Reproducibility in Parallel and Distributed Computing: Challenges, State of the Practice, Limitations, and Opportunities

Quentin Guilloteau, Florina M. Ciorba

Reproduciblity in Parallel and Distributed Computing: Challenges, State of the Practice, Limitations, and Opportunities

University of Basel

Quentin Guilloteau and Florina M. Ciorba

Department of Mathematics and Computer Science

The Question of Reproducibility

Researchers perform computations on parallel & distributed computing systems.

High performance ☺
Distributed ☺
Shared ☺

Difficult to control ☹

⇓

Is Reproducibility Achievable?

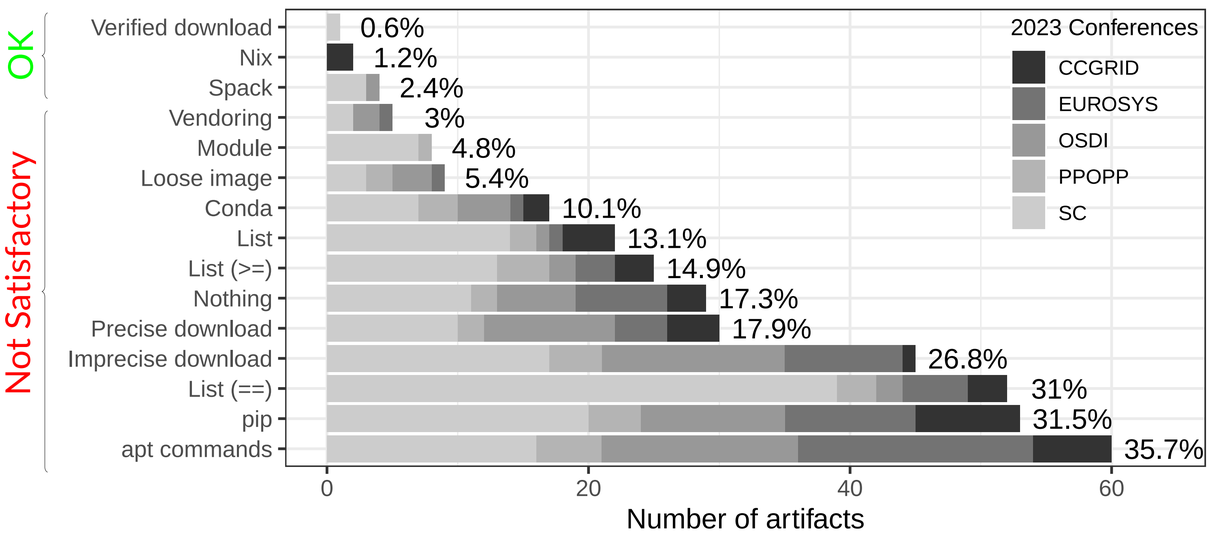
↪ What can researchers control? **The software environments.**

Community's Answer: Artifact Evaluation

Upon article's acceptance, authors submit **code** and other **artifacts** to be **evaluated and reproduced** by reviewers. Reproducible artifacts are then rewarded with **badges** [1]. ⇝

Current State of the Practice

We surveyed 296 papers of top conferences in parallel and distributed computing from 2023 and recorded how their software environments were described [2]:



↪ **Finding:** 17% of the artifacts did not describe their software environments!

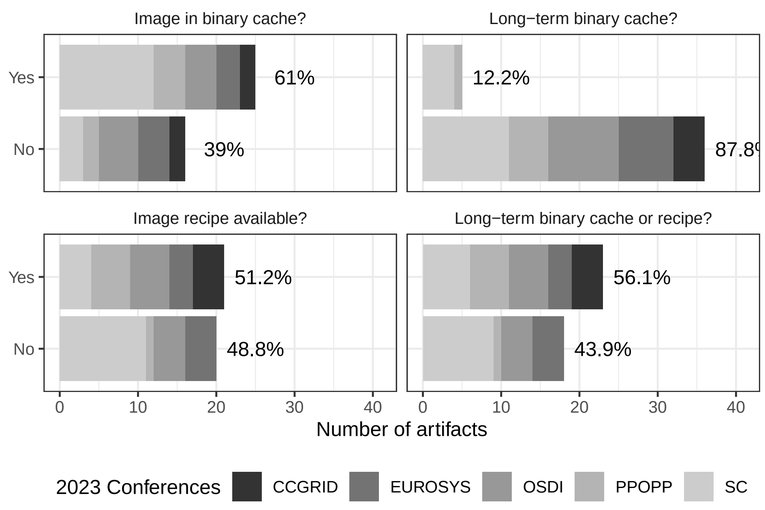
↪ **Finding:** Most research artifacts **are not shared in reproducible software environments**.

