EPiC Series in Computing

Volume 54, 2018, Pages 14–22

ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems

ARCH-COMP18 Category Report: Bounded Model Checking of Hybrid Systems with Piecewise Constant Dynamics

Lei Bu[1], Rajarshi Ray[2], and Stefan Schupp[3]

[1] State Key Laboratory of Novel Software Techniques, Nanjing University, Nanjing, Jiangsu, P.R. China
bulei@nju.edu.cn

[2] National Institute of Technology Meghalaya, Shillong, India
rajarshi.ray@nitm.ac.in

[3] RWTH Aachen University, Theory of hybrid systems, Aachen, Germany
stefan.schupp@cs.rwth-aachen.de

Abstract

This report presents results of a friendly competition for formal verification of continuous and hybrid systems with linear continuous dynamics. The friendly competition took place as part of the workshop Applied Verification for Continuous and Hybrid Systems (ARCH) in 2018. In its second edition, three tools have been applied to solve three different benchmark problems in the category of*bounded model checking of hybrid systems with piecewise constant dynamics* (in alphabetical order): BACH, HyDRA, and XSpeed. This report is a snapshot of the current landscape of tools and the types of benchmarks they are particularly suited for. Due to the diversity of problems, we are not ranking tools and we also welcome more tools to join in this friendly competition in the future event.

1 Introduction

Disclaimer The presented report of the ARCH friendly competition for *bounded model checking of hybrid systems with piecewise constant dynamics* aims at providing a landscape of the current capabilities of verification tools. We would like to stress that each tool has unique strengths—not all of the specificities can be highlighted within a single report. To reach a consensus in what benchmarks are used, some tools may benefit more from the presented choice than others. Meanwhile, in order to invite more tools to join the competition, the flow condition is restricted to the format of $dx/dt = a$, instead of more general form of $dx/dt \in [a, b]$. The obtained results have been verified by an independent repeatability evaluation. To establish further trustworthiness of the results, the code with which the results have been obtained is publicly available at gitlab.com/goranf/ARCH-COMP.

This report summarizes results obtained in the 2018 friendly competition of the ARCH workshop[1] for bounded model checking of hybrid systems with piecewise constant dynamics. More specifically, the flow condition is restricted to the format of $dx/dt = a$, instead of more general form of $dx/dt \in [a, b]$. Tool developers run their tools summarized in Sec. 2 on different benchmark problems presented in Sec. 3 and report the results obtained from their own machines also in Sec. 3.

The results reported by each participant have not been checked by an independent authority and are obtained on the machines of the tool developers. Thus, one has to factor in the computational power of the used processors summarized in Sec. A as well as the efficiency of the programming language of the tools. It is not the goal of the friendly competition to rank the results, the goal is to present the landscape of existing solutions in a breadth that is not possible by scientific publications in classical venues. Those would require the presentation of novel techniques, while this report showcases the current state of the art.

The selection of the benchmarks 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 anybody. All tools presented in this report use some form of reachability analysis. This, however, is not a constraint set by the organizers of the friendly competition. We hope to encourage further tool developers to showcase their results in future editions.

2 Participating Tools

The tools participating in the category *Bounded Model Checking of Hybrid Systems with Piecewise Constant Dynamics* are introduced below in alphabetical order.

BACH BACH [3, 2] is a bounded reachability checker for Linear Hybrid Automata (LHA) model, Hybrid Systems with Piecewise Constant Dynamics (HPWC) in the term of ARCH competition. The tool provides GUI for LHA modeling and also bounded reachability checkers for both single automaton and automata network. Be different from classical bounded checkers of LHA, which encodes the "complete" bounded state space of the system into a huge SMT problem, BACH conducts the bounded checking in a "path-oriented" layered style. It finds potential paths which can reach the target location on the graph structure first, then encodes the feasibility of such path into a linear programming problem and solve it afterwards. In this way, as the number of paths in the discrete graph structure of an LHA under a given bound is finite, all candidate paths can be enumerated and checked one by one to tackle the bounded reachability analysis of LHA. Furthermore, the memory usage is well controlled as it only encodes and solves one path at a time. Meanwhile, BACH provides an efficient way to locate the infeasible path segment core when a path is reported as infeasible to guide the backtracking in the graph structure traversing to achieve good performance [13]. Such infeasible path segments can also be used to derive complete state arguments under certain conditions [14].

