EPiC Series in Computing

Volume 96, 2023, Pages 189–195

Proceedings of 10th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH23)

ARCH-COMP23 Repeatability Evaluation Report

Taylor T. Johnson[1]

Vanderbilt University,
Department of Computer Science,
Institute for Software Integrated Systems,
Nashville, TN, United States
taylor.johnson@vanderbilt.edu
http://www.TaylorTJohnson.com

Abstract

The repeatability evaluation for the 7th International Competition on Verifying Continuous and Hybrid Systems (ARCH-COMP'23) is summarized in this report. The competition took place as part of the workshop Applied Verification for Continuous and Hybrid Systems (ARCH) in 2023, affiliated with the 2023 Cyber-Physical Systems and Internet-of-Things Week (CPS-IoT Week). In its seventh edition, tools submitted artifacts through a new automated evaluation system and were synchronized with a Git repository for the repeatability evaluation and archiving, which were applied to solve benchmark instances through different competition categories. Due to procedural changes in execution through the automated system, fewer participants than in past iterations participated in the repeatability evaluation this year. The process was generally to submit scripts to automatically install and execute the tools in containerized virtual environments (specifically Dockerfiles to execute within Docker containers, along with execution scripts). With the automated evaluation system, most participating categories presented performance evaluation information from this common execution platform.

1 Introduction

This report summarizes the *repeatability evaluation* of the 2023 ARCH-COMP friendly competition of the ARCH workshop[1], and aims to provide an overview of the reproducibility of results for the participating verification tools. Verification researchers publish papers that emphasize computational contributions that depend on computational artifacts, but re-creation of these computational elements is often challenging because implementation details are unavoidably absent in papers. To address this, some authors post code and data to websites, but there is often limited formal incentive to do so, and typically there is no easy way to determine whether others can actually use or extend the results. Thus, over time, computational results may become non-reproducible, sometimes even by the researchers who originally produced them. Over about the past decade and increasingly in the past few years, the research community has instituted artifact evaluations and repeatability evaluations in various phases of review

[1]Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH), cps-vo.org/group/ARCH

processes to address these issues. Similarly, the goal of the repeatability evaluation for ARCH-COMP is to improve the reproducibility of computational results for the tools competing on the selected benchmarks evaluated in the competition and to provide further confidence in the results. This year, to give researchers immediate feedback on their submissions, we deploy an automatic evaluation system (https://arch.repeatability.cps.cit.tum.de/frontend) for the competition. This ensures that the submitted tools are repeatable at submission time and gives tool authors immediate feedback on the results of their submissions.

The remainder of this report presents a summary of the repeatability evaluation (RE) results. The results obtained in the competition have been evaluated by an independent repeatability evaluation conducted by the author of this report. To establish further confidence in the results, the artifacts, code, documentation, benchmarks, etc. with which the repeatability results have been obtained are publicly available on the ARCH website (https://cps-vo.org/group/ARCH) and a Git version control repository (https://gitlab.com/goranf/ARCH-COMP), and are also available at the aforementioned automatic evaluation system link.

2 Repeatability Evaluation Overview

The repeatability evaluation of the competition featured seven categories and eleven software tools, where several tools participated in multiple categories, but have been counted distinctly for their participation in each category. While the introduction of the automatic evaluation system has led to overall improvement in the RE, it also led to some confusion among participants, leading to some categories not participating in the RE in this iteration, which we will remedy with clearer and unified instructions in the future. The categories of problems that tools participated in the repeatability evaluation are:

- AFF: affine and piecewise affine dynamics (3 tools),
- AINNCS: artificial intelligence and neural network control systems (3 tools),
- FALS: falsification (no tools participating in the RE),
- HSTP: hybrid systems theorem proving (1 tool),
- NLN: nonlinear dynamics (4 tools),
- PCDB: piecewise constant dynamics and bounded model checking (no tools participating in the RE), and
- SM: stochastic models (no tools participating in the RE).

