AutoAppendix: Towards One-Click Reproducibility of Computational Artifacts

Chameleon User Meeting 2024

Klaus Krassnitzer

Nov 18, 2024

Institute of Science and Technology Austria

Outline

1. Motivation
2. AutoAppendix
3. Reproducibility Challenges and Proposed Solutions
4. Chameleon Cloud Jupyter Notebook Templates

Motivation

Supercomputing Conference (SC) Reproducibility Initiative

- Introduced at SC15
- Paper authors submit Artifact Description (AD) and Artifact Evaluation (AE) appendices
  - AD: Relation between contributions and artifacts, necessary information to reproduce results
  - AE (optional): Concrete reproduction instructions
- Authors apply for *reproducibility badges*
  - Artifacts Available
  - Artifacts Evaluated – Functional
  - Results Replicated
- Reproducibility Committee evaluates artifacts and awards badges
  - 8-hour evaluation period
  - Downgrades (e.g., from RR to AF) possible

Badge Verdicts at SC24

Applied / Verdict	Artifacts Available	Artifacts Functional	Results Replicated	Total
Artifacts Available	20	1	0	21
Artifacts Functional	**6**	6	1	13
Results Replicated	**7**	**14**	24	45
Total	33	21	25	79

Badge Verdicts at SC24

Applied / Verdict	Artifacts Available	Artifacts Functional	Results Replicated	Total
Artifacts Available	20	1	0	21
Artifacts Functional	**6**	6	1	13
Results Replicated	**7**	**14**	24	45
Total	33	21	25	79

Out of 45 applications for *Results Replicated* badge, **only 24 were successful**

Badge Verdicts at SC24

Applied / Verdict	Artifacts Available	Artifacts Functional	Results Replicated	Total
Artifacts Available	20	1	0	21
Artifacts Functional	**6**	6	1	13
Results Replicated	**7**	**14**	24	45
Total	33	21	25	79

Out of 45 applications for *Results Replicated* badge, **only 24 were successful**

How can this be improved?

AutoAppendix

AutoAppendix

- 2-month project conducted during the UCSC Summer of Reproducibility program
- Survey of the *state of reproducibility* of SC'24 papers.
- **Goals**:
  - Understand current challenges in achieving reproducibility
  - Identify best practices and potential for automation
- **Approach**: Select papers *without* special hardware requirements
  $\rightarrow$ Reproducibility depends on software stack and documentation

Paper Selection Criteria

- Select only from papers applying for *Results Replicated* badge
- No special hardware setup (e.g. specific clusters)
- Free software stack
- Experiments reproducible on *a single* ***Chameleon Cloud*** *node*
- Final Selection: **18 papers**

ID	Node Type	Site	Project Type
111	skylake	TACC	Simulator
174	skylake	TACC	Plotting
190	skylake	TACC	Plotting
191	icelake_r650	TACC	OpenMP
202	icelake_r650	TACC	Algorithm/Study
227	v100	UC	CUDA
332	gtx_6000	UC	CUDA
362	gtx_6000	UC	CUDA
368	icelake_r650	TACC	Framework
376	p100	TACC	CUDA
407	mi100	TACC	OpenCL
428	icelake_r650	TACC	Study
466	v100	UC	Study
467	p100	TACC	Framework
482	icelake_r650	TACC	Algorithm/Study
483	icelake_r650	TACC	Framework
496	icelake_r650	TACC	Algorithm
506	icelake_r650	TACC	Compiler

Reproducibility Challenges and Proposed Solutions

Challenge (I): Software Setup

Problem

- Failing builds due to missing dependencies
- Incompatible software versions
- Insufficient documentation

Challenge (I): Software Setup

Problem

- Failing builds due to missing dependencies
- Incompatible software versions
- Insufficient documentation

Solution Ideas:

- State operating system and software versions for all relevant components
- Use reproducible software environments
  - Containerization Solutions: Docker, Singularity, ...
  - Reproducible Builds: Spack, Guix, Nix

Challenges (II): Data Availability

Problem:

- Artifact repositories have limited storage, not sufficient for certain datasets (e.g. ML training data)

→ Data has to be stored externally

- Data may *not be available* anymore
- Download may be slow

Challenges (II): Data Availability

Problem:

- Artifact repositories have limited storage, not sufficient for certain datasets (e.g. ML training data)

→ Data has to be stored externally

  - Data may *not be available* anymore
  - Download may be slow

Solution Ideas:

- Ensure data availability, most importantly in the evaluation period
- Ensure reasonable download times and provide estimates

Challenges (III): Long-Running Operations

Problem:

- Experiment takes a long time, documentation is not clear about how long the operation should take
- No output $\rightarrow$ user does not know if the application is still running
- Too much output may cause performance issues/inconsistencies

Challenges (III): Long-Running Operations

Problem:

- Experiment takes a long time, documentation is not clear about how long the operation should take
- No output $\rightarrow$ user does not know if the application is still running
- Too much output may cause performance issues/inconsistencies

Solution Ideas:

- State expected run time on the *evaluation hardware* for all long-running steps
- If applicable, provide *progress output* to the user

