ARCH-COMP20 Category Report: Stochastic Models

Alessandro Abate[1], Henk Blom[2], Nathalie Cauchi[1], Joanna Delicaris[3], Arnd Hartmanns[4], Mahmoud Khaled[5], Abolfazl Lavaei[6], Carina Pilch[3], Anne Remke[3], Stefan Schupp[7], Fedor Shmarov[8], Sadegh Soudjani[8], Abraham P. Vinod[9], Ben Wooding[8], Majid Zamani[10], and Paolo Zuliani[8]

[1] University of Oxford, Oxford, UK
[2] Delft University of Technology, Delft, The Netherlands
[3] University of Münster, Germany
[4] University of Twente, Enschede, The Netherlands
[5] Technical University of Munich, Germany
[6] LMU Munich, Germany
[7] RWTH Aachen University, Aachen, Germany
[8] Newcastle University, Newcastle upon Tyne, UK
[9] The University of Texas at Austin, USA
[10] University of Colorado Boulder, USA

Abstract

This report presents the results of a friendly competition for formal verification and policy synthesis of stochastic models. It also introduces new benchmarks within this category, and recommends next steps for this category towards next year's edition of the competition. The friendly competition took place as part of the workshop Applied Verification for Continuous and Hybrid Systems (ARCH) in Spring/Summer 2020.

1 Introduction

Disclaimer The presented report of the ARCH friendly competition for stochastic modelling group aims at providing a unified point of reference on the current state of the art in the area of stochastic models together with the currently available tools and framework for performing formal verification and optimal policy synthesis to such models. We further provide a set of benchmarks which we aim to push forward the development of current and future tools. To establish further trustworthiness of the results, the code describing the benchmarks together with the code used to compute the results is publicly available at https://gitlab.com/goranf/ARCH-COMP.

This friendly competition is organized by Alessandro Abate (alessandro.abate@cs.ox.ac.uk), Stefan Schupp (stefan.schupp@cs.rwth-aachen.de), and Sadegh Soudjani (sadegh.soudjani@newcastle.ac.uk). The authors acknowledge the contributions by Henk Blom (H.A.P.Blom@tudelft.nl) in preparing an overview of the tools, benchmarks, and their properties, which will be used in future editions of the competition.

This report presents the results obtained during the ARCH Friendly Competition 2020 in the group *stochastic models*. The benchmark collection as been extended by 3 interesting benchmarks which exhibit novel challenges for tools participating in this category.

In this year's edition overall 9 tools and frameworks participated in the evaluation of earlier and new benchmarks. This year the following tools and frameworks participate (in alphabetical order): AMYTISS, FAUST^2, hpnmg, Mascot-SDS, modes, ProbReach, SDCPN & IPS, SReachTools, and StocHy. Following the general tendency of centralized execution, one additional goal this year was the execution of the tools on centralized cloud services to obtain comparable results among the participating tools.

Furthermore, all the tools and frameworks have been compiled into Docker format (a container software), which allows for easier readability evaluation of the generated results together with the sharing of the tools themselves to both the ARCH and the wider research community.

This report has the following structure. Section 2 provides a short overview of the participating tools and frameworks. Section 3 presents already established benchmarks and a set of new benchmark descriptions, which include a discussion of the individual models syntax and semantics, and a presentation of the specifications of interest is presented in Section 4. Next, in Section 5 we present the results of the friendly competition where the participating tools or algorithmic frameworks that are used to solve instances of the collection of benchmarks. We identify key challenges and discuss future plans in Section 6.

2 Participating Tools & Frameworks

Here we present the tools which participated this year in alphabetical order.