HyDRA The Hybrid systems Dynamic Reachability Analysis (HyDRA) tool implements flow-pipe construction based reachability analysis for linear hybrid automata. The tool is built on top of HyPro [8, 12], a C++ library for reachability analysis. HyPro provides different implementations of state set representations tailored for reachability analysis such as boxes, convex polyhedra, support functions, or zonotopes, all sharing a common interface. This interface allows to easily exchange the utilized state set representation in HyDRA. We use this to

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

extend state-of-the art reachability analysis by CEGAR-like parameter refinement loops, which (among other parameters) allow to vary the used set representation. Furthermore, HyDRA incorporates the capability to explore different branches of the search tree in parallel. Being in an early state of development, HyDRA already shows promising results on some benchmarks, although there is still room for improvements. An official first release is planned.

XSpeed The tool *XSpeed* implements algorithms for reachability analysis of continuous and hybrid systems with linear dynamics. The focus of the tool is to exploit the modern multicore architectures to enhance the performance through parallel computations. The algorithms in XSpeed are based on symbolic states represented using support functions. The tool can analyze hybrid automata models in the *SpaceEx* input format. It allows to compute the reachability in bounded depth as well as reachability till fixed point. XSpeed realizes two algorithms to enhance the performance of reachability analysis of purely continuous systems. The first is the parallel support function sampling algorithm and the second is the time-slicing algorithm [10, 11]. The performance of hybrid systems reachability analysis is enhanced using an adaptation of the G.J. Holzmann's parallel BFS algorithm in the SPIN model checker, called the AGJH algorithm [5]. In addition, a task parallel and an asynchronous variant of AGJH are also implemented in the tool. The details of the algorithms in XSpeed can be found in [6]. The tool is available at http://xspeed.nitmeghalaya.in/

3 Verification of Benchmarks

We have agreed on four benchmarks, each one of them having unique features. The *Motorcade* [9] instance used in the competition is a single model with small number of variables and locations. *Navigation* [4] instance is much larger than *Motorcade* in the aspect of locations. We also have Distributed Controller and TTEthernet from the HPWC category. Next, let us briefly discuss the specificities and results of each benchmark problem.

Types of Inputs As the HBMC category focuses on HPWC model this year, all the three benchmarks are HPWC models with dynamic laws in the form of $dx/dt = a$, instead of more general form of $dx/dt \in [a, b]$. Therefore, the models used in the competition are a little different from their original versions. This restriction may causes different behavior in some benchmarks as well. For example, in NAV model, the point is supposed to move in different directions in the original model with $dx/dt \in [-a, a]$. However, in our version, as it is fixed to $dx/dt = a$, it can only goes in one direction.

3.1 Motorcade, a.k.a. Adaptive Cruise Controller

3.1.1 Model

The first automaton is a central arbiter for a automated motorcade in a highway introduced in [9]. The system works as follows: when two vehicles come within a distance 4 of each other, a collision may happen. Then the arbiter asks the approaching car to slow down and the leading car to speed up. When the distance between the two vehicles involved in the possible collision exceeds 4, the arbiter model goes back to the dynamics of the cruise mode.

MOTORnn is the model of the system with nn cars. This model is considered safe with respect to specification UBSnn (no collisions). The size of this model can be easily expanded by introducing more cars into the system, which will increase new locations and variables in the model. However, in order to make the benchmark suitable for most of the competitors, we select systems with 5, 10, and 15 vehicles for competition. The model with 5 vehicles is shown below in Fig.1.

