EPiC Series in Computing

Volume 103, 2024, Pages 1–14

Proceedings of the 11th Int. Workshop on Applied Verification for Continuous and Hybrid Systems

ARCH-COMP24 Category Report: Hybrid Systems with Piecewise Constant Dynamics and Bounded Model Checking

Lei Bu[1], Atanu Kundu[2], Rajarshi Ray[2], and Yuhui Shi[1]

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

[2] Indian Association for the Cultivation of Science, Kolkata, India
{mcsak2346, rajarshi.ray}@iacs.res.in

Abstract

This report presents the results of a friendly competition for formal verification of continuous and hybrid systems with piecewise constant dynamics. The friendly competition took place as part of the workshop Applied Verification for Continuous and Hybrid Systems (ARCH) in 2024. In this edition, three tools have been applied to solve nine different benchmark problems in the category for piecewise constant dynamics: BACH, SAT-Reach, and XSpeed. Compared to the last competition in 2022, three new benchmarks have been added (Nuclear Reactor System, FDDI, Ring-Shape Fischer's Protocol). The result 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, yet the presented results probably provide the most complete assessment of tools for the safety verification of continuous and hybrid systems with piecewise constant dynamics up to this date.

1 Introduction

Disclaimer The presented report of the ARCH friendly competition for *continuous and hybrid systems with piecewise constant dynamics* and *bounded model checking* 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 compromises had to be made so that some tools may benefit more from the presented choice than others. 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 2024 friendly competition of the ARCH workshop[1] in PCDB category for two types of problems: for verifying hybrid systems with piecewise constant bounds on the dynamics (HPCD) and for bounded model checking (BMC) of HPCD systems.

The PCDB category concerns hybrid systems where in each location (mode, piece of the hybrid state space), the dynamics are given by a differential inclusion of the form

$$\dot{x}(t) \in \mathcal{U},$$

where $\mathcal{U}$ is a convex subset of $\mathbb{R}^n$. Specifically, the BMC task concerns the bounded model checking of HPCD systems where the bound is described as the depth of the discrete jump of the system.

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 *PCDB: Continuous and Hybrid Systems with Piecewise Constant Dynamics and Bounded Model Checking* are introduced below in alphabetical order.

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

BACH BACH [7, 6, 20] 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 [21]. Such infeasible path segments can also be used to derive complete state arguments under certain conditions [22].

SAT-Reach SAT-Reach [15], is a bounded model checker for hybrid systems with piecewise affine dynamics, modeled as an affine hybrid automaton (AHA). Piecewise affine dynamics subsumes piecewise constant dynamics and the tool is therefore capable of analyzing LHA models as well. The tool can analyze both single automaton and automata networks. SAT-Reach can parse models designed using the GUI of SpaceEx and provides a command-line-interface to the user. The input to the tool is an hybrid automaton model file represented in an XML format accepted by many hybrid systems verification tools, and a configuration file to describe the initial states, forbidden states, and parameter settings of the tool. The novelty in the tool is a path-guided reachability analysis in which potential paths are obtained from the discrete structure of the hybrid automaton by sat-solving, and a set-based reachability analysis [10] is subsequently carried out along the candidate paths. SAT-Reach incorporates an effective path pruning by detecting infeasible paths based on reachability analysis. A new algorithm for efficiently tackling compositional affine hybrid automata has been recently implemented which does not require the expensive product automata construction. The tool is open-source and available online at SAT-Reach.

XSpeed XSpeed implements algorithms for reachability analysis of continuous and hybrid systems with affine 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 and provides a command-line-interface to the users. It supports reachability computations 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 [17]. 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, which is called the AGJH algorithm [12]. In addition, a task parallel and an asynchronous variant of AGJH are also implemented in the tool. The tool is available at `https://gitlab.com/raj.ray84/XSpeed-plan`.

