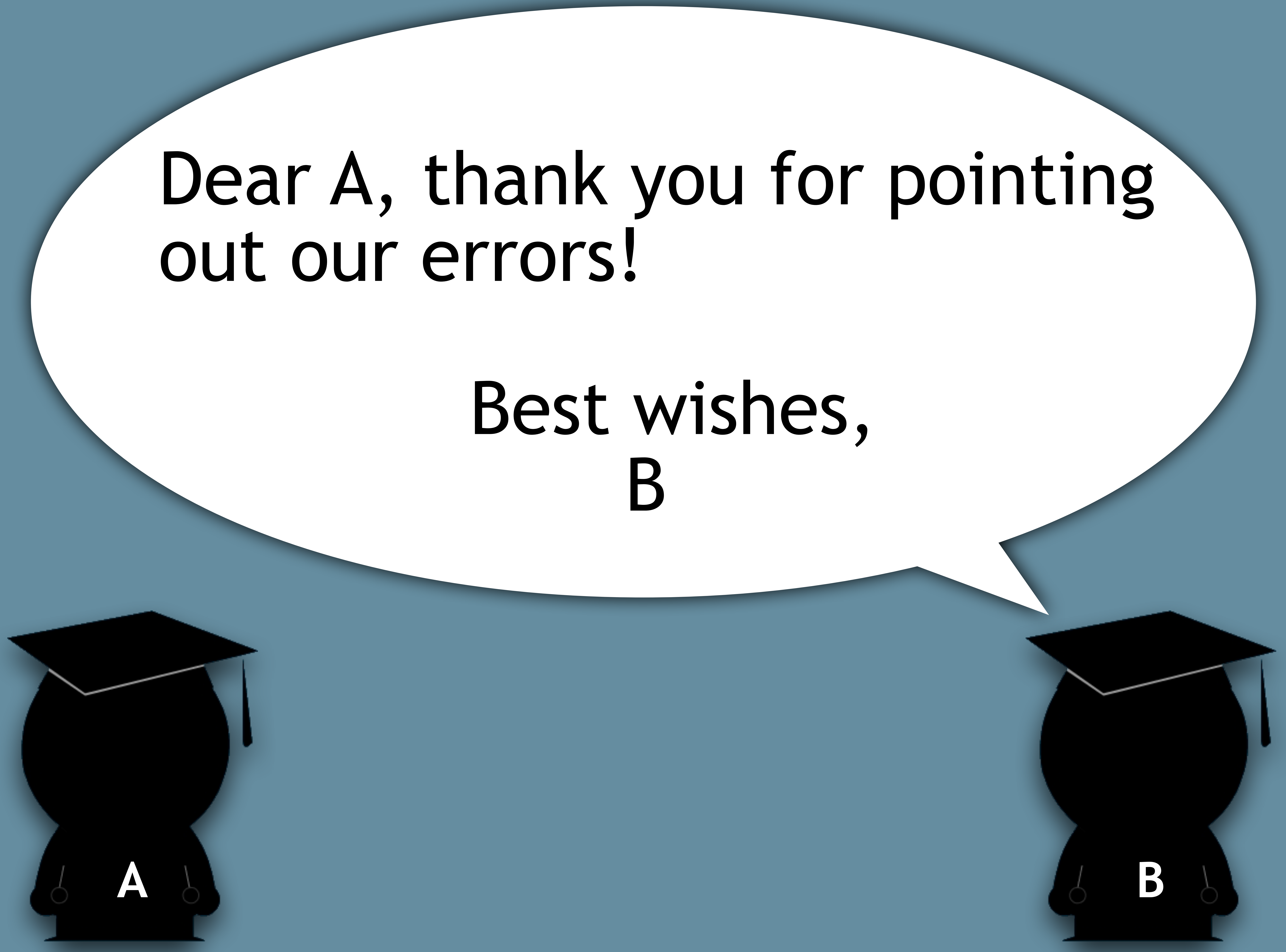
Why do we care about reproducibility and repeatability?

Why do we care about reproducibility and repeatability?

Why do we care about reproducibility and repeatability?

Why do we care about reproducibility and repeatability?


Dear B, I read your nice paper, thanks for sharing the code! However, I'm unable to reproduce your results.
Sincerely,
A
A
B


Dear A, thank you for pointing out our errors!
Best wishes,
B
A
B


Dear A, thank you for pointing out our errors!
Best wishes,
B
Respect
A

BLOG@CACM

Yes, Computer Scientists Are Hypercritical

By Jeannette M. Wing
October 6, 2011
Comments (15)

VIEW AS: SHARE:

Are computer scientists hypercritical? Are we more critical than scientists and engineers in other disciplines? Bertrand Meyer's August 22, 2011 The Nastiness Problem in Computer Science blog post partially makes the argument referring to secondhand information from the National Science Foundation (NSF). Here are some NSF numbers to back the claim that we are hypercritical.

This graph plots average reviewer ratings of all proposals submitted from 2005 to 2010 to NSF overall (red line), just Computer & Information Science & Engineering (CISE) (green line), and NSF minus CISE (blue line). Proposal ratings are based on a scale of 1 (poor) to 5 (excellent). For instance, in 2010, the average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding CISE, 3.30.

https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext

BLOG@CACM

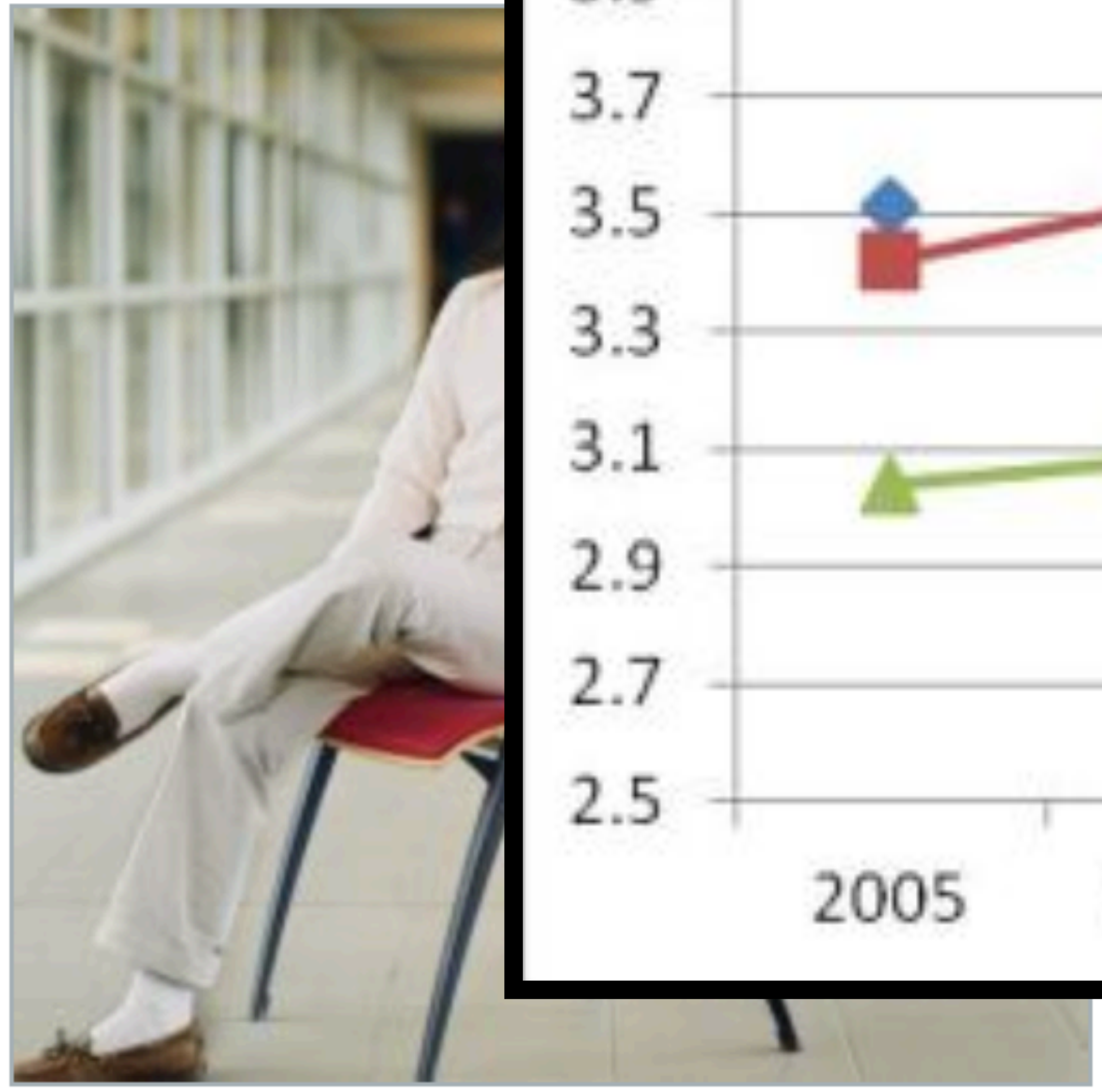
Yes, Computer Scientists Are Hypercritical

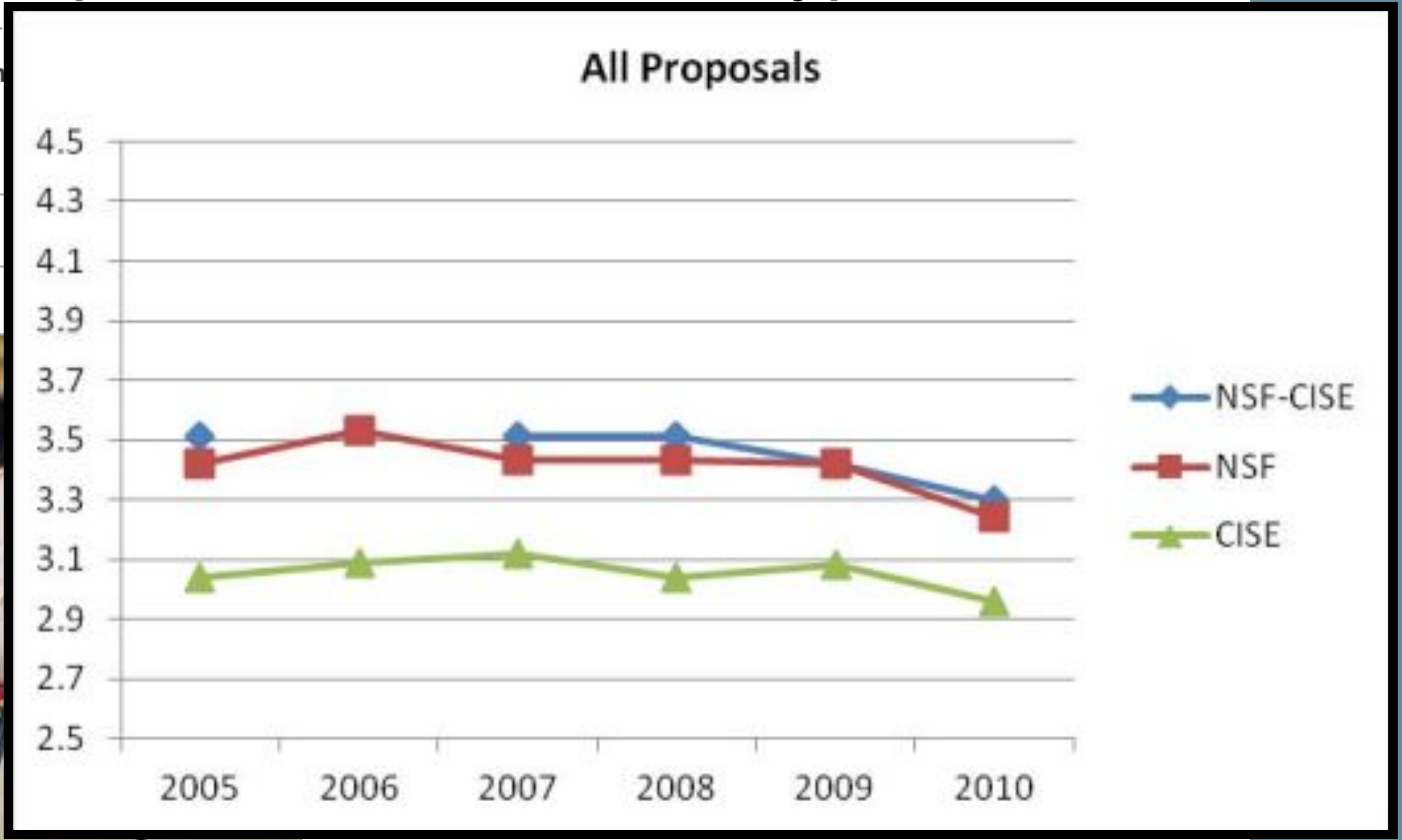
By Jeannette M. Win

October 6, 2011

Comments (15)

VIEW AS:





average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding CISE, 3.30.

https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext

DBMS Research First 50 Years, Next 50 Years

Jeffrey F. Naughton

- SIGMOD 2010
- 350 submissions
- Number of papers with all reviews "accept" or higher:

1

BLOG@CACM

The Nastiness Problem in Computer Science

By Bertrand Meyer
August 22, 2011
Comments (33)

VIEW AS: | SHARE:

Are we malevolent grumps? Nothing personal, but as a community computer scientists sometimes seem to succumb to negativism. They admit it themselves. A common complaint in the profession is that instead of taking a cue from our colleagues in more cogently organized fields such as physics, who band together for funds, promotion, and recognition, we are incurably fractious. In committees, for example, we damage everyone's chances by badmouthing colleagues with approaches other than ours. At least this is a widely perceived view (*"Circling the wagons and shooting inward,"* as Greg Andrews put it in a recent discussion). Is it accurate?

One statistic that I have heard cited is that in 1-to-5 evaluations of projects submitted to the U.S. National Science Foundation the

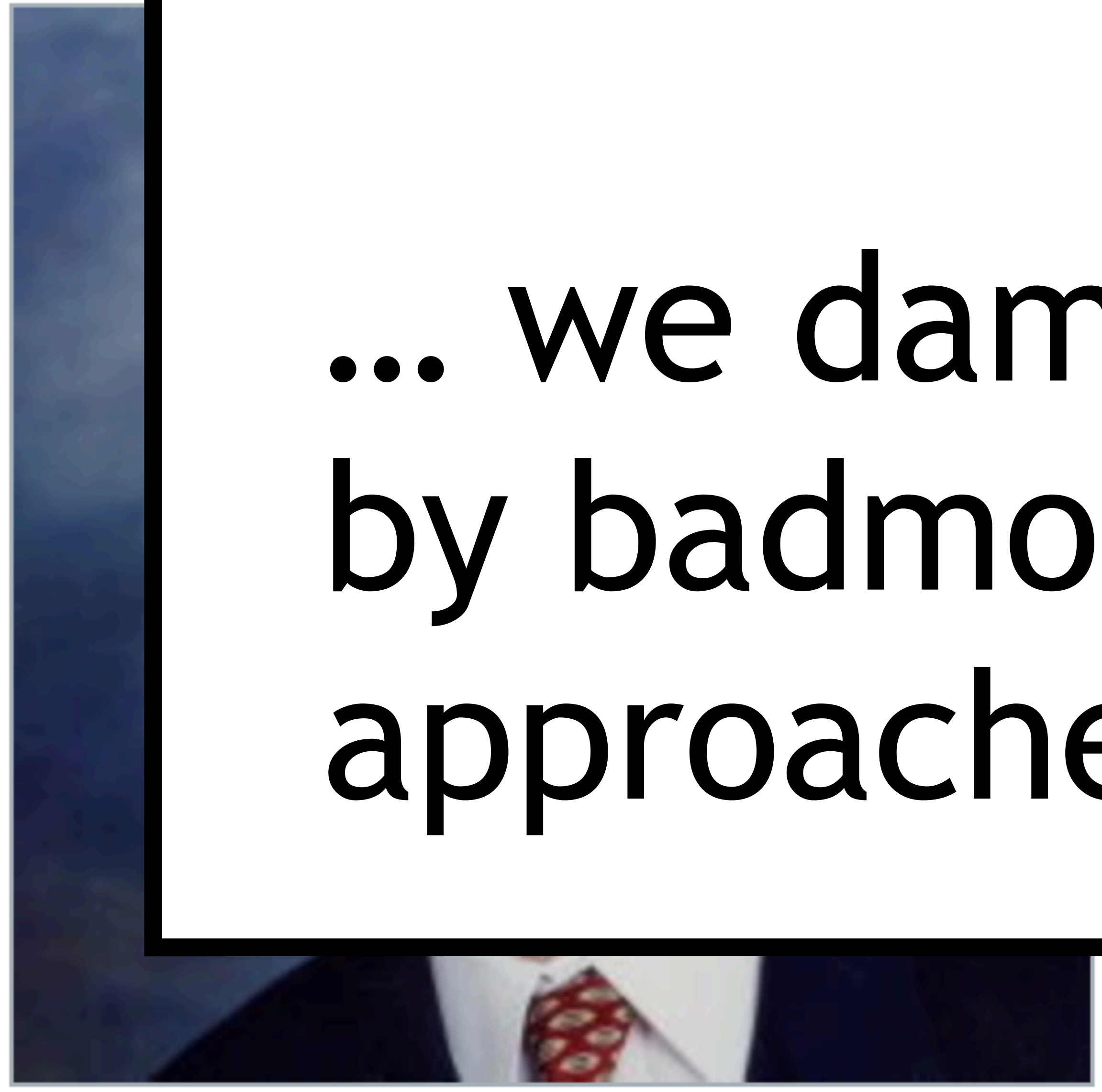
https://cacm.acm.org/blogs/blog-cacm/123611-the-nastiness-problem-in-computer-science/fulltext