AMYTISS AMYTISS is a software tool, implemented as a kernel on top of the acceleration ecosystem pFaces [41], for designing correct-by-construction controllers of stochastic discrete-time systems. It implements parallel algorithms to (1) build finite Markov decision processes (MDPs) as finite abstractions of given original stochastic discrete-time systems, and (2) synthesize controllers for the constructed finite MDPs satisfying bounded-time safety specifications and reach-avoid specifications. The underlying computation parts are similar to the ones used in FAUST^2 and StocHy, and are used for compositional computations [44, 45, 46, 47, 48]. This tool significantly improves performances w.r.t. the computation time by parallel execution in different heterogeneous computing platforms including CPUs, GPUs and hardware accelerators (*e.g.,* FPGA). In addition, AMYTISS proposes a technique to reduce the required memory for computing finite MDPs as *on-the-fly abstractions* (OFA). In the OFA technique, computing and storing the probability transition matrix are skipped. Instead the required entries of the finite MDP are on-the-fly computed as they are needed for the synthesis part via the standard dynamic programming. This technique impressively reduces the required memory but at the cost of repeated computation of their entries in each time step from 1 to a finite-time horizon T_d. This gives the user an additional control over the trade-off between the computation time and memory usage. The tool is available at https://github.com/mkhaled87/pFaces-AMYTISS.

FAUST^2 The tool FAUST^2 [69](*Formal Abstractions of Uncountable-STate STochastic processes*) generates formal abstractions of discrete-time Markov processes (dtMP) defined over continuous state spaces. The dtMP model is abstracted into a finite-state Markov chain or a Markov decision process. The abstract model is formally put in relationship with the concrete dtMP via a user-defined maximum threshold on the approximation error introduced by the abstraction procedure. FAUST^2 allows exporting the abstract model to well-known probabilistic model checkers, such as PRISM [42] or STORM [19]. Alternatively, it can handle internally the computation of basic PCTL properties (e.g. safety or reach-avoid) over the abstract model, and refine the outcomes over the concrete dtMP via a quantified error that depends on the abstraction procedure and the given formula. The toolbox relies on approaches developed and adapted to different classes of systems and under different assumptions [62, 64, 66, 67] and is used in solving various types of verification and synthesis problems [68, 70, 71]. The tool is available at https://sourceforge.net/projects/faust2/.

hpnmg The tool hpnmg [36] is a model checker for Hybrid Petri nets with an arbitrary but finite number of general transition firings against specifications formulated in STL [38]. Each general transition firing results in a random variable which follows a continuous probability distribution. It efficiently implements and combines algorithms for a symbolic state-space creation [37], transformation to a geometric representation as convex polytopes [39], model checking a potentially nested STL formula and integrating over the resulting satisfaction set to yield the probability that the specification holds at a specific time.

The tool is implemented in C++ and relies on the library HyPro [57] for efficient geometric operations on convex polytopes, as well as on the GNU Scientific Library (GSL) for multi-dimensional integration using Monte Carlo integration [49]. It is available at https://zivgitlab.uni-muenster.de/ag-sks/tools/hpnmg.

HYPEG The Java-based library HYPEG [54] is a dedicated statistical simulator for hybrid Petri nets with general transitions (HPnGs) [29], which combine discrete and continuous components with a possibly large number of random variables, whose stochastic behavior follows arbitrary probability distributions. HYPEG uses time-bounded discrete-event simulation and well-known statistical model checking techniques to verify complex properties, including time-bounded reachability [55]. These techniques comprise several hypothesis tests as well as different approaches for the computation of confidence intervals. In the latest version of HYPEG, continuous behavior that can be expressed by systems of ordinary differential equations can be simulated using an approximative approach [53, 51], whereas piecewise-linear continuous behavior is simulated without approximation.

The tool is available at https://zivgitlab.uni-muenster.de/ag-sks/tools/HYPEG.

Mascot-SDS Newcastle (SS) Mascot-SDS [50] is an open-source tool for synthesizing controllers with formal correctness guarantees for discrete-time dynamical systems in the presence of stochastic perturbations. The tool that supports infinite-horizon control specifications for stochastic dynamical systems and computes over- and under-approximations of the winning domain for the specification. The current version of the tool is developed for the "always eventually" specification (repeated reachability), but its new version that is scