Vers une science reproductible

Roberto Di Cosmo
Software Heritage
INRIA and IRIF

roberto@dicosmo.org

December 6th, 2017
Outline

1. The Science Crisis
2. The state of Software reproducibility
3. The Software Heritage initiative
4. Conclusion
Inconsistencies all around us

What causes cancer?

Is everything we eat associated with cancer?

**Inconsistency** an incompatibility between two propositions that cannot both be true

SEPT2 is Septin 2 or September 2nd?

Gene name errors are widespread in the scientific literature Zie-

**Corruption** The process by which a computer database or program becomes debased by alteration or the introduction of errors
And it gets worse!

Doctored data?

Federal prosecutors have launched a gigantic fraud case against Duke University, North Carolina, accusing Duke University of embossing $290 million in federal research grants, by presenting doctored data with their grant applications. On a Friday in March 2013, a researcher working in the lab of a prominent pulmonary scientist at Duke University in Durham, North Carolina, was arrested on charges of embezzlement. The researcher, biologist Edna Petit-Kant, later pled guilty to embezzling more than $25,000 from the Duke University Health System, buying merchandise from Amazon, Walmart, and Target—even taking receipts to legitimize her purchases. A state judge ultimately levied a fine, and sentenced her to probation and community service. Then Petit-Kant’s troubles get worse. Read the rest here 13/03

What are drugs good for?

Non reproducible results ...

FIGURE 1 | Analysis of the reproducibility of published data in 67 in-house projects.

Roberto Di Cosmo
Towards reproducible science December 6th, 2017 3 / 23
We face a science crisis

"Sub-prime science"? (Nicholas Humprey)

- inconsistencies
- data corruption, fraud
- non reproducible findings… (picture from Nature, Sep. 2015)

The world starts noticing

October 2013

John Oliver, *Science* May 2016
How we built our scientific knowledge

The experimental method

- make an observation
- formulate an hypothesis
- set up an experiment
- formulate a theory

And then we reproduce and verify.

Reproducibility is the key

*non-reproducible single occurrences are of no significance to science*

_Karl Popper, The Logic of Scientific Discovery, 1934_
Reproducibility in the digital age

For an experiment involving software, we need
- open access to the scientific article describing it
- open data sets used in the experiment
- source code of all the components
- environment of execution
- stable references between all this

Remark

The first two items are already widely discussed!

... what about software?
Software is an essential component of modern scientific research

Top 100 papers (Nature, October 2014)

 [...] the vast majority describe experimental methods or software that have become essential in their fields.

http://www.nature.com/news/the-top-100-papers-1.16224
Pressure to make research code available is now raising

Evaluation of software artefacts (optional)

- tools are usable, in line with expectations
- started as a contest in 2011 (ESEC/FSE) (winner Vouillon and Di Cosmo)
- now going mainstream: POPL’17, POPL’16, ECOOP’16, OOPSLA’16, CGO’16, VISSOFT’16, PLDI’16, CGO’15, PPoPP’15, VISSOFT’15, ISSTA’15, OOPSLA’15, PLDI’15, POPL’15, CAV’15, ECOOP’15, FSE’15, ISSTA’14, OOPSLA’14, PLDI’14, ECOOP’14, FSE’14, SAS’13, OOPSLA’13, ECOOP’13, FSE’13, FSE’11
Use the Source, Luke!

Repeatability, replicability, reproducibility: having (all) the source code used in an experiment is needed for the first two. Some claim it is not worth the effort “Replicability is not Reproducibility: Nor is it Good Science”, Chris Drummond, ICML 2009

Sure, diversity is important, but:

- Source code is like the proof used in a theorem: can we really accept \textit{Fermat statements} like “the details are omitted due to lack of space”?

- modern complex systems make even the simplest experiment depend on a wealth of components and configuration options

- access to all the source code is not just necessary to \textit{replicate}, it is also useful to \textit{evolve and modify}, to \textit{build new experiments} from the old ones
"The source code for a work means the preferred form of the work for making modifications to it."

— GPL Licence

---

**Hello World**

**Program (excerpt of binary)**

- `4004e6: 55`
- `4004e7: 48 89 e5`
- `4004ea: bf 84 05 40 00`
- `4004ef: b8 00 00 00 00`
- `4004f4: e8 c7 fe ff ff`
- `4004f9: 90`
- `4004fa: 5d`
- `4004fb: c3`

**Program (source code)**

```c
/* Hello World program */

#include<stdio.h>

void main()
{
    printf("Hello World");
}
```

---

Roberto Di Cosmo
Towards reproducible science  December 6th, 2017  10 / 23
Software Source Code is *special*

Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

“Programs must be written for people to read, and only incidentally for machines to execute.”

**Quake 2 source code (excerpt)**

```c
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalves = 1.5F;
    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = y * ( threehalves - ( x2 * y * y ) ); // 1st iteration
    // y = y * ( threehalves - ( x2 * y * y ) ); // 2nd iteration, this can be removed
    return y;
}
```

**Net. queue in Linux (excerpt)**

```c
/*
 * SFB uses two B[1][N] : L x N arrays of bins (L levels, N bins per level)
 * This implementation uses L = 8 and N = 16
 * This permits us to split one 32bit hash (provided per packet by rxhash or
 * external classifier) into 8 subhashes of 4 bits.
 */
#define SFB_BUCKET_SHIFT 4
#define SFB_NUMBuckets (1 << SFB_BUCKET_SHIFT) /* N bins per Level */
#define SFB_BUCKET_MASK (SFB_NUMBuckets - 1)
#define SFB_LEVELS (32 / SFB_BUCKET_SHIFT) /* L */

/* SFB algo uses a virtual queue, named "bin" */
struct sfb_bucket {
    u16 qlen; /* length of virtual queue */
    u16 p_mark; /* marking probability */
};
```

Len Shustek, Computer History Museum

“Source code provides a view into the mind of the designer.”

Roberto Di Cosmo

Towards reproducible science December 6th, 2017
Apollo 11 Guidance Computer (~60.000 lines), 1969

"When I first got into it, nobody knew what it was that we were doing. It was like the Wild West."

Margaret Hamilton

Linux Kernel

... now in your pockets!

are we taking care of all this?
Collberg’s report from the trenches

Analysis of 613 papers

- 8 ACM conferences: ASPLOS’12, CCS’12, OOPSLA’12, OSDI’12, PLDI’12, SIGMOD’12, SOSP’11, VLDB’12
- 5 journals: TACO’9, TISSEC’15, TOCS’30, TODS’37, TOPLAS’34

all very practical oriented

The basic question

can we get the code to build and run?

The workflow
This can be debated (see http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/), but...

... that’s a whopping 81% of non reproducible works!
The reasons (or, “the dog ate my program”)

Why so much software fails to pass the test?

Many issues, nice anecdotes, and it finally boils down to

- Availability
- Traceability
- Environment
- Automation (do you use continuous integration?)
- Documentation
- Understanding (including free/open source software)

The first two are important software preservation issues

Yes, code is fragile: it can be destroyed, and we can lose trace of it
Outline

1. The Science Crisis
2. The state of Software reproducibility
3. The Software Heritage initiative
4. Conclusion
Our mission

Collect, preserve and share the source code of all the software that is available

Past, present and future

Preserving the past, enhancing the present, preparing the future
Supporting more accessible and reproducible science

A global library referencing all software used in all research fields

- completes the infrastructure for Open Access in science
- provides intrinsic persistent identifiers needed for scientific reproducibility
- enables large scale, verifiable software studies
The Knowledge Conservancy Magic Triangle

Legenda (links are important!)

- articles: ArXiv, HAL, …
- data: Zenodo, …
- software: "Software Heritage to the rescue"
Brief summary

- researchers do all the work... and publishers lock the results behind paywalls
- public money is wasted in subscriptions... and subtracted to research
- we reacted with Open access... and got caught in the "gold open access" trap

The essential missing bit in the conversation

Art. L. 131-4.
La cession par l’auteur de ses droits sur son oeuvre peut être totale ou partielles. Elle doit comporter au profit de l’auteur la participation proportionnelle aux recettes provenant de la vente ou de l’exploitation.

Yes, it’s the greatest copyright violation in history!

Authorities refuse to see... but spend 100+Millions in programs like ISTEX
Archive coverage

Source files
3,718,806,509

Commits
853,277,241

Projects
65,546,644

~150 TB blobs, ~5 TB database (as a graph: ~7 B nodes + ~60 B edges)

Our sources

- GitHub — full, up-to-date mirror
- Debian — automation in progress; GNU
- Gitorious, Google Code — processing (Archive Team & Google)
- Bitbucket — WIP

The richest source code archive already, … and growing daily!
Going global

April 3rd, 2017: landmark Inria Unesco agreement...

https://www.softwareheritage.org/blog

September 28th, 2017

Mauritius Call on information access
An unique opportunity

Library of Alexandria of code

Take *urgent* action to
- recover the past
- structure the future

A CERN for Software

Build a *common infrastructure*
- software research, better science
- for society as a whole

Come in, we’re open www.softwareheritage.org

tons of research problems, and our code: forge.softwareheritage.org
Age of responsibility

Technology reshapes society

- reproducitibility, replicability, science
- trust, justice, freedom, transparency

Code is law (Lawrence Lessig)

- safety, security, privacy, anonymity
- ethical and social challenges

We need you!

- value "responsible" research
- contribute to Free Software
- ask questions, join the conversation

Giambattista Vico (1688 - 1744)

Conoscere, è saper fare