BLOG@CACM

The Nastiness Problem in Computer Science

By Bertrand Meyer
August 22, 2011
Comments (33)

VIEW



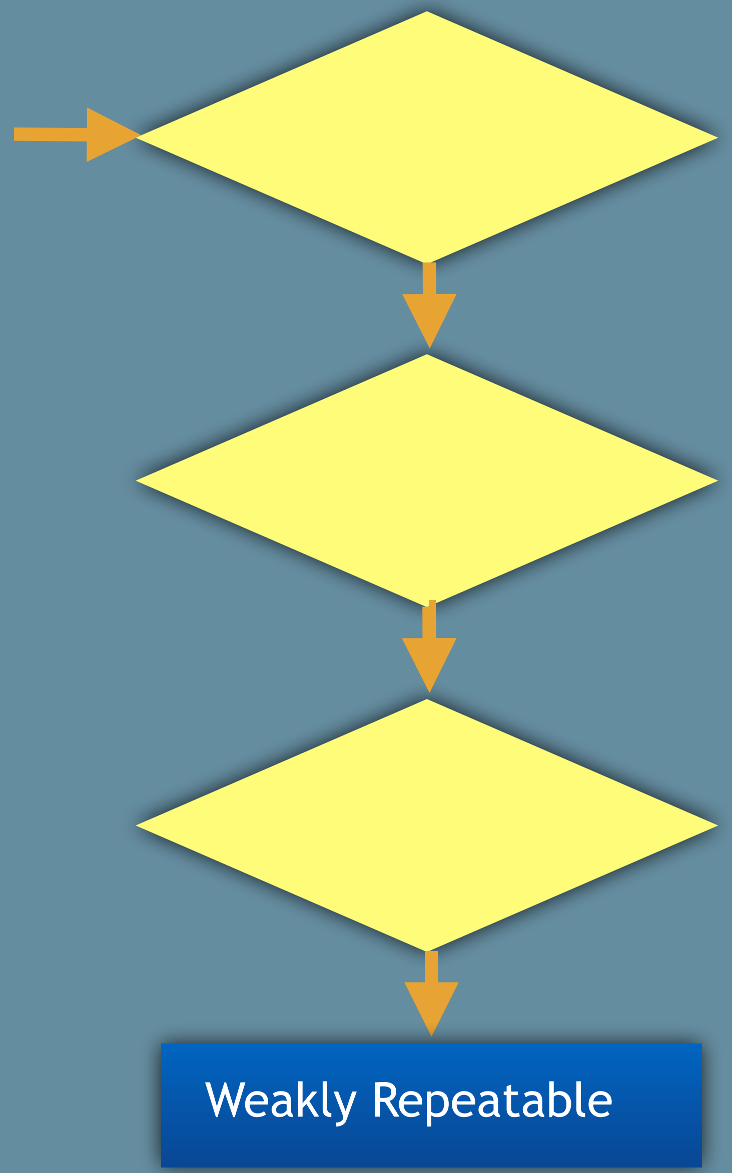
Are we malevolent grumps?

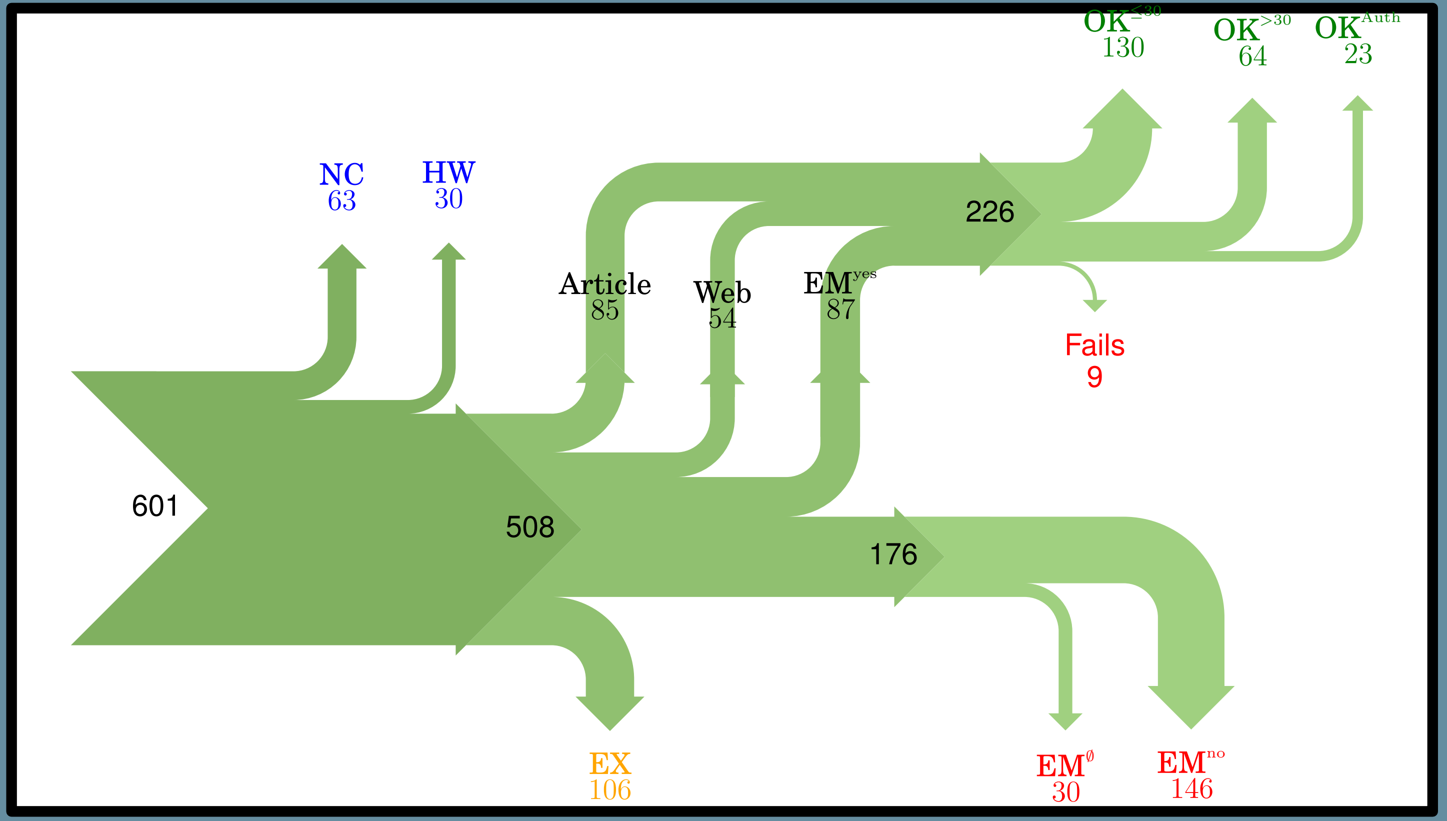
… we damage everyone’s chances by badmouthing colleagues with approaches other than ours.

One statistic that I have heard cited is that in 1-to-5 evaluations of projects submitted to the U.S. National Science Foundation the

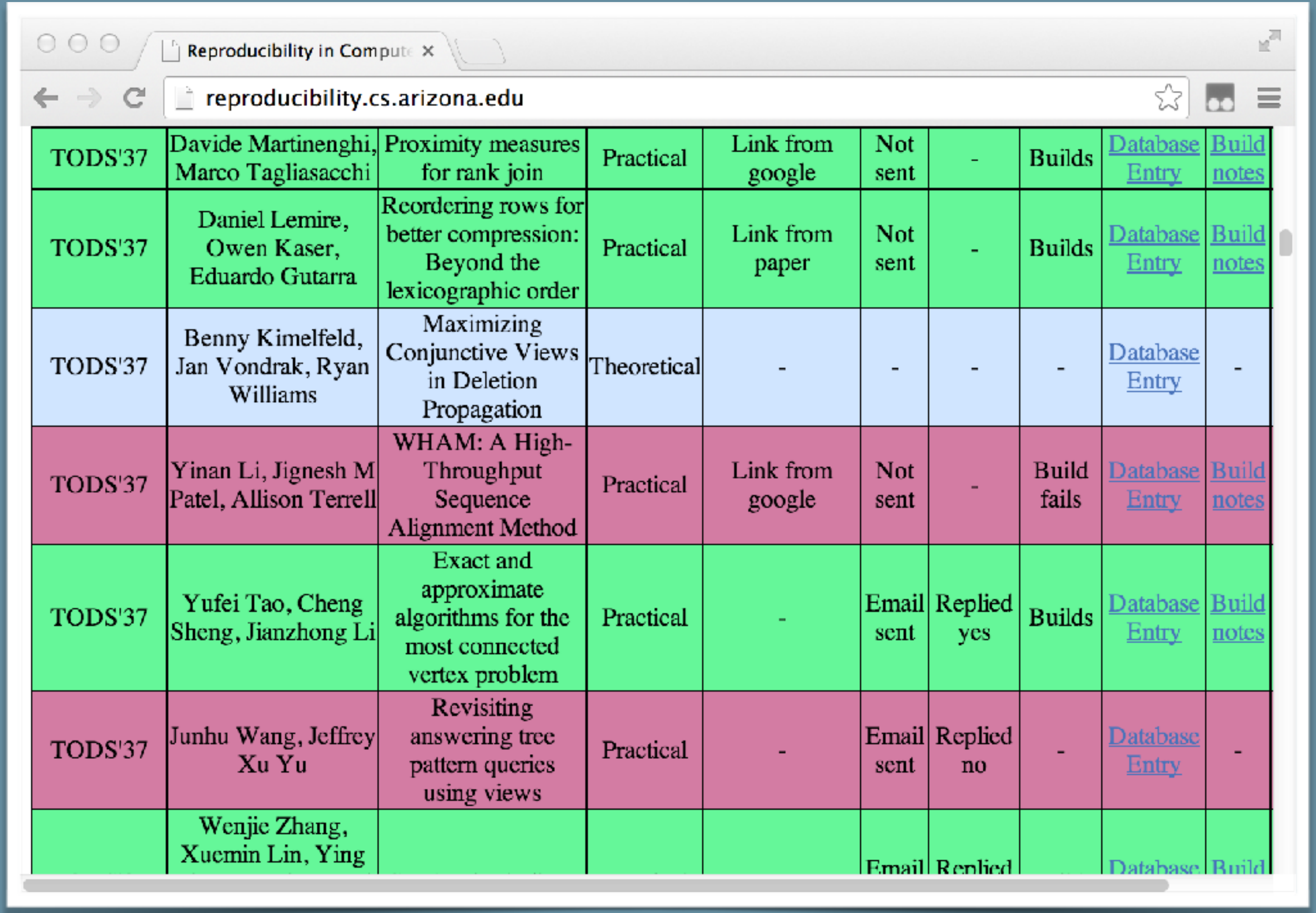
https://cacm.acm.org/blogs/blog-cacm/123611-the-nastiness-problem-in-computer-science/fulltext

What Happened Next?

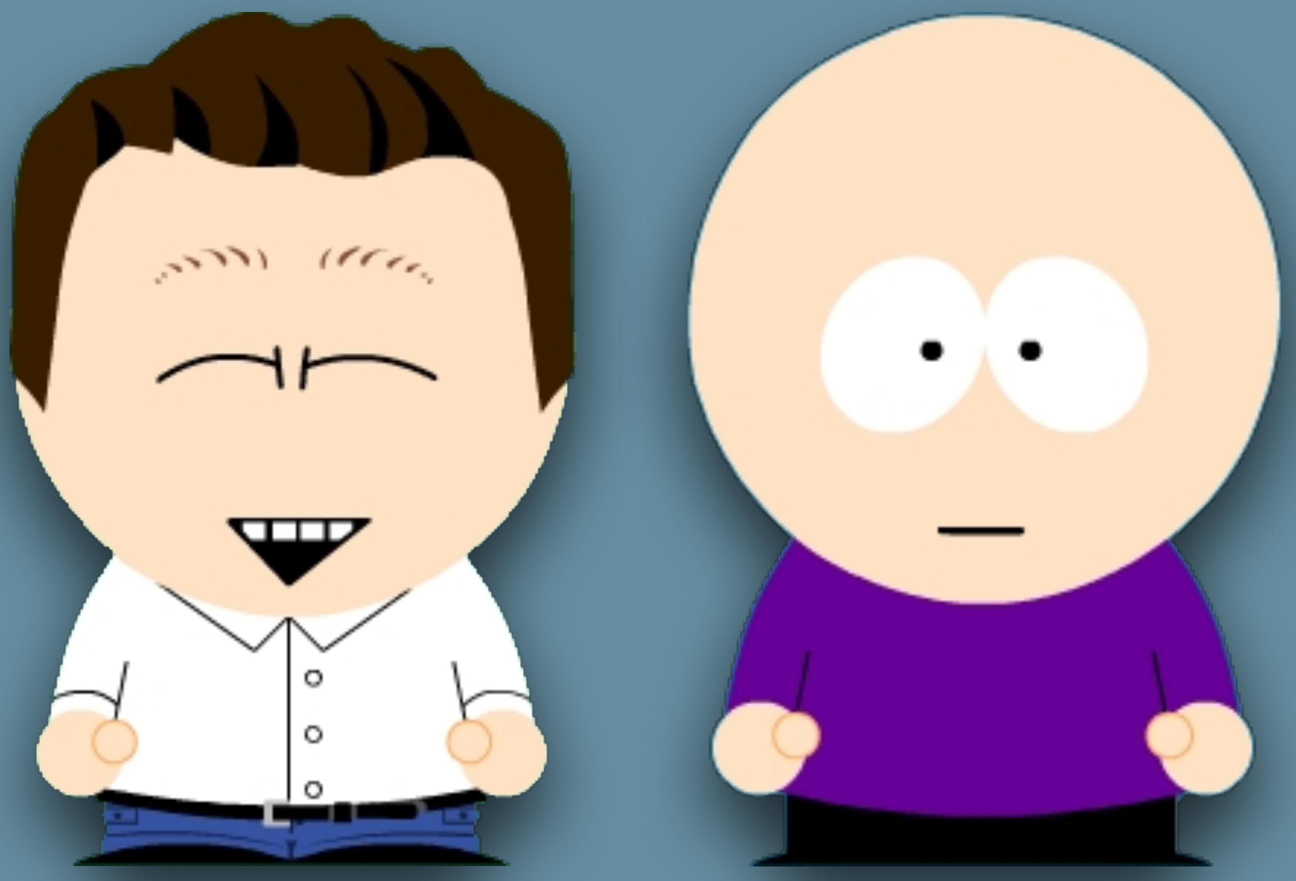




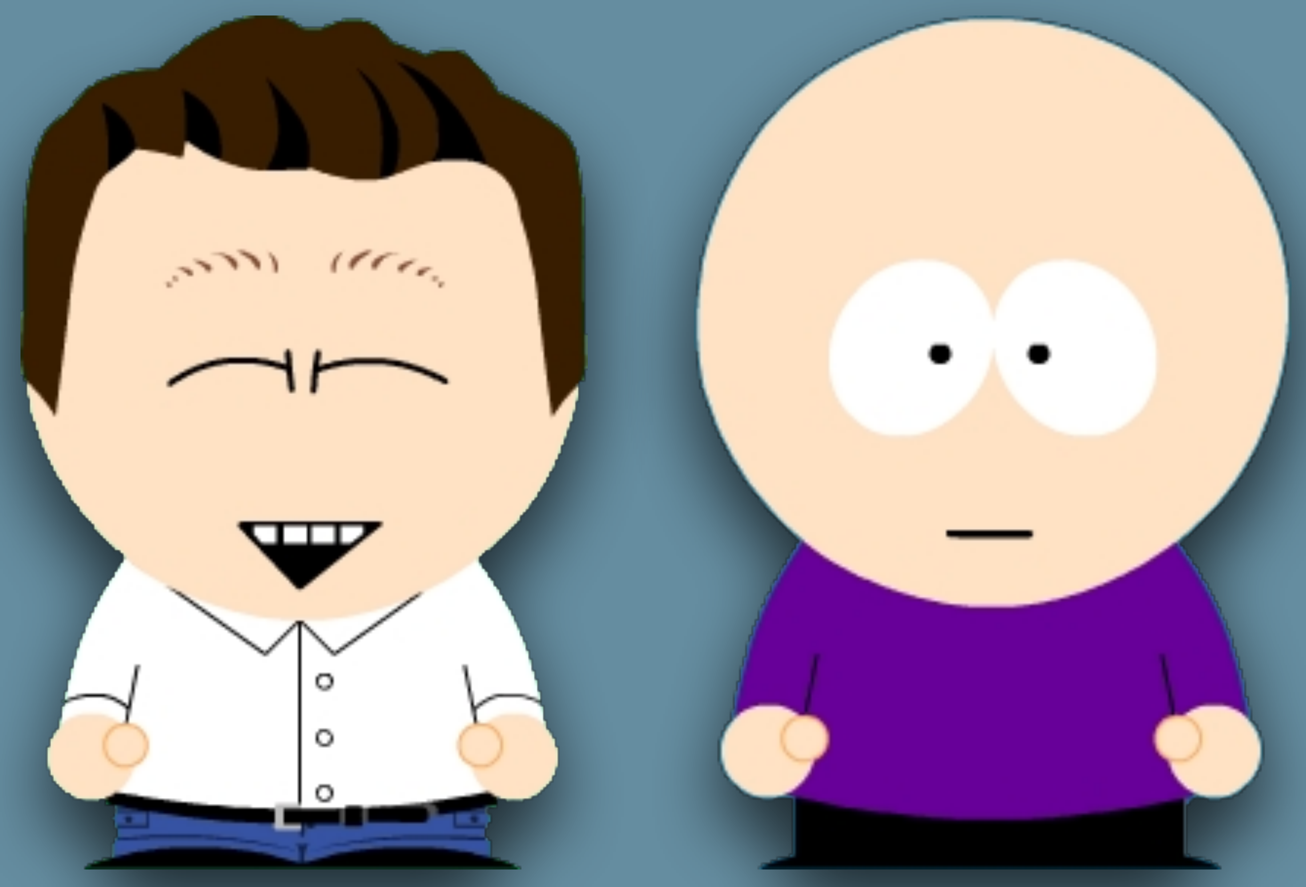
Submitted Paper



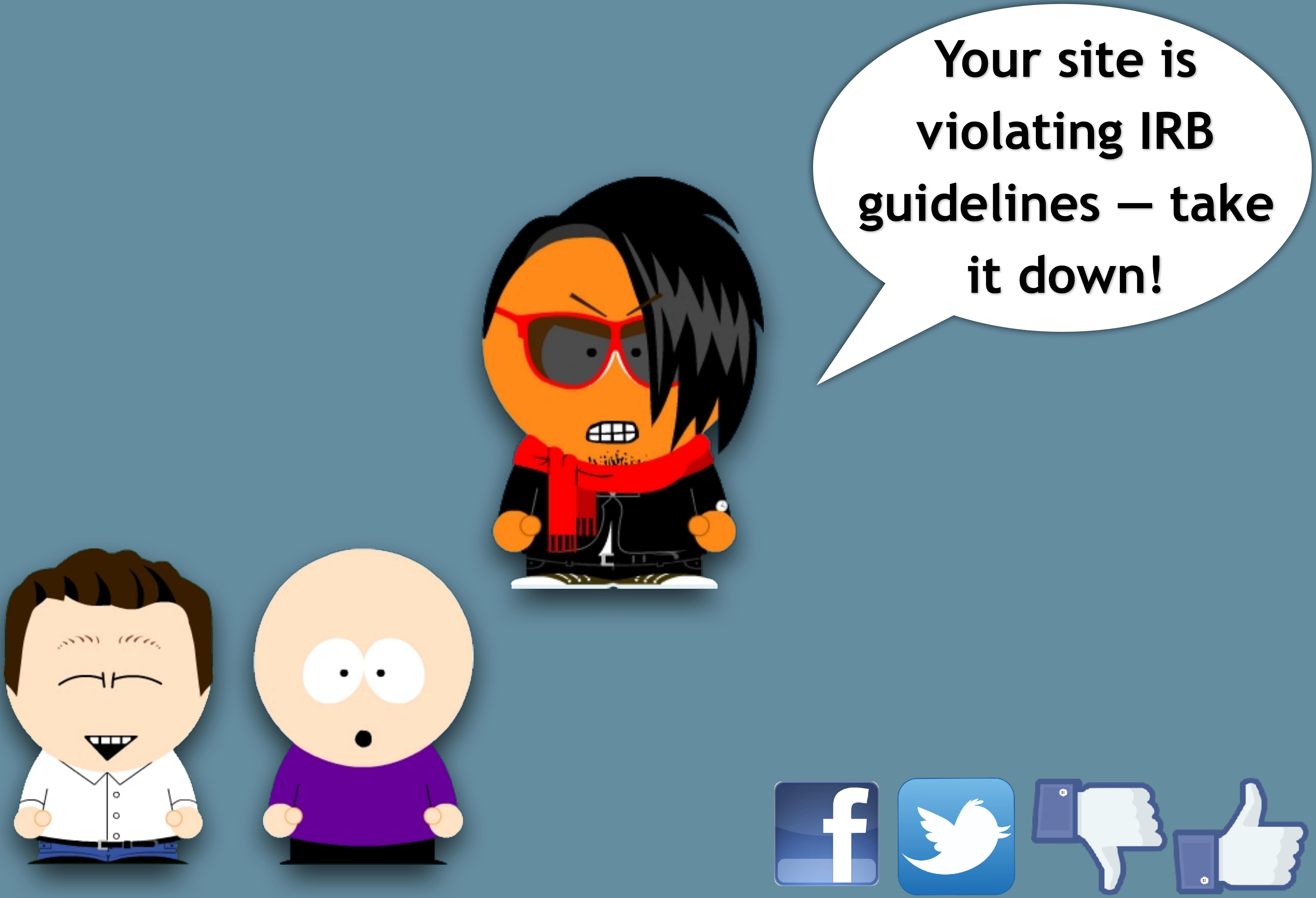
TODS'37	Davide Martinenghi, Marco Tagliasacchi	Proximity measures for rank join	Practical	Link from google	Not sent	-	Builds	Database Entry	Build notes
TODS'37	Daniel Lemire, Owen Kaser, Eduardo Gutarra	Reordering rows for better compression: Beyond the lexicographic order	Practical	Link from paper	Not sent	-	Builds	Database Entry	Build notes
TODS'37	Benny Kimelfeld, Jan Vondrak, Ryan Williams	Maximizing Conjunctive Views in Deletion Propagation	Theoretical	-	-	-	-	Database Entry	-
TODS'37	Yinan Li, Jignesh M Patel, Allison Terrell	WHAM: A High-Throughput Sequence Alignment Method	Practical	Link from google	Not sent	-	Build fails	Database Entry	Build notes
TODS'37	Yufei Tao, Cheng Sheng, Jianzhong Li	Exact and approximate algorithms for the most connected vertex problem	Practical	-	Email sent	Replied yes	Builds	Database Entry	Build notes
TODS'37	Junhu Wang, Jeffrey Xu Yu	Revisiting answering tree pattern queries using views	Practical	-	Email sent	Replied no	-	Database Entry	-
	Wenjie Zhang, Xuemin Lin, Ying				Email	Replied		Database	Build



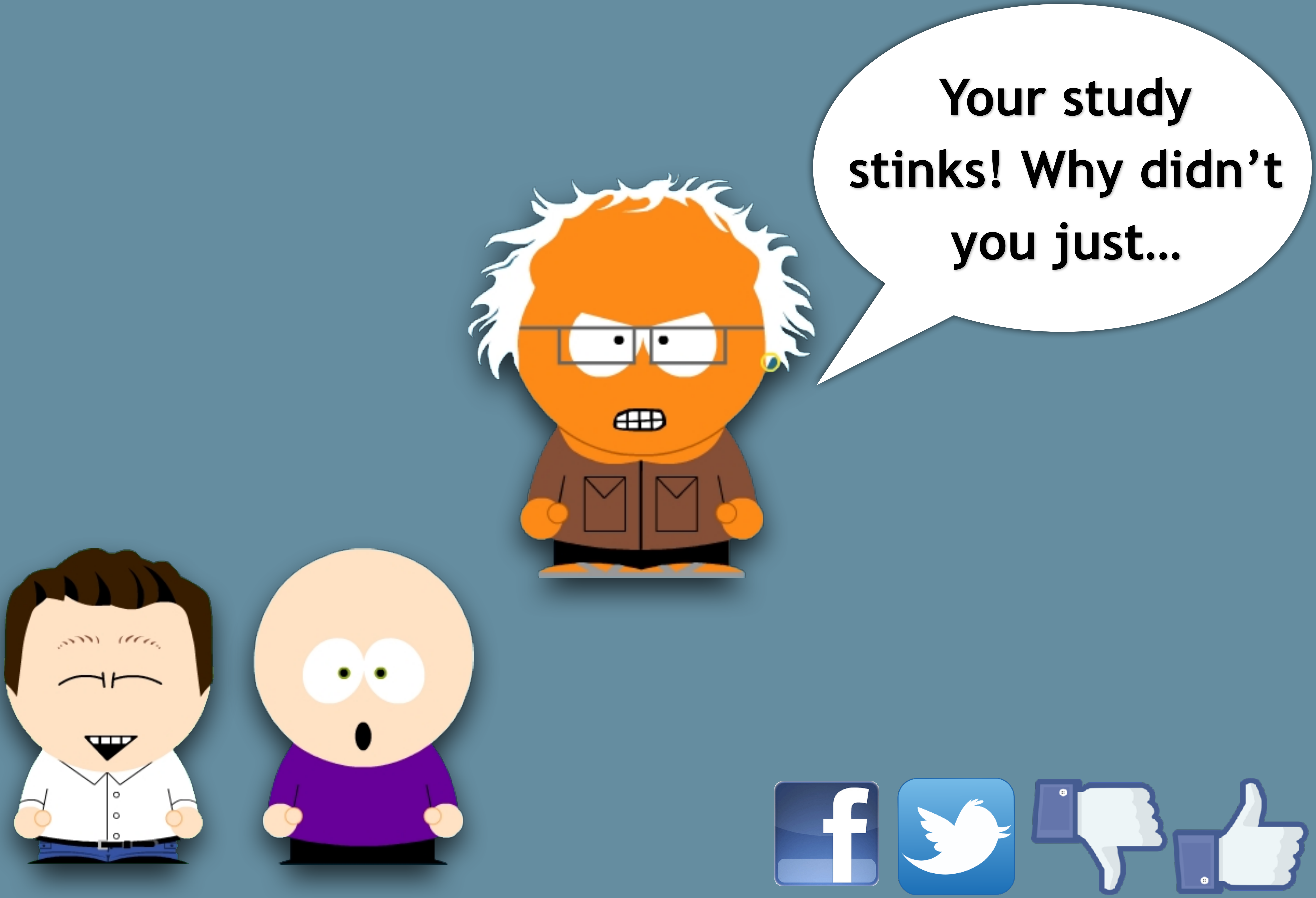
Hate Us on Facebook!



Hate Us on Facebook!

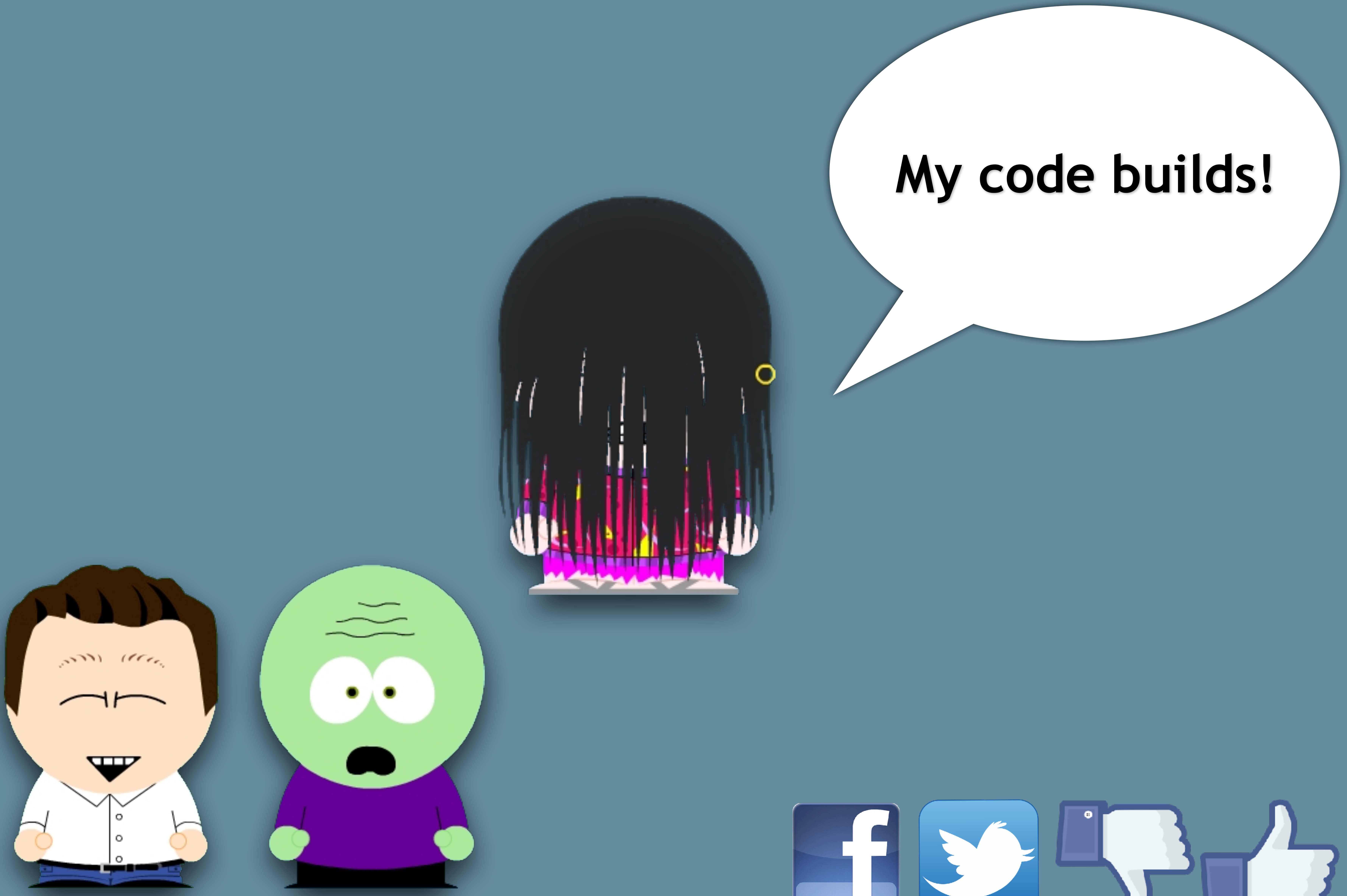


Hate Us on Facebook!



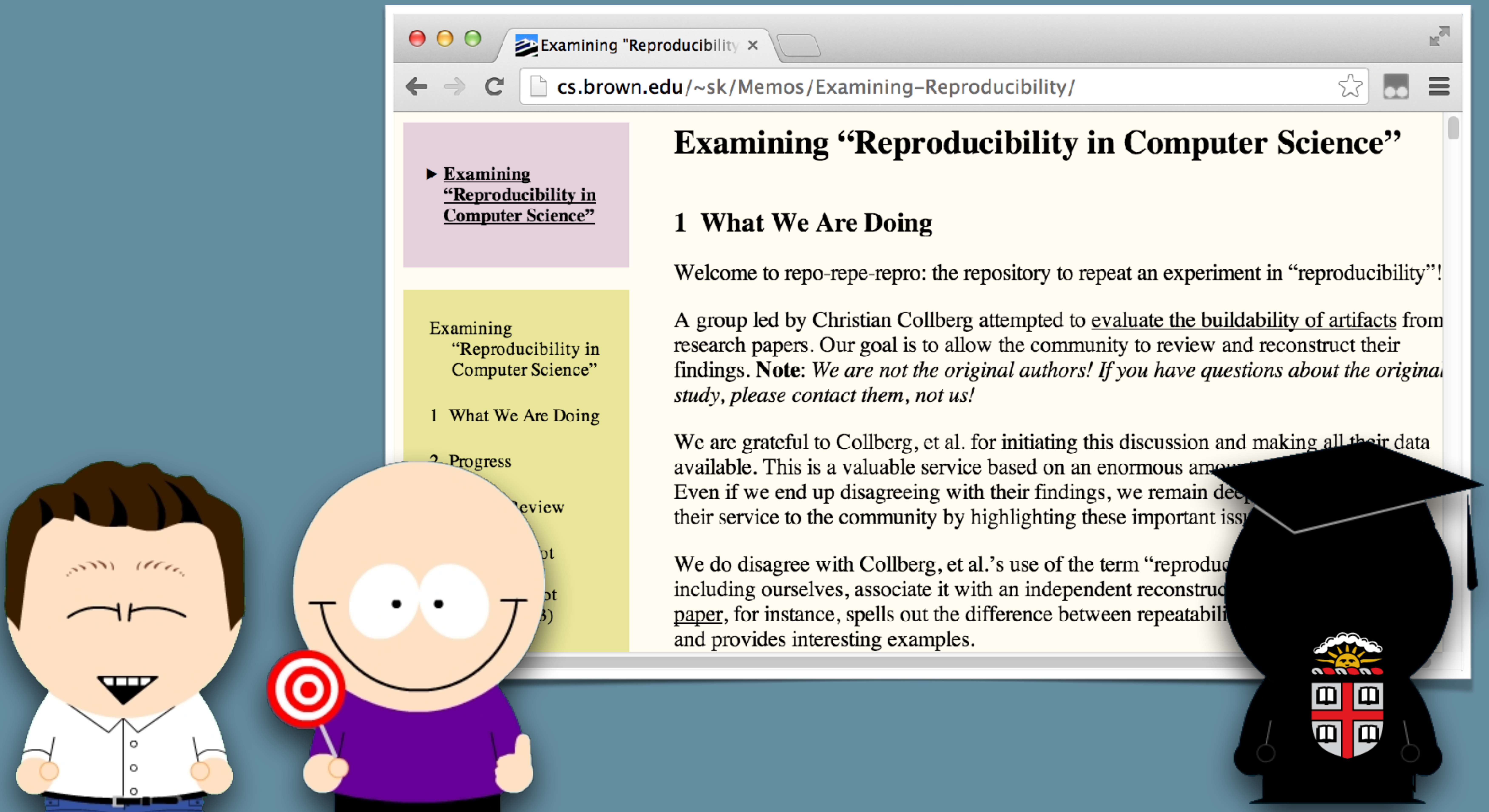
Hate Us on Facebook!

Hate Us on Facebook!

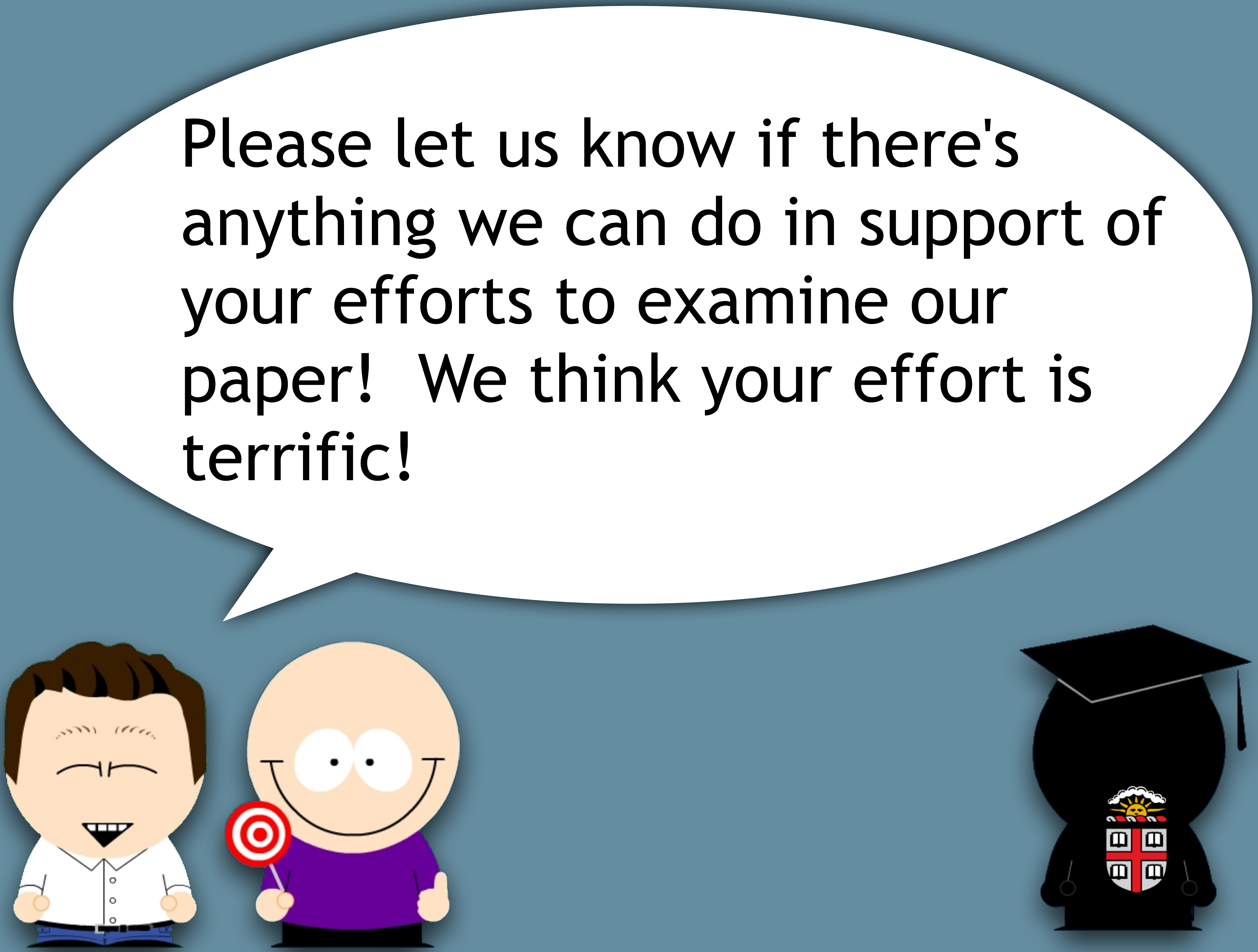


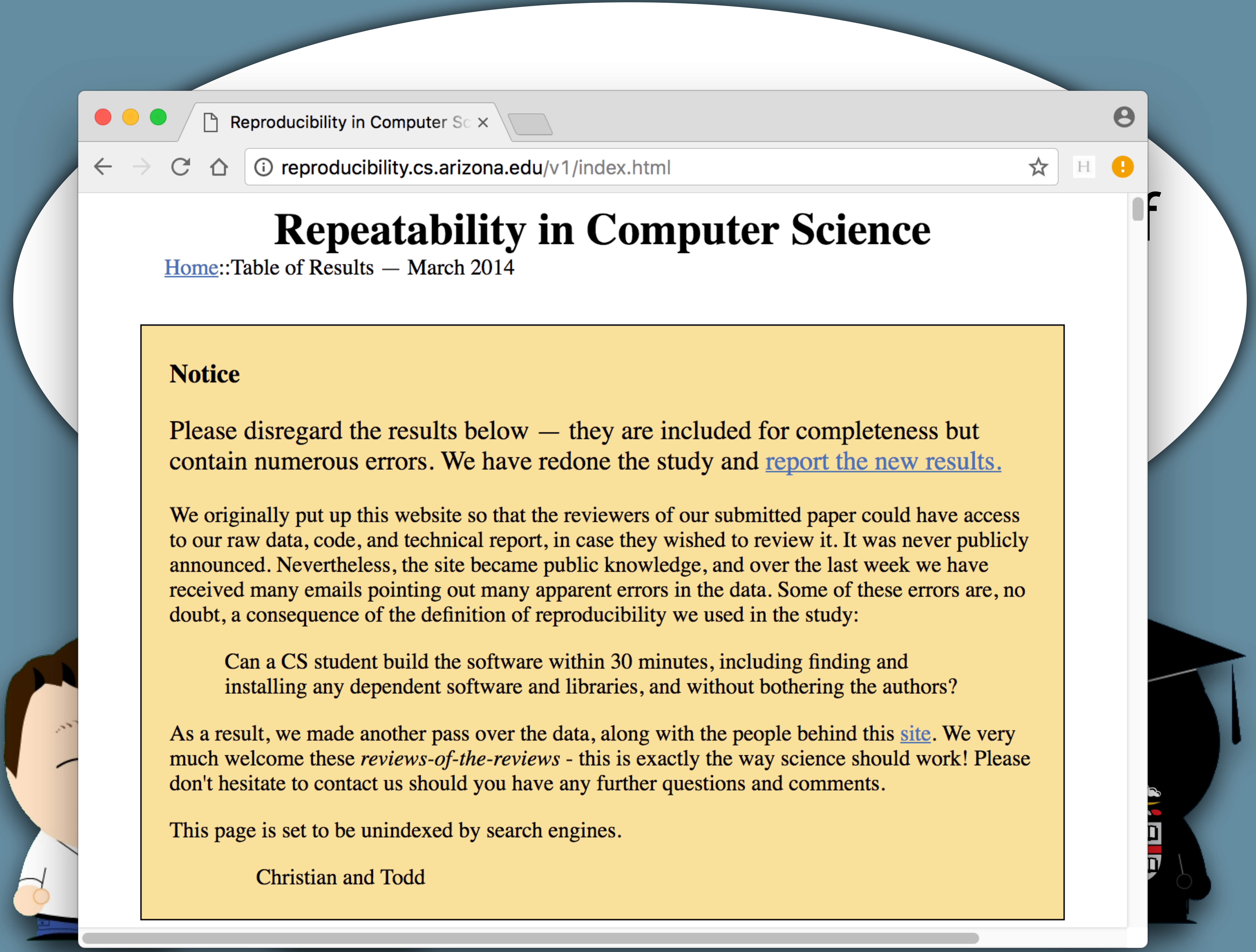
Hate Us on Facebook!

Turnabout is Fair Play!



http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/


Please let us know if there's anything we can do in support of your efforts to examine our paper! We think your effort is terrific!



Repeatability in Computer Science

Home::Table of Results — March 2014

Notice

Please disregard the results below — they are included for completeness but contain numerous errors. We have redone the study and report the new results.

We originally put up this website so that the reviewers of our submitted paper could have access to our raw data, code, and technical report, in case they wished to review it. It was never publicly announced. Nevertheless, the site became public knowledge, and over the last week we have received many emails pointing out many apparent errors in the data. Some of these errors are, no doubt, a consequence of the definition of reproducibility we used in the study:

> Can a CS student build the software within 30 minutes, including finding and installing any dependent software and libraries, and without bothering the authors?

As a result, we made another pass over the data, along with the people behind this site. We very much welcome these *reviews-of-the-reviews* - this is exactly the way science should work! Please don't hesitate to contact us should you have any further questions and comments.