Common Tools: Environment Modules

- Managed by system administrators ⇝ can **disappear** at any time ☹
- System-specific ⇝ **not portable** across computing platforms ☹
- Side effects ⇝ unloading may not reset environment to previous state ☹

Common Tools: Containers

Finding: The software environments of 12% of artifacts use containers.



Full paper [2] available:

↪ Half of these artifacts **do not share the container recipe**.

⇝ How can then the *same* container image be rebuilt?

Typical container recipe:

Can you find all **reproducibility issues** with this `Dockerfile`?

```
FROM ubuntu
RUN apt-get update -y && apt-get install -y packageA ...
RUN git clone https://github.com/me/my-repo.git
WORKDIR my-repo
RUN curl -L https://tinyurl.com/patchrepo -o fix.patch
RUN git apply fix.patch
RUN make
ENTRYPOINT ["./my_bin"]
```

docker

↪ Container recipes are based on **non-reproducible** tools and practices.

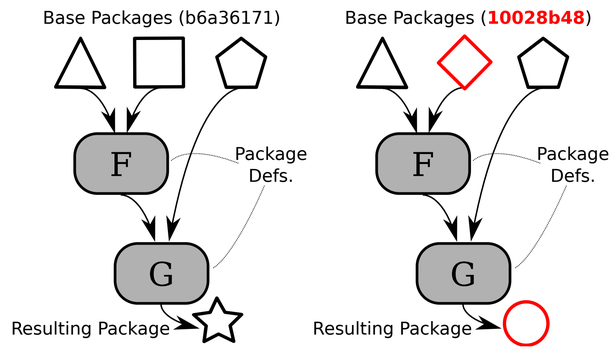
⇝ Most containers are not reproducible ☹

Limitations in the State of the Practice

Limited reproducibility and nearly impossible to extend existing work, to introduce a specific variation (*e.g.*, different compiler version) ⇝ ☹

A Solution: Functional Package Managers (FPMs)

- Nix [3], Guix [4]
- Capture the **entire** software stack
- packages ▸ functions
  - inputs ▸ dependencies
  - body ▸ commands to execute
- Base packages available on Git
- **Precise introduction of variations**



Storing packages: the FPMs way

```
/nix/store
├── yaz7pyf0ah88g2v505l38n0f3wg2vzdj-glibc-2.37
│   └── lib
│       └── libc.so
├── y9zg6ryffgc5c9y67fcmfdkyyiivjzpj-glibc-2.27
│   └── lib
│       └── libc.so
└── nc5qbagm3wqfg2lv1gwj3r3bn88dpqr8-mypkg-0.1.0
    ├── bin
    │   └── myprogram
    └── lib
        └── libmylib.so
```

- Immutable, read-only directory
- Packages stored as `hash(inputs)-pkgname`
- Can store **several versions** of the same package
- **Precise** definition of the software environment

↪ FPMs solve most reproducibility issues but require a **change of practice** and incur a **steep learning curve** due to their new concepts in package management.

A Proposal: the *Artifacts Longevity* Badge

Proposed in [2], takes into account:

- How is the artifact shared (source code, data)
- How is the software environment captured & described
- Where & how have the experiments been executed

Artifacts Longevous acm [a]

[a] In discussion with ACM

↪ **Aim:** Reward work that can be reused in the **future** (by authors or others).

Next Steps

- Longitudinal study of the state of the practice
- Awarding longevity badges to artifacts that meet the requirements
- Future testing of artifacts receiving the longevity badge
- **Survey** how parallel and distributed computing systems are used in different research fields →

Take our survey!

Bibliography

[1] https://www.acm.org/publications/policies/artifact-review-and-badging-current

[2] Guilloteau Quentin, Ciorba Florina M., Poquet Millian, Goepp Dorian, and Richard Olivier. "Longevity of Artifacts in Leading Parallel and Distributed Systems Conferences: a Review of the State of the Practice in 2023." ACM Conference on Reproducibility and Replicability (REP 2024), 2024.

[3] Dolstra Eelco, Merijn De Jonge, and Eelco Visser. "Nix: A Safe and Policy-Free System for Software Deployment." LISA. Vol. 4., 2004.

[4] Courtès Ludovic and Ricardo Wurmus. "Reproducible and user-controlled software environments in HPC with Guix." Euro-Par 2015: Parallel Processing Workshops: Euro-Par 2015 International Workshops, Vienna, Austria, August 24-25, 2015.

DMI HPC

DAPHNE