Table 1: Computation Times of the Adaptive Cruise Controller.

instance	ACCS05	ACCU05	ACCS06	ACCU06	ACCS07	ACCU07	ACCS08	ACCU08
safety	safe	unsafe	safe	unsafe	safe	unsafe	safe	unsafe
# vars	5	5	6	6	7	7	8	8
# locs	32	32	64	64	128	128	256	256
tool	computation time in [s]							
bounded spec. [2]	BD05	BD05	BD06	BD06	BD07	BD07	BD08	BD08
BACH	3.44	0.05	4.53	0.11	6.83	0.10	7.97	0.14
XSpeed	387.7($B:2$)	-	182.32($B:1$)	-	-	-	-	-
SAT-Reach	377.01($B:3$)	-	82.64($B:3$)	-	-	-	-	-

3 Verification of Benchmarks

3.1 Adaptive Cruise Controller

Model The adaptive cruise controller is a distributed system for assuring that safety distances in a platoon of cars are satisfied [4]. It is inspired by a related benchmark in [14]. For n cars, the number of discrete states is 2^n and the number of continuous variables is n. Each variable x_i encodes the relative position of the i-th car, for $i = 0, \ldots, n-1$. The i-th car is considered to be in front of the $i+1$-th car. The relative velocity of each car has a drift $|\dot{x}_i - \dot{x}_{i+1}| \leq 1$ when cruising and $|\dot{x}_i - \dot{x}_{i+1} - \varepsilon| \leq 1$ when recovering, where ε is the slow-down parameter. The cars can stay in cruise mode as long as the distance to the preceding vehicle is greater 1. They can go into recovery mode when the distance is smaller than 2.

ACCS*nn* The model with *nn* cars, $\varepsilon = 2$. This model is considered safe with respect to specification BD*nn* (no collisions).

ACCU*nn* The model with *nn* cars, $\varepsilon = 0.9$. This model is considered unsafe with respect to specification BD*nn* (collisions are possible).

Specification The distance between adjacent cars should be positive:

$$x_{\texttt{ldr}} - x > 0,$$

where x and $x_{\texttt{ldr}}$ are the positions of the car and the car in front, respectively.

For bounded state space, we have BD*nn*.

BD*nn* For $i = 0, \ldots, n-1$: $x_i - x_{i+1} > 0$ within default discrete search depth 30.

Results The computation times of various tools are listed in Tab. 1.

Note on XSpeed and SAT-Reach The flow expressions of the form $c_1 \leq d' \leq c_2$ are modified to $d' = c$, where c is a new added variable in the system and the initial specification includes the constraints $c_1 \leq c \leq c_2$.

[2] The search depth p is indicated as $(B:p)$, and counted as the number of discrete transitions taken.

Table 2: Computation Times of the Distributed Controller.

instance	DISC02	DISC03	DISC04	DISC05
safety	safe	safe	safe	safe
# vars	9	13	17	21
# locs	192	1024	5120	24976
tool	computation time in [s]			
bounded spec.	BD02	BD03	BD04	BD05
BACH	0.14	0.24	0.29	0.48
XSpeed	15.81	347.85($B:13$)	872.09($B:12$)	556.35($B:11$)
SAT-Reach	2.98	68.93	978.96($B:25$)	-

3.2 Distributed controller

Model The benchmark is an extension of the benchmarks presented in [13], 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 $4\times(1+n)\times 4^n$ locations, $2n+1$ continuous variables and $2n$ discrete variables.

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

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

For bounded state space, we have BDnn.

BDnn All nn sensors are not in location `send` within default discrete search depth 30.

Results The computation times of various tools are listed in Tab. 2.

Note on XSpeed and SAT-Reach The guard conditions of the form $x+y<c$ is converted to $x+y\leq c$.

3.3 Fischer's Protocol

Model Fischer's protocol is a time based protocol of mutual exclusion between processes, originally from [16]. The flow constraints are given by $\frac{1}{2}\leq \dot{x}_1\leq \frac{3}{2},\ldots,\frac{1}{2}\leq \dot{x}_m\leq \frac{3}{2}$, where x_i is the clock of the i-th process. The product automaton has $(n+1)\times 4^n$ locations and n variables.

FISCSnn protocol with nn processes, considered safe with respect to specification BDnn.

FISCUnn protocol with nn processes, considered unsafe with respect to specification BDnn.

Table 3: Computation Times of the Fischer Benchmark.

instance	FISCS04	FISCU04	FISCS05	FISCU05	FISCS06	FISCU06
safety	safe	unsafe	safe	unsafe	safe	unsafe
# vars	4	4	5	5	6	6
# locs	1280	1280	6144	6144	28672	28672
tool	computation time in [s]					
bounded spec.	BD04	BD04	BD05	BD05	BD06	BD06
BACH	11.93($B:20$)	0.20	49.70($B:20$)	0.46	131.11($B:20$)	0.97

Specification The protocol is correct if no two processes are ever in the critical section at the same time.

For bounded state space, we have BD*nn*.

BD*nn* There are no two processes such that both are in location `cs` (critical section) at the same time within default discrete search depth 30.

Results The computation times of various tools are listed in Tab. 3.

3.4 Navigation (NAV)

3.4.1 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 [11] 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 m as 3 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.

3.4.2 Specification

The specification 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 $l_{\underbrace{(m-1)\dots(m-1)}_{n}}$ is reachable.

For bounded state space, we have BD_*m*_*n*.

BD_*m*_*n* The system is not in location $l_{\underbrace{(m-1)\dots(m-1)}_{n}}$ within default discrete search depth 30.

3.4.3 Result

Computation Times The computation times of various tools for the NAV benchmark are listed in Tab. 4.

Table 4: Computation Times on the NAV Benchmark

instance	NAV_3_2	NAV_3_3	NAV_3_4
safety	safe	safe	safe
# vars	3	4	5
# locs	9	27	81
tool	computation time in [s]		
bounded spec.	BD_3_2	BD_3_3	BD_3_4
BACH	0.10	10.00	37.63
XSpeed	446.74($B:15$)	116.95($B:15$)	690.32($B:15$)
SAT-Reach	618.33($B:18$)	314.77($B:19$)	662.25($B:15$)

Note on XSpeed and SAT-Reach Since XSpeed and SAT-Reach expects affine dynamics, the flow expressions of the form $x' \in [x_l, x_h]$ is converted to $x' = xc$, where xc is an introduced variable together with added constraint $x_l \leq xc \leq x_h$ in the initial condition.

3.5 TTEthernet

Model The TTEthernet protocol is a protocol for the remote synchronization of possibly drifted clocks distributed over multiple components, taken from [5]. 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.

TTESn protocol with n SM. This model is considered safe with respect to specification BDn. The global time horizon is limited to 3000 ms.

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

For bounded state space, we have BDnn.

BDn For all i, j, $sm_i - sm_j \leq 2max\_drift$ within default discrete search depth 30, where $max\_drift = 0.001$ ms.

Results The computation times of various tools are listed in Tab. 5.

3.6 Dutch Railway Network

We consider a finite-horizon safety problem over max-plus linear (MPL) systems. An MPL system is described by recurrence equation

$$\mathbf{x}(k+1) = A \otimes \mathbf{x}(k), k = 0, 1, \ldots, \tag{1}$$

Table 5: Computation Times of the TTEthernet Benchmark

instance	TTES05	TTES07	TTES09
safety	safe	safe	safe
# vars	9	11	13
# locs	16384	262144	4194304
tool	computation time in [s]		
bounded spec.	BD05	BD07	BD09
BACH	0.12	0.13	0.18

where $\mathbf{x}(k) = [x_1(k) \dots x_n(k)] \in \mathbb{R}^n$ is the state variables representing the time stamps of the discrete events at time horizon k and A is $n \times n$ max-plus algebraic matrix representing model under consideration. It should be noted that the matrix operation on (1) is defined over max-plus algebra: see [3] for more detailed descriptions about max-plus algebra and its operations, and to [1, 2] for more details on formal verification of MPL systems.

Given an MPL system (1), a time horizon N, a set of initial conditions X_0 and an unsafe set S, a finite-horizon safety problem is an instance problem to check whether the system can reach the unsafe set within the given time horizon.

Model In [18, Appendix B], the high-scale Dutch railway networks are modeled as a max-plus-linear(MPL) system. For details on MPL system, please refer to [2] That model has 214 state variables $x_1(k), \dots, x_{214}(k)$ representing the k-th departure from the selected stations. For ARCH-COMP this year, we use a subset of that model by considering only the first 21 state variables $x_1(k), \dots, x_{21}(k)$. The model instance is defined formally as follows:

DRNW03 initial condition $X_0 = \{\mathbf{x} \in \mathbb{R}^{21} : 0 \leq x_1 \leq x_2 \leq \dots \leq x_{21} \leq 1\}$.

The model is easily embedded in a hybrid automaton with a single location, where the time derivative of all variables is zero, and a self-loop transition that models the discrete dynamics for each region.

Specification We have four specifications of interest for bounded state space:

BD01 $\exists k = 0, \dots, 30$ such that $x_1(k) > x_2(k) > \dots > x_{21}(k)$ (not satisfied)

BD02 $\exists k = 0, \dots, 30$ such that $x_1(k) - x_2(k) > 20$ or $x_1(k) - x_2(k) < -20$ (satisfied)

BD03 $\exists k = 0, \dots, 30$ such that $x_9(k) - x_{13}(k) > 40$ or $x_9(k) - x_{13}(k) < -40$ (satisfied)

BD04 $\exists k = 0, \dots, 30$ such that $x_{17}(k) - x_{21}(k) > 60$ or $x_{17}(k) - x_{21}(k) < -60$ (not satisfied)

In the sense of a safety specification, the above specifications specify unsafe states. If the unsafe sets are reachable, the corresponding specification BD01, . . . , BD04 is satisfied.

Results The computation times of various tools are listed in Tab. 6.

Table 6: Computation Times of the Dutch Railway Network Benchmark

instance	DRNW03	DRNW03	DRNW03	DRNW03
safety	safe	unsafe	unsafe	safe
# vars	21	21	21	21
# locs	1	1	1	1
tool	computation time in [s]			
bounded spec.	BD01	BD02	BD03	BD04
BACH	0.08	0.14	0.29	70.96

Note on the model Given an MPL system (1), it is possible that there exist $k_0, c \in \mathbb{N}$, $\lambda \in \mathbb{R}$ such that for all $\mathbf{x}(0) \in \mathbb{R}^n$ the trajectory of (1) starting from $\mathbf{x}(0)$ satisfies

$$x_i(k+c) = (\lambda \times c) + x_i(k),\ i = 1, \ldots, n,\ k \geq k_0.$$

The smallest such k_0 and c are called the transient and the cyclicity of A, respectively. Furthermore, the scalar λ corresponds to the max-plus eigenvalue of the state matrix A. We refer the interested readers to [3, Section 3.7] about this periodic behavior and how to compute transient and cyclicity.

For the underlying DRNW03 model, the corresponding transient and cyclicity is $k_0 = 26$ and $c = 5$, respectively. As a result, $x_i(k+5) - x_j(k+5) = x_i(k) - x_j(k)$ for all $i, j \in \{1, \ldots, 21\}$ and $k \geq 26$. Since, the unsafe sets BD01, ..., BD04 can be expressed as the conjunction or disjunction of inequalities $x_i(k) - x_j(k) > c$ for $c \in \mathbb{R}$, it is suffice to check the reachability problems BD01, ..., BD04 up to time horizon $N = 26 + 5 - 1 = 30$.

3.7 Nuclear Reactor System

Model The nuclear reactor system is a benchmark from [19]. The system contains a controller that controls a nuclear reactor with rods that absorb neutrons one by one, and k rods that are subject to the control of the controller process. Each rod that has just been moved out must stay out of the water and cool for several time units. Each process of the system has a clock with rate $0.9 \leq \dot{x}_i \leq 1.1$ for all i, where x_i is the clock of the i-th process. Each rod has a location "out" with invariant $x \leq \varepsilon$. The product automaton of n rods has $(n+1) \cdot 3^n$ locations and $n+1$ variables.

NRSS*nn* The model with *nn* rods, $\varepsilon = 10$. This model is considered safe with respect to specification BD*nn*.

NRSU*nn* The model with *nn* rods, $\varepsilon = 10000$. This model is considered unsafe with respect to specification BD*nn*.

Specification The system is considered safe if there is always only one rod absorbing neutrons in the heavy water.

For bounded state space, we have BD*nn*.

BD*nn* It is never the case that all rods are in location `recover` and the controller is in location `rod-0`, within default discrete search depth 30.

Table 7: Computation Times of the Nuclear Reactor System Benchmark

instance	NRSS05	NRSU05	NRSS06	NRSU06	NRSS07	NRSU07	NRSS08	NRSU08	NRSS09	NRSU09	NRSS10	NRSU10
safety	safe	unsafe	safe	unsafe	safe	unsafe	safe	unsafe	safe	unsafe	safe	unsafe
# vars	6	6	7	7	8	8	9	9	10	10	11	11
# locs	1458	1458	5103	5103	17469	17469	59049	59049	196830	196830	649539	649539
tool	computation time in [s]											
bounded spec.	BDS05	BDU05	BDS06	BDU06	BDS07	BDU07	BDS08	BDU08	BDS09	BDU09	BDS10	BDU10
BACH	0.18	0.04	0.27	0.05	0.42	0.05	0.42	0.06	0.49	0.09	0.68	0.09
SAT-Reach	17.01	10.68	18.33	14.97	19.37	19.23	21.25	26.57	24.25	31.57	37.30	42.98
XSpeed	1.50	-	2.18	-	2.66	-	3.23	-	-	-	-	-

Results The computation times of various tools are listed in Tab. 7.

Note on SAT-Reach The flow expressions given as $c_1 \leq x' \leq c2$ are rephrased by introducing a new variable c into the system. This new approach transforms the flow expressions into $x' = c$, with the stipulation that c adheres to the constraints $c_1 \leq c \leq c_2$ as specified initially.

3.8 FDDI

Model The FDDI protocol is a set of standards for data transmission on fiber optic lines in a LAN. The system is a ring topology model based on the system in [8, 9]. Each station in the system waits for the signal of previous one to transmit data. The product automaton of n stations has 6^n locations and $3n$ variables.

FDDISnn The model with nn stations, where each idle station sends synchronous messages before a constant time. This model is considered safe with respect to specification BDnn.

FDDIUnn The model with nn stations, where each idle station does not send any synchronous messages. This model is considered unsafe with respect to specification BDnn.

Specification The system is considered safe if the time elapsed within two consecutive receptions of the token by any station is bounded by a constant time limit.

For bounded state space, we have BDnn.

BDnn All rods are not in location `s4` within default discrete search depth 30.

Results The computation times of various tools are listed in Tab. 8.

Note on SAT-Reach and XSpeed We utilized the horizon 600 and time step 100 for the analyzed instances in both SAT-Reach and XSpeed.

3.9 Ring-Shape Fischer's Protocol

Model Ring-shape Fischer's protocol is a variant of the original Fischer's protocol (star-shape). It contains a ring of processes where each process shares a variable with its left and right neighbor; the variables are used to access critical sections in mutual exclusion with the neighbors. The flow constraints are given by $0.9 \leq \dot{x}_i \leq 1.1$ for all i, where x_i is the clock of the i-th process. The product automaton of n processes has 8^n locations and $2n$ variables.

Table 8: Computation Times of the FDDI Benchmark

instance	FDDIS05	FDDIU05	FDDIS06	FDDIU06	FDDIS07	FDDIU07	FDDIS08	FDDIU08
safety	safe	unsafe	safe	unsafe	safe	unsafe	safe	unsafe
# vars	15	15	18	18	21	21	24	24
# locs	7776	7776	46656	46656	279936	279936	1679616	1679616
tool	computation time in [s]							
bounded spec.	BDS05	BDU05	BDS06	BDU06	BDS07	BDU07	BDS08	BDU08
BACH	0.06	0.06	0.06	0.05	0.07	0.06	0.08	0.14
SAT-Reach	1.31	0.11	1.59	0.17	1.88	0.24	2.27	0.33
XSpeed	0.03	0.5	0.02	2.63	-	-	-	-

Table 9: Computation Times of the Ring-Shape Fischer Benchmark

instance	RINGS04	RINGU04	RINGS05	RINGU05	RINGS06	RINGU06
safety	safe	unsafe	safe	unsafe	safe	unsafe
# vars	8	8	10	10	12	12
# locs	4096	4096	32768	32768	262144	262144
tool	computation time in [s]					
bounded spec.	BDS04	BDU04	BDS05	BDU05	BDS06	BDU06
BACH	116.94	26.47	237.84	58.50	340.11	103.41

RINGS*nn* protocol with *nn* processes, considered safe with respect to specification BD*nn*.

RINGU*nn* protocol with *nn* processes, considered unsafe with respect to specification BD*nn*.

Specification The protocol is correct if no two processes are ever in the critical section at the same time.

For bounded state space, we have BD*nn*.

BD*nn* There are no two processes such that both are in location `loc8` (critical section) at the same time within default discrete search depth 30.

Results The computation times of various tools are listed in Tab. 9.

4 Conclusions and Outlook

This report presents the results of the eighth edition of a friendly competition for the formal verification of continuous and hybrid systems of the ARCH'24 workshop, in the category on PCDB: piecewise constant dynamics and BMC of such systems. The reports of other categories can be found in the proceedings and on the ARCH website: cps-vo.org/group/ARCH. The tool with which the results have been obtained is publicly available at gitlab.com/goranf/ARCH-COMP.

In the spirit of a friendly competition, this report does not provide any ranking of tools. For the reported instances, BACH solved almost all the cases, including the newly added ones, in the category of bounded problem efficiently.

5 Acknowledgments

Lei Bu and Yuhui Shi are supported by the National Natural Science Foundation of China (No.62172200, No.62232008) and the Leading-edge Technology Program of Jiangsu Natural Science Foundation (No. BK20202001).

References

[1] D. Adzkiya, B. De Schutter, and A. Abate. Computational techniques for reachability analysis of max-plus-linear systems. *Automatica*, 53(0):293–302, 2015.

[2] D. Adzkiya, B. De Schutter, and A. Abate. Finite bisimulations of max-plus-linear systems. *IEEE Transactions on Automatic Control*, 58(12):3039–3054, 2012.

[3] François Baccelli, Guy Cohen, Geert Jan Olsder, and Jean-Pierre Quadrat. *Synchronization and Linearity: An Algebra for Discrete Event Systems*. John Wiley & Sons Ltd, 1992.

[4] Sergiy Bogomolov, Goran Frehse, Mirco Giacobbe, and Thomas A. Henzinger. Counterexample-guided refinement of template polyhedra. In Axel Legay and Tiziana Margaria, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 23rd International Conference, TACAS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings, Part I*, volume 10205 of *Lecture Notes in Computer Science*, pages 589–606, 2017.

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

[6] 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.

[7] 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.

[8] Russell J. Clark and Amarnath Mukherjee. Book review: FDDI handbook: High speed networking using fiber and other media, by raj jain (addison-wesley 1994). *Comput. Commun. Rev.*, 24(2):44, 1994.

[9] Conrado Daws, Alfredo Olivero, Stavros Tripakis, and Sergio Yovine. The tool KRONOS. In *Hybrid Systems III: Verification and Control, Proceedings of the DIMACS/SYCON Workshop on Verification and Control of Hybrid Systems, October 22-25, 1995, Ruttgers University, New Brunswick, NJ, USA*, pages 208–219, 1995.

[10] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. SpaceEx: Scalable Verification of Hybrid Systems. In Shaz Qadeer Ganesh Gopalakrishnan, editor, *Proc. 23rd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2011.

[11] 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.

[12] 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.

[13] 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.

[14] Sumit Kumar Jha, Bruce H. Krogh, James E. Weimer, and Edmund M. Clarke. Reachability for linear hybrid automata using iterative relaxation abstraction. In *HSCC*, 2007.

[15] Atanu Kundu, Sarthak Das, and Rajarshi Ray. Sat-reach: A bounded model checker for affine hybrid systems. *ACM Transactions on Embedded Computing Systems*, 22(2):1–36, 2023.

[16] Leslie Lamport. A fast mutual exclusion algorithm. *ACM Transactions on Computer Systems (TOCS)*, 5(1):1–11, 1987.

[17] Rajarshi Ray, Amit Gurung, Binayak Das, Ezio Bartocci, Sergiy Bogomolov, and Radu Grosu. Xspeed: Accelerating reachability analysis on multi-core processors. In Nir Piterman, editor, *Hardware and Software: Verification and Testing - 11th International Haifa Verification Conference, HVC 2015, Haifa, Israel, November 17-19, 2015, Proceedings*, volume 9434 of *Lecture Notes in Computer Science*, pages 3–18. Springer, 2015.

[18] Subiono. *On classes of min-max-plus systems and their applications*. PhD thesis, Delft University of Technology, 2000. http://resolver.tudelft.nl/uuid:e5181e99-fb8f-4588-a37a-46ff2007c5c7.

[19] Farn Wang. Symbolic parametric safety analysis of linear hybrid systems with bdd-like data-structures. *IEEE Trans. Software Eng.*, 31(1):38–51, 2005.

[20] Yuming Wu, Lei Bu, Jiawan Wang, Xinyue Ren, Wen Xiong, and Xuandong Li. Mixed semantics guided layered bounded reachability analysis of compositional linear hybrid automata. In Bernd Finkbeiner and Thomas Wies, editors, *Verification, Model Checking, and Abstract Interpretation - 23rd International Conference, VMCAI 2022, Philadelphia, PA, USA, January 16-18, 2022, Proceedings*, volume 13182 of *Lecture Notes in Computer Science*, pages 473–495. Springer, 2022.

[21] 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.

[22] 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.

A Implementation Languages and Used Machines

A.1 BACH

- Implementation language: C++
- Processor: 12th Gen Intel(R) Core(TM) i5-12500 @ 3.0GHz
- Memory: 16 GB

- Average CPU Mark on www.cpubenchmark.net: 19905 (full), 3694 (single thread)

A.2 SAT-Reach

- Implementation language: C++
- Processor: AMD (R) Ryzen 7 5800U @ 1.90 x 8
- Memory: 16 GB
- Average CPU Mark on www.cpubenchmark.net: 18861 (full), 3092 (single thread)

A.3 XSpeed

- Implementation language: C++
- Processor: AMD (R) Ryzen 7 5800U @ 1.90 x 8
- Memory: 16 GB
- Average CPU Mark on www.cpubenchmark.net: 18861 (full), 3092 (single thread)