Statutes of the First International Competition for the Verification of Continuous and Hybrid Systems (ARCH-COMP 2017)

Goran Frehse, Matthias Althoff, Sergiy Bogomolov, Taylor T. Johnson, Stanley Bak

v1.0, 2016-11-14

Introduction

ARCH-COMP is a friendly competition of scientific software (in the spirit of, e.g., SV-COMP) in the context of algorithmic verification of continuous and hybrid systems. The first goal of the competition is to provide a forum for observation: which methods are particularly suitable to which types of problems; which types of problems already have good solutions; etc. The second goal of the competition is to establish a consensus for comparing different software implementations in the context of verification, as such comparisons are routinely demanded by reviewers of scientific publications. It is clear that this first instance of the competition is only the first step towards achieving these goals. For instance, it is difficult to formally define problem instances as well as formal evaluation criteria that fits more than one tool. All reported results are at best rough indicators of performance.

The organizers gratefully acknowledge the work of Dirk Beyer on SV-COMP (sv-comp.sosy-lab.org), which has inspired this competition in many ways.

General

The competition for 2017 adopts the following practices:

1. There is no ranking or scoring of tools. **The competition results consist of the performance measurements for each tool instance successfully applied to a problem instance.**

2. Experiments are carried out by the tool authors themselves, who then submit the performance measurements and a repeatability package to the evaluation chair, as detailed below.
3. The submitted results are approved by the jury.
4. The jury tries to resolve conflicts by way of consensus as far as possible. If unsuccessful, the general chairs reserve the right to resolve conflicts so that the competition can move forward.

The competition takes place in the following phases:

- during *registration* determines participating tools and their representatives,
- in the *classification phase,* jury members and tool instances are attributed to problem categories,
- in the *problem phase*, problem instances are defined by the jury,
- in the *evaluation phase*, experimental results are submitted and approved by the jury in two rounds; new problem instances may be created as a result of jury deliberations after the first round,
- the *final results* being presented at the ARCH17 workshop.

Working groups will interact via the ARCH forum at http://cps-vo.org/node/9442/og/forum. This is also where problem instances will be discussed and published during the evaluation phase.

Organizers

- Matthias Althoff, Goran Frehse (general chairs)
- Sergiy Bogomolov (publicity chair)
- Taylor T. Johnson, Stanley Bak (evaluation chairs)

Calendar

2016-10-03	Call for participation (deadline)
2016-11-21	Attribution of problem categories and working groups (start of problem phase)
2016-11-22	Submission of candidate problem instances
2017-01-15	Publication of problem instances and participating tools (start of first round of evaluation phase)
2017-02-15	Publication of the results of the first round of evaluation
2017-02-28	Publication of new problem instances (start of second round of evaluation phase)
2017-04-01	Publication of the results of the second round of evaluation (end of evaluation phase and start of final jury deliberation)
2017-04-21	Presentation of final results at ARCH17

Publication and Presentation

All participating tools will be presented by a short paper (~4 pages) in the proceedings of the ARCH17 workshop. The papers presenting the tools are due on the start date of the evaluation phase. Participating tools have the option, but not the obligation, to present their tool at ARCH17. The final results of the competition are presented by the competition chairs in an overview paper and in a presentation at ARCH17.

Participation

Participants are solicited by a public call for participation. The call for ARCH-COMP 2017 ended on October 3, 2016. Participating tools must be publicly available for download on the internet.

Participating Tools and Tool Instances

Since many tools incorporate more than one verification method, we differentiate between tools and tool instances. The actual participants of the competition are *tool instances*. Two instances of the same tool should refer to significantly different methods being used by the tool, not minor parameter tunings. **Each tool (not tool instance) has exactly one tool representative.**

Jury

The job of the jury is to come up with a consensus for problem categories, define the problem instances, and oversee the competition such that any inadvertently "unfair" treatment is avoided or rectified. Each tool representative is a member of the jury, and only tool representatives and the organizers are on the jury.
For each problem category, a subset of the jury called *working group* consists of the tool representatives of the tools participating in that category, as well as the organizers. A jury member can participate in more than one working group. Each working group is led by one of the organizers (designated by the general chairs), who is responsible for all organizational aspects (presiding discussions, ensuring the workflow, etc.). The definition of the problem instances and the ratification of experiments are carried out by the working group for each category.

Problem Instances

A problem instance consists of a model for the system and a property to be verified. Both should be specified in a form that is unambiguous. The goal is to maximize the number of tools that can

handle the instance, not formal nitpicking. Every problem instance is ratified by the working group of the jury, which gives it a unique identifier (see appendix).

Where possible, the **problem instance should originate in one of the benchmarks in the ARCH repository** (http://cps-vo.org/group/ARCH/benchmarks) and be specified in the SpaceEx file format, consisting of an XML file for the system and a CFG file for the specification (forbidden states).
(http://spaceex.imag.fr/documentation/user-documentation/spaceex-modeling-language-33).
A tool representative can submit two problem instances per tool instance. Ideally, the two problem instances should have opposing verification results (safe/unsafe, etc.).

The working group can suggest new problem instances for the second round of evaluation, e.g., to correct an inconsistent problem instance from the first round.

Experiments

Tool authors submit the results of an experiment by sending all of the following to the evaluation chairs:

1. the identifier of the problem instance,
2. the identifier of the tool instance,
3. an executable of the tool (including all binaries and libraries required to run the tool),
4. the exact model file read by the tool for this experiment,
5. all further data necessary to run the experiment (scripts, configuration files, etc.),
6. the performance measurements:
    a. runtime in seconds (real time),
    b. memory use in kilobytes,
    c. type of machine and processor,
    d. number of parallel computation threads.

All data except the executable will be made available to all participants via the competition website. The executable is treated as confidential and is distributed only to the evaluation chairs. The executable should be accompanied by instructions on how to run the tool, what OS is used, and how to interpret the result of the tool (whether the result is safe or unsafe).

All results are collected by the evaluation chairs, which then present them to the working group for ratification. The working group can suggest modifications to the experiment, in which case the tool representative submits a new result. This may occur, e.g., if the working group detects a deviation between the model file and the problem instance. Experiments can be re-submitted until the fixed deadline for submission.

Appendix

Problem Instance Identifier

Each problem instance has a unique *identifier* of the form XXXX-YYYYyy-ZZZzz, where

- XXXX is a four-letter code for the problem category,
- YYYY is a four-letter code for the system,
- yy is a two-letter code for the exact instance of the system, in particular for variations and evolutions of the system,
- ZZZ is a three-letter code for the specification to be verified,
- zz is a two-letter code for the exact instance of the specification, in particular for variations and evolutions of the specification.

The codes for systems and specifications are unique in the following sense: Two problem instances that share the same code XXXX-YYYYyy should refer to the exact same system. Two instances with the code XXXX-YYYYaa-ZZZzz and XXXX-YYYYbb-ZZZzz should refer to the exact same specification, but may differ in the system, e.g., through different parameter values, and produce different verification results.

Technical Points for Problem Instances

For translating to and from the SpaceEx format, consider the tools HySST (verivital.com/hyst) and SL2SX (www-verimag.imag.fr/~minopoli/sl2sx.html). Stanley Bak is available for assisting with questions about translations.

Technical Points for Experiments

Memory consumption can be measured using the tool memtime (http://www.update.uu.se/~johanb/memtime/).
If in doubt about the number of computational threads used, consider counting them using the tools ps (ps -eLf), top, or the activity monitor on OS X.