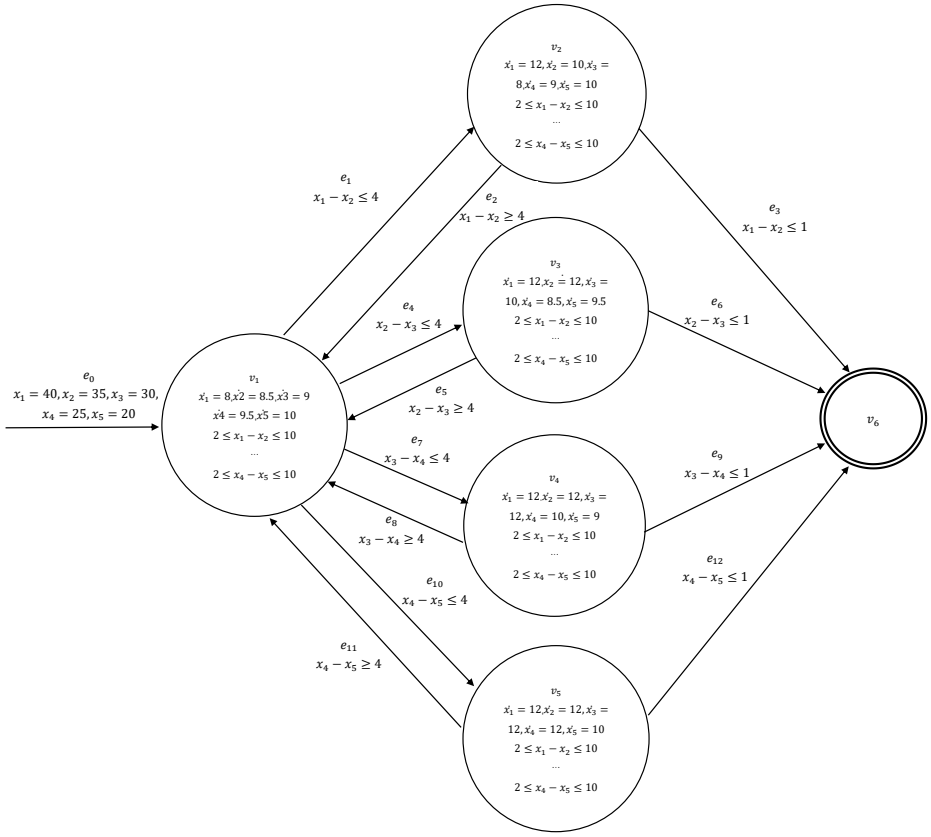

Figure 1: Motorcade-5 vehicles

3.1.2 Specification

The safety property, UBDnn, of this system is that no two vehicles collide with each other, in another word, whether location $Error\, v_{n+1}$ is reachable in bound 20.

3.1.3 Results

First of all, as different tools may have different settings, we have a specific column “remark” listed in all the forms. If the tool is executed directly according to the specification, it will be marked as “-”. Otherwise, the setting will be marked there.

Table 1: Computation Times on the Motorcade Benchmarks with Default Bound 20.

instance	MOTOR05 UBD05	MOTOR10 UBD10	MOTOR15 UBD15			
safety	safe	safe	safe			
# vars.	5	10	15			
# locs.	6	11	16			
tool	computation time in [s]			remark	lang.	machine
BACH	0.05	0.12	0.2		C++	M_{BACH}
HyDRA	6.44	-	-	2 threads, Time=40000, $\delta = 10$	C++	M_{HyDRA}
XSpeed	23.12	-	-	Bound=10, DIR=Box, $\delta = 10$	C++	M_{XSpeed}

Computation Times The computation times of various tools for the Motorcade benchmark are listed in Tab. 1. The performance for *XSpeed* is given for an exploration till bound 10, using a time step of $\delta = 10$ and box template directions. *XSpeed* does not reach the target error state in bound 10.

3.2 Navigation (NAV)

3.2.1 Model

The navigation example is also a single automaton. It models the motion of a point robot in a n-dimensional cube. The cube is partitioned into m^n rectangular regions and each such region is associated with a vector field described by the flow equations. We use a generalization method introduced in [4] to generate such a navigation mode, NAV_m_n. Similar with the motorcade model, in order to generate a not too complex model, we set both m and n as 2, 3, and 4 respectively. As the model is too large to put in the paper, we will omit the graphical presentation here.

3.2.2 Specification

The specification, NBD_m_n, is to check whether there is a behavior of the system which can reach the specific state in the farthest corner. In the benchmark model, Whether $l_{\underbrace{(n-1)\ldots(n-1)}_{n-1}}$ is reachable in bound 20.

3.2.3 Result

Computation Times The computation times of various tools for the NAV benchmark are listed in Tab. 2. The results in XSpeed for NAV_2_2, NAV_3_3 are generated with a time-step δ of $1e-4$ with *BOX* template directions. The reported time in XSpeed for the instances NAV_2_2, NAV_3_3 are till the completion of bound 20. In the case of NAV_4_4 instance, the XSpeed reported time is for a bound of 16, with $\delta = 0.01$. The reported time in HyDRA for the navigation instances have been computed using boxes as the state set representation, $\delta = 1e-4$ and a jump bound of 20.

Table 2: Computation Times on the NAV Benchmark with Default Bound 20.

instance	NAV_2_2 NBD_2_2	NAV_3_3 NBD_3_3	NAV_4_4 NBD_4_4			
safety	safe	safe	safe			
# vars.	2	3	4			
# locs.	4	27	256			
tool	computation time in [s]			remark	lang.	machine
BACH	0.03	0.13	7.34		C++	M_{BACH}
HyDRA	0.22	0.40	1.2		C++	M_{HyDRA}
XSpeed	0.33	0.6	91.92	Bound=16 for NAV_4_4	C++	M_{XSpeed}

3.3 Distributed controller

Model The benchmark is an extension of the benchmarks presented in [7], to which multiple sensors with multiple priorities have been added. It models the distributed controller for a robot that reads and processes data from different sensors. A scheduler component determines what sensor data must be read according to the priority of the sensor. The controller has 1 continuous and n discrete variables, the scheduler has n continuous and n discrete variables, and each sensor has 1 continuous variable. The controller has 4 locations, the scheduler has $1 + n$, and each sensor has 4 locations. The product automaton has $4 \times (1 + n) \times 4$ locations, $n + 2$ continuous variables and $2n$ discrete variables. Note that some tools, such as PHAVer, do not support discrete variables and may model the discrete variables as continuous variables.

DISCnn The model with nn sensors. This model is considered safe with respect to specification UBSnn.

Specification The system is considered safe if at no point in time all sensors send data simultaneously.

UBSnn It is never the case that all nn sensors are in location `send`.

Results The computation times of various tools are listed in Tab. 3. The presented running times for HyDRA account for a run with a maximal jump depth of 15 jumps.

3.4 TTEthernet

Model The TTEthernet protocol is a protocol for the remote synchronization of possibly drifted clocks distributed over multiple components, taken from [1]. The system consists of two compression masters (CM) and k synchronization masters (SM). Each CM has two clocks cm_i, each SM has one clock sm_i. Both CM and SM are modeled by a hybrid automaton with 4 locations each. The product automaton has $4 + k$ variables and 4^{k+2} locations.

TTESnn protocol with nn SM. This model is considered safe with respect to specification UBDnn. The global time horizon is limited to 3000 ms.

Table 3: Computation Times of the Distributed Controller with Default Bound 20.

instance	DISC04 UBS04	DISC05 UBS05	DISC06 UBS06			
safety	safe	safe	safe			
#vars.	14	17	20			
#locs.	80	96	112			
tool	computation time in [s]			remark	lang.	machine
BACH	0.29	0.4	0.7		C++	M_{BACH}
HyDRA	234	–	–	Bound=15	C++	M_{HyDRA}
XSpeed	-	-	-		C++	M_{XSpeed}

Table 4: Computation Times of the TTEthernet Benchmark with Default Bound 20.

instance	TTES05 UBD05	TTES07 UBD07	TTES09 UBD09			
safety	safe	safe	safe			
# vars.	9	11	13			
# locs.	16384	262144	4194304			
tool	computation time in [s]			remark	lang.	machine
BACH	0.16	0.22	0.26		C++	M_{BACH}
HyDRA	412	–	–	Bound=10	C++	M_{HyDRA}
XSpeed	-	-	-		C++	M_{XSpeed}

Specification The difference between the clocks of the SM should not exceed a threshold of $2max\_drift$.

UBDnn For all i, j, $sm_i - sm_j \leq 2max\_drift$, where $max\_drift = 0.001$ ms.

Results The computation times of various tools are listed in Tab. 4. The presented running times for HyDRA account for a run with a maximal jump depth of 10 jumps.

4 Conclusion and Outlook

This report presents the results on the second friendly competition for the *bounded model checking of hybrid systems with piecewise constant dynamics* as part of the ARCH'18 workshop. The reports of other categories can be found in the proceedings and on the ARCH website: cps-vo.org/group/ARCH.

Compared with last year's competition, this year we increase the number of benchmarks and also enhance the difficulty of existing cases as well. Similar with last year's result, a most interesting observation of the results is that different techniques/tools have different specialities. For example, BACH is a path-oriented BMC checker, while HyDRA and XSpeed are fixed-point computation based. In the experiments, on case like Motorcade, which structure is not very complex but may have many possible execution trajectories. BACH finishes the checking quickly, while fixed-point computation based method may have difficulty in the computation.

On the other hand, the NAV model shows a different story. The structure of the model is large. There are many potential paths on the structure of the model. However, due to the restriction of the flow condition, only one way is possible according to the state computation. Therefore, fixed-point computation based method runs also very quick on this case.

We would like to introduce more complex models with high dimension and/or large initial set in the future event to see what kind of system can be analyzed by existing tools. We would also like to encourage other tool developers to consider participation in the next year.

5 Acknowledgments

The authors gratefully acknowledge financial support of the Joint NSFC-ISF Research Program, jointly funded by the National Natural Science Foundation of China and the Israel Science Foundation (No.61561146394), and the National Natural Science Foundation of China (No.61572249), and the National Institute of Technology Meghalaya, India and DST-SERB, Government of India under project grant No. YSS/2014/000623.