For the categories that have tools that participated in the RE, the tools evaluated, broken into their competition categories and alphabetically sorted, are:

- AFF
  - CORA [1],
  - JuliaReach [5, 16], and
  - Verse [14].
- AINNCS
  - CORA [12, 13],

  - JuliaReach [5], and
  - NNV [21, 20, 17, 18, 15].

- HSTP
  - HHL Prover [19].

- NLN
  - Ariadne [2, 3],
  - CORA [1],
  - JuliaReach [4], and
  - Verse [14].

All of the tools listed above were deemed repeatable based on the evaluation, as summarized next and detailed further in the next section that describes in more depth the process and results. Note that due to confusion in the processes this year that led to only some participants using the automatic evaluation system, those tools that participated in ARCH-COMP23 and are not included above are not deemed as being not reproducible. As the automatic evaluation system gives immediate feedback on the reproducibility of the tools in both, in case the repeatability fails and the benchmark results and times of their submission, tool authors can revise their submissions as desired. Thus, the effort tool authors put into the competition is also valued in the repeatability evaluation.

3 Repeatability Evaluation Details

The repeatability evaluation was conducted primarily before and partially following the presentations of the competition results at the ARCH'23 workshop. The basic mechanism followed in the repeatability evaluation was similar to that done in related conferences, and builds on the evaluation conducted in prior iterations of ARCH-COMP [6, 7, 8, 9, 10, 11], but augmented this year with the automatic evaluation system. The primary difference in the ARCH-COMP repeatability relative to those done at conferences is this evaluation was done primarily by the author of this report, and not an evaluation committee, as well as aided by the automatic evaluation system this year, that allowed authors to automatically produce computational results based on their artifacts. In many repeatability evaluations, three basic criteria are generally evaluated: coverage, instructions, and quality, each of which may be rated on a scale, typically of one through five, where one indicates a missing component or significantly below acceptability, and five indicates the criteria significantly exceeds expectations. Coverage evaluates the repeatability packages' ability to regenerate the images, tables, and log files presented in the competition. Instructions evaluates the packages' ability to describe to another researcher how to reproduce the results, including installation of the tool and how to execute it. Quality evaluates the packages' level of documentation and trustworthiness of results with respect to the quality of the software tool and the results it produces. This report does not describe the ratings of these review criteria for each tool evaluated, only the aggregate result of whether the submission was repeatable or not as deemed by the submitted package and corresponding artifacts.

The automatic evaluation system ensured the repeatability of the tools at submission time. Details can be found on the submission systems website (https://arch.repeatability.cps.cit.tum.de/frontend/getting-started) and are summarized next. Each submission consists of a zip file containing a Dockerfile, all required code and license files, and a Bash script to run the repeatability evaluation. Tool authors are asked to store their benchmark results in a standardized csv format to make them displayable on the website. After submission, the server automatically runs the evaluation within a Docker container, saves the results, and displays them back to the tool authors through the website. In case the repeatability fails, the error message is forwarded to the tool authors as well. Submissions are initially only visible to the tool authors. After correcting repeatability issues and double-checking their benchmark results, tool authors can publish their results to the public leaderboard. The public leaderboard can be seen by everybody and lists the benchmark results per category in searchable tables. Thus, every tool author can compare their results at submission time. This improves the transparency of the competition.

In prior iterations of the competition, the participants were sent instructions to provide their tool setup instructions and tool execution commands for the benchmarks evaluated in their respective categories, which were collected on a Git repository (https://gitlab.com/goranf/ARCH-COMP) by the competitors issuing commits and subsequent pull/merge requests that were reviewed and approved by the author of this report. We plan to automatically add the repeatability package of published tool results to the git repository in the next competition to make the processes clearer, however, one has to ensure to not leak private data (e.g. license files) to the public, but we retroactively updated the repository with final submissions in this iteration while preserving privacy. A description of how to run the file should also be included by the tool authors. The repeatability evaluation was performed on the competition benchmarks, the selection of which has been conducted within the forum of the ARCH website (cps-vo.org/group/ARCH), which is visible for registered users and registration is open for anyone to enable sharing of these models and benchmarks.