This page is set to be unindexed by search engines.

Christian and Todd

f A group of independent researchers set out to verify our build results through a crowd-sourced effort; http://cs.brown.edu/˜sk/Memos/Examining-Reproducibility

To encourage repeatable re
repeatability engineering a
commitments to sharing re

BY CHRISTIAN COLLBERG AND TODD A. PROEBSTING

Repeatability in Computer Systems Research

backed up, we made a second ORA request, this time for the email messages among the authors, hoping to trace the whereabouts of the source code. The legal department first responded with: "… the records will not be produced pursuant to [ORA sub-clause]." When we pointed out reasons why this clause does not apply, the university relented but demanded $2,263.66 " … to search for, retrieve, redact and produce such records." We declined the offer.

We instead made a Freedom of Information Act request to the National Science Foundation for the funded grant proposals that supported the research. In one, the principal investigator wrote, "We will also make our data and software available to the research community when appropriate." In the

IN 2012, WHEN reading a paper f
computer security conference, w
there is a clever way to defeat th
in the paper, and, in order to sh
the authors (faculty and graduat
ranked U.S. computer science d
for access to their prototype syst
no response. We thus decided t
algorithms in the paper but soo
obstacles, including a variable u
function defined but never used
formula that did not typecheck. We asked the authors for clarification and received a single response: "I unfortunately have few recollections of the work … "

We next made a formal request to the university for the source code under the broad Open Records Act (ORA) of the authors' home state. The university's

- many challenges, so funding agencies should provide support for the engineering resources necessary to enable repeatable research.
- To incentivize authors to share their research artifacts, publishers should require pre-publication declarations from authors specifying their commitment to sharing code and data.

ILLUSTRATION BY ANTHONY FREDA

Acknowledgments

We would like to thank Saumya Debray, Shriram Krishnamurthi, Alex Warren, and the anonymous reviewers for valuable input.

ence

are included for completeness but
the study and report the new results.

could have access
was never publicly
week we have
these errors are, no

ng and
he authors?

this site. We very
is exactly the way science should work! Please
rther questions and comments.

Christian and Todd

contributed articles

A group of independent researchers set out to verify our build results through a crowd-sourced effort; http://cs.brown.edu/~sk/Memos/Examining-Reproducibility

To encourage repeatable re
repeatability engineering a
commitments to sharing re

BY CHRISTIAN COLLBERG AND TODD A. PROEBSTING

Repeatability in Computer Systems Re

backed up, we made a second ORA request, this time for the email messages among the authors, hoping to trace the whereabouts of the source code. The legal department first responded with: "... the records will not be produced pursuant to [ORA sub-clause]." When we pointed out reasons why this clause does not apply, the university relented but demanded $2,263.66 "... to search for, retrieve, redact and produce such records." We declined the offer.

IN 2012, W
computer
there is a
in the pap
the autho
ranked U.
for access
no respon
algorithm
obstacles,
function
formula t
for clarifi
unfortuna
We next made a formal request to the university for the source code under the broad Open Records Act (ORA) of the authors' home state. The university's

research artifacts, publishers should require pre-publication declarations from authors specifying their commitment to sharing code and data.

Repeatability and Benefaction in Computer Systems Research

A Study and a Modest Proposal

University of Arizona TR 14-04

Christian Collberg collberg@gmail.com
Todd Proebsting proebsting@cs.arizona.edu
Alex M Warren amwarren@email.arizona.edu

Christian and Todd

ShriramKrishnamurthi
@ShriramKMurthi
Follow

They did *crap* work, would not admit to when caught out and even pretended it hadn't happened.

https://twitter.com/ShriramKMurthi/status/863462366226370561

…these researchers have done a disservice to science by publishing a study they knew to be **horse manure**, and then piling more **bull crap** on it when caught … they are simply trying to build a reputation off a problem they don't really care to solve …

https://www.facebook.com/jvitekjr/posts/10155809013435351

To the University of Arizona Institutional Review Board:

Revoke their IRB permission!

1. Their deception study was bad
   – I don’t trust them!

1. Their deception study was bad
   – I don't trust them!
2. They're violating my privacy!

FindResearch.org

The authors

- *have*
- *have not*

verified

1. Their deception study was bad
   – I don't trust them!
2. They're violating my privacy!
3. They're spying on my computer!

FindResearch.org

The authors
- *have*
- *have not*

verified

3rd Law of Artifact Sharing

(Mother's Law)

Without a culture of respectful academic interchange, where failure is seen as an accepted part of the progression of science, sharing will not become default behavior.

Risks  Rewards

Risks  Rewards

Credibility: They may trust your work more when they can try it.

Risks  Rewards

Credibility: They may find bugs and not trust your results.

Credibility: They may trust your work more when they can try it.

Risks

Rewards

Credibility: They may find bugs and not trust your results.

Credibility: They may trust your work more when they can try it.

Visibility: They may notice your work when they can build on it.

Risks

Rewards

Credibility: They may find bugs and not trust your results.

ROI: They may ignore your code in spite of your efforts to share.

Credibility: They may trust your work more when they can try it.

Visibility: They may notice your work when they can build on it.

2nd Law of Artifact Sharing
(ROI)

The root of the scientific transparency problem is sociological, not technological: we do not share solid artifacts because there is little professional glory to be gained from doing so.

Recommendations for VISSOFT

Recommendations for VISSOFT

1. Agree on a checklist

- ☑ Share everything
- ☑ Document software you can't include
- ☑ Clearly link paper to artifact
- ☑ Include external code
- ☑ Ensure availability
- ☑ Use permanent email addresses

Recommendations for VISSOFT

1. Agree on a checklist
2. Describe experiments

Recommendations for VISSOFT

1. Agree on a checklist
2. Describe experiments
3. Require sharing statement

CSET'18 CFP

… include in the paper an artifact sharing statement describing whether some or all of the artifacts will be made available to the community … This statement should be present during both submission and in the final version of the paper. … while sharing may be taken into account by reviewers, it is not a requirement for acceptance.

Recommendations for VISSOFT

1. Agree on a checklist
2. Describe experiments
3. Require sharing statement
4. Ensure consistency between paper and artifact:

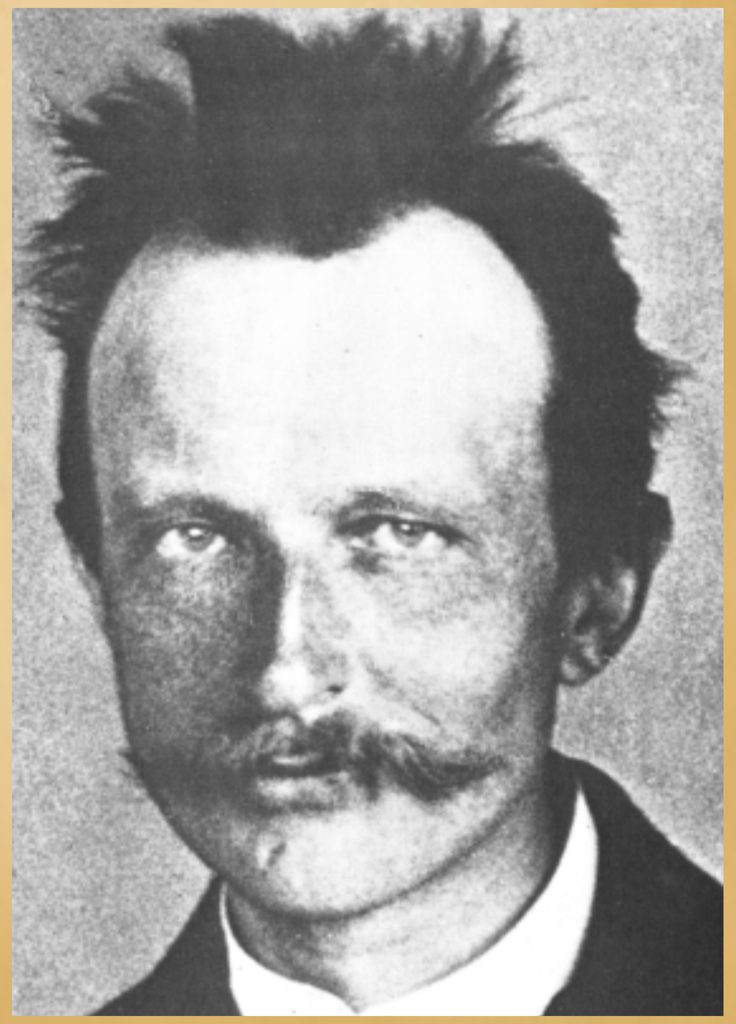
Camera-Ready: July 31, 2019

Artifact Submission: July 8, 2019

1st Law of Artifact Sharing

(Corollary to Max Planck's Quip)

Scientific transparency advances one funeral at a time.




Thank you!