A Specification of Used Machines

A.1 $\mathbf{M}_{BACH}$

- Processor: Intel(R) Core(TM)2 Quad CPU Q9500 @ 2.83GHz x 4
- Memory: 4 GB
- Average CPU Mark on www.cpubenchmark.net: 3636 (full), 1203 (single thread)

A.2 $\mathbf{M}_{Lyse}$

A.3 $\mathbf{M}_{HyDRA}$

- Processor: Intel Core i7-4790K CPU @ 4.00GHz x 8
- Memory: 15.9 GB
- Average CPU Mark on www.cpubenchmark.net: 11185

A.4 $\mathbf{M}_{XSpeed}$

- Processor: Intel Core i7-4770 CPU @ 3.4GHz x 4
- Memory: 8 GB
- Average CPU Mark on www.cpubenchmark.net: 9806

References

[1] Sergiy Bogomolov, Christian Herrera, and Wilfried Steiner. Benchmark for verification of fault-tolerant clock synchronization algorithms. In *ARCH (2016)*.

[2] Lei Bu, You Li, Linzhang Wang, Xin Chen, and Xuandong Li. BACH 2 : Bounded reachability checker for compositional linear hybrid systems. In *Design, Automation and Test in Europe, DATE 2010, Dresden, Germany, March 8-12, 2010*, pages 1512–1517, 2010.

[3] Lei Bu, You Li, Linzhang Wang, and Xuandong Li. BACH : Bounded reachability checker for linear hybrid automata. In *Formal Methods in Computer-Aided Design, FMCAD 2008, Portland, Oregon, USA, 17-20 November 2008*, pages 1–4, 2008.

[4] Roberto Giacobazzi, Josh Berdine, and Isabella Mastroeni, editors. *Verification, Model Checking, and Abstract Interpretation, 14th International Conference, VMCAI 2013, Rome, Italy, January 20-22, 2013. Proceedings*, volume 7737 of *Lecture Notes in Computer Science*. Springer, 2013.

[5] Amit Gurung, Arup Deka, Ezio Bartocci, Sergiy Bogomolov, Radu Grosu, and Rajarshi Ray. Parallel reachability analysis for hybrid systems. In *2016 ACM/IEEE International Conference on Formal Methods and Models for System Design, MEMOCODE 2016, Kanpur, India, November 18-20, 2016*, pages 12–22. IEEE, 2016.

[6] Amit Gurung, Rajarshi Ray, Ezio Bartocci, Sergiy Bogomolov, and Radu Grosu. Parallel reachability analysis of hybrid systems in xspeed. *International Journal on Software Tools for Technology Transfer*, Feb 2018.

[7] Thomas A. Henzinger and Pei-Hsin Ho. HYTECH: the cornell hybrid technology tool. In Panos J. Antsaklis, Wolf Kohn, Anil Nerode, and Shankar Sastry, editors, *Hybrid Systems II*, volume 999 of *Lecture Notes in Computer Science*, pages 265–293. Springer, 1994.

[8] Hypro project website. Available at `http://ths.rwth-aachen.de/research/projects/hypro/`.

[9] Sumit Kumar Jha, Bruce H. Krogh, James E. Weimer, and Edmund M. Clarke. Reachability for linear hybrid automata using iterative relaxation abstraction. In *Hybrid Systems: Computation and Control, 10th International Workshop, HSCC 2007, Pisa, Italy, April 3-5, 2007, Proceedings*, pages 287–300, 2007.

[10] Rajarshi Ray and Amit Gurung. Poster: Parallel state space exploration of linear systems with inputs using xspeed. In *Proc. of HSCC'15*, pages 285–286. ACM, 2015.

[11] Rajarshi Ray, Amit Gurung, Binayak Das, Ezio Bartocci, Sergiy Bogomolov, and Radu Grosu. XSpeed: Accelerating reachability analysis on multi-core processors. In *Proc. of HVC 2015*, volume 9434 of *LNCS*, pages 3–18, 2015.

[12] Stefan Schupp, Erika Abraham, Ibtissem Ben Makhlouf, and Stefan Kowalewski. HyPro: A C++ library for state set representations for hybrid systems reachability analysis. In *Proc. NFM'17*, volume 10227 of *LNCS*, pages 288–294. Springer, 2017.

[13] Dingbao Xie, Lei Bu, Jianhua Zhao, and Xuandong Li. SAT-LP-IIS joint-directed path-oriented bounded reachability analysis of linear hybrid automata. *Formal Methods in System Design*, 45(1):42–62, 2014.

[14] Dingbao Xie, Wen Xiong, Lei Bu, and Xuandong Li. Deriving unbounded reachability proof of linear hybrid automata during bounded checking procedure. *IEEE Trans. Computers*, 66(3):416–430, 2017.