4 Conclusion and Outlook

This brief report summarizes the repeatability evaluation for the seventh competition for the formal verification of continuous and hybrid systems (ARCH-COMP'23), conducted as part of the ARCH'23 workshop at the 2023 Cyber-Physical Systems and Internet-of-Things Week (CPS-IoT Week). Detailed reports for the categories can be found in the proceedings (https://cps-vo.org/group/ARCH/proceedings) and on the ARCH website (http://cps-vo.org/group/ARCH). All documentation, benchmarks, and execution scripts for the repeatability evaluation are also archived on the ARCH website, and authors contributed their repeatability evaluations to the Git repository: https://gitlab.com/goranf/ARCH-COMP.

As in previous iterations of the competition and corresponding repeatability evaluation, several aspects to improve the process were identified. The most important aspect identified in this iteration of the RE is to improve instructions and clarity for the participants, to ensure further usage of the automatic evaluation system, along with archiving results. For this, we will update the repository in advance of the competition with the RE instructions to use the automatic evaluation system, and refer participants to it in advance. Additionally, we will augment the submission system, likely so that submissions to the automatic evaluation system

are performed by pulling from the main repository with the archival competition results, to ensure that repository contains the evaluated code for archival purposes. Finally, we will likely archive the results and logs to that repository as well, so that both the artifact execution files and outputs are available.

Beyond these suggested procedural improvements, there are still numerous aspects to address as discussed in prior RE reports (such as model input formats, output/log formats, semantics of more novel classes of models, etc.), but in part through this competition and evaluation, our efforts may serve to enhance the reproducibility of computational results and increase the scientific rigor in the community.

Acknowledgments

The author is grateful to Tobias Ladner, as well as others in Matthias Althoff's research group, for the automatic evaluation system and its description and overview for this iteration of the ARCH-COMP RE. The material presented in this report is based upon work supported by the National Science Foundation (NSF) under grant number 1910017, 2220401, and 2220426, the Air Force Office of Scientific Research (AFOSR) under contract numbers FA9550-22-1-0019 and FA9550-23-1-0135, and the Defense Advanced Research Projects Agency (DARPA) contract number FA8750-23-C-0518. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR, DARPA, nor NSF.

A Specifications of Used Machines

This year, we run all tools on the same hardware using tool-specific Docker images on the submission system described previously. The specification of the server used for the evaluation is as follows.

- Processor: AMD EPYC 7742 64-Core
- Memory: 995 GB
- Host Operating System: Ubuntu 22.04
- Docker: 20.10.21

References

[1] Matthias Althoff. An introduction to cora 2015. In Goran Frehse and Matthias Althoff, editors, *ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems*, volume 34 of *EPiC Series in Computing*, pages 120–151. EasyChair, 2015.

[2] Andrea Balluchi, Alberto Casagrande, Pieter Collins, Alberto Ferrari, Tiziano Villa, and Alberto L. Sangiovanni-Vincentelli. Ariadne: a framework for reachability analysis of hybrid automata. In *PROCEEDINGS OF THE INTERNATIONAL SYPOSIUM ON MATHEMATICAL THEORY OF NETWORKS AND SYSTEMS*, 2006.

[3] Luca Benvenuti, Davide Bresolin, Pieter Collins, Alberto Ferrari, Luca Geretti, and Tiziano Villa. Assume-guarantee verification of nonlinear hybrid systems with ariadne. *International Journal of Robust and Nonlinear Control*, 24(4):699–724, 2014.

[4] Sergiy Bogomolov, Marcelo Forets, Goran Frehse, Kostiantyn Potomkin, and Christian Schilling. Juliareach: A toolbox for set-based reachability. In *Proceedings of the 22Nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pages 39–44, New York, NY, USA, 2019. ACM.

[5] Sergiy Bogomolov, Marcelo Forets, Goran Frehse, Frédéric Viry, Andreas Podelski, and Christian Schilling. Reach set approximation through decomposition with low-dimensional sets and high-dimensional matrices. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*, HSCC '18, pages 41–50, New York, NY, USA, 2018. ACM.

[6] Taylor T. Johnson. ARCH-COMP17 repeatability evaluation report. In Goran Frehse and Matthias Althoff, editors, *ARCH17. 4th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 48 of *EPiC Series in Computing*, pages 175–180. EasyChair, 2017.

[7] Taylor T. Johnson. ARCH-COMP18 repeatability evaluation report. In Goran Frehse, editor, *ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 54 of *EPiC Series in Computing*, pages 128–134. EasyChair, 2018.

[8] Taylor T. Johnson. ARCH-COMP19 repeatability evaluation report. In Goran Frehse and Matthias Althoff, editors, *ARCH19. 6th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 61 of *EPiC Series in Computing*, pages 162–169. EasyChair, 2019.

[9] Taylor T Johnson. ARCH-COMP20 repeatability evaluation report. In Goran Frehse and Matthias Althoff, editors, *ARCH20. 7th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH20)*, volume 74 of *EPiC Series in Computing*, pages 175–183. EasyChair, 2020.

[10] Taylor T. Johnson. Arch-comp21 repeatability evaluation report. In Goran Frehse and Matthias Althoff, editors, *8th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH21)*, volume 80 of *EPiC Series in Computing*, pages 153–160. EasyChair, 2021.

[11] Taylor T Johnson. Arch-comp22 repeatability evaluation report. In Goran Frehse, Matthias Althoff, Erwin Schoitsch, and Jeremie Guiochet, editors, *Proceedings of 9th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH22)*, volume 90 of *EPiC Series in Computing*, pages 222–230. EasyChair, 2022.

[12] Niklas Kochdumper, Christian Schilling, Matthias Althoff, and Stanley Bak. Open-and closed-loop neural network verification using polynomial zonotopes. In *NASA Formal Methods Symposium*, pages 16–36. Springer, 2023.

[13] Tobias Ladner and Matthias Althoff. Automatic abstraction refinement in neural network verification using sensitivity analysis. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–13, 2023.

[14] Yangge Li, Haoqing Zhu, Katherine Braught, Keyi Shen, and Sayan Mitra. Verse: A python library for reasoning about multi-agent hybrid system scenarios. In Constantin Enea and Akash Lal, editors, *Computer Aided Verification*, pages 351–364, Cham, 2023. Springer Nature Switzerland.

[15] Diego Manzanas Lopez, Sung Woo Choi, Hoang-Dung Tran, and Taylor T. Johnson. Nnv 2.0: The neural network verification tool. In Constantin Enea and Akash Lal, editors, *Computer Aided Verification*, pages 397–412, Cham, 2023. Springer Nature Switzerland.

[16] Christian Schilling, Marcelo Forets, and Sebastian Guadalupe. Verification of neural-network control systems by integrating taylor models and zonotopes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):8169–8177, Jun. 2022.

[17] Hoang-Dung Tran, Feiyang Cai, Manzanas Lopez Diego, Patrick Musau, Taylor T. Johnson, and Xenofon Koutsoukos. Safety verification of cyber-physical systems with reinforcement learning control. *ACM Trans. Embed. Comput. Syst.*, 18(5s), October 2019.

[18] Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T. Johnson. Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In Shuvendu K. Lahiri and Chao Wang, editors, *Computer Aided Verification*, pages 3–17, Cham, 2020. Springer International Publishing.

[19] Shuling Wang, Naijun Zhan, and Liang Zou. An improved hhl prover: An interactive theorem prover for hybrid systems. In Michael Butler, Sylvain Conchon, and Fatiha Zaïdi, editors, *Formal Methods and Software Engineering*, pages 382–399, Cham, 2015. Springer International Publishing.

[20] Weiming Xiang and Taylor T Johnson. Reachability analysis and safety verification for neural network control systems. *arXiv preprint arXiv:1805.09944*, 2018.

[21] Weiming Xiang, Hoang-Dung Tran, and Taylor T. Johnson. Output reachable set estimation and verification for multi-layer neural networks. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, March 2018.