Challenges: Summary

- **Software Setup**: Document software versions and use reproducible environments
- **Data Availability**: Ensure data is available and provide download estimates
- **Long-Running Operations**: State expected run times and provide progress output

Challenges: Summary

- **Software Setup**: Document software versions and use reproducible environments
- **Data Availability**: Ensure data is available and provide download estimates
- **Long-Running Operations**: State expected run times and provide progress output

These are only a few examples of the faced challenges, others include:

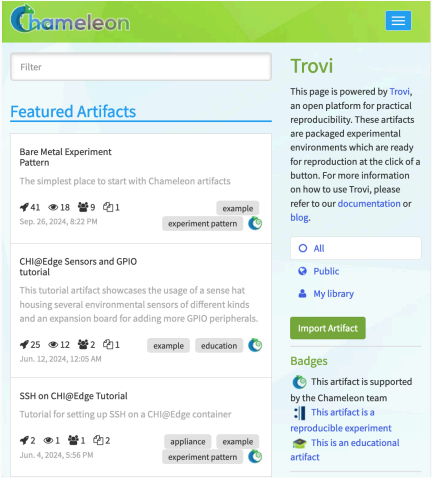
- **Documentation**: Clear and self-contained
- **Hardware Compatibility**: Reproducible on common hardware
- **Expected Results**: State expected behavior and (intermediate) results

Results were compiled into a set of **guidelines** and **best practices**

Chameleon Cloud Jupyter Notebook Templates

Chameleon Cloud Jupyter interface

- Allows for automated node provisioning and setup over `python-chi`
- Experiments can be executed fully automatically without manual intervention
- *Trovi* enables easy artifact sharing

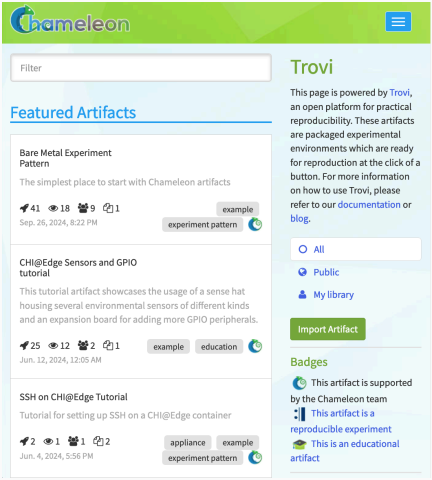


Chameleon Cloud Jupyter interface

- Allows for automated node provisioning and setup over `python-chi`
- Experiments can be executed fully automatically without manual intervention
- *Trovi* enables easy artifact sharing

→ **Very useful for artifact evaluation!**

- No need for manual node setup
- Transparency in software setup and steps



Jupyter Notebook Templates

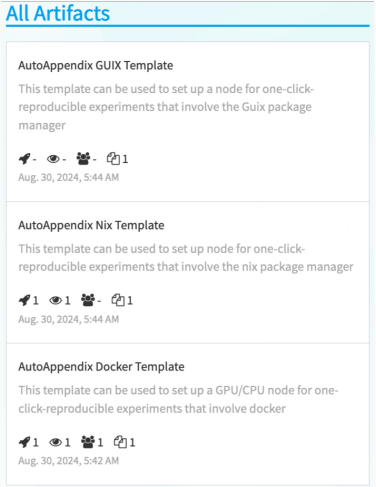
- Only 1 submission for SC24 used this approach

Jupyter Notebook Templates

- Only 1 submission for SC24 used this approach
- **How can we make this more accessible?**

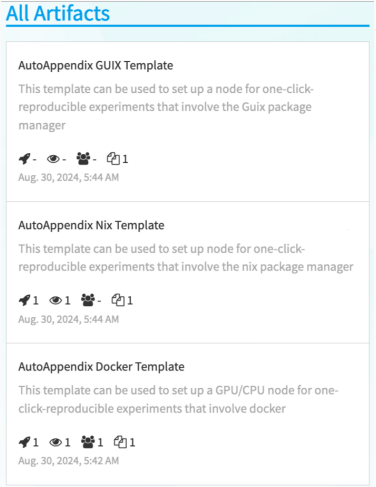
Jupyter Notebook Templates

- Only 1 submission for SC24 used this approach
- **How can we make this more accessible?**
- Three new Jupyter Notebook templates on *Trovi*
  - Docker
  - Nix
  - Guix



Jupyter Notebook Templates

- Only 1 submission for SC24 used this approach
- **How can we make this more accessible?**
- Three new Jupyter Notebook templates on *Trovi*
  - Docker
  - Nix
  - Guix
- **Features**:
  - Thorough documentation
  - Centralized Configuration
  - Sample Experiments



Summary

- **AutoAppendix** project aimed to identify challenges in achieving reproducibility
- **Guidelines** and **best practices** were compiled from the challenges encountered
- **Jupyter Notebook Templates** were created to encourage the use of Chameleon Cloud's Jupyter Interface
- **Future Work**:
  - Refine guidelines and best practices
  - Evaluate the impact of the templates and refine
  - Create templates for other software environments

Thank you for your attention!

Questions?

Find the AutoAppendix {Docker, Nix, Guix} Templates on *Trovi*:

`https://www.chameleoncloud.org/experiment/share/?filter=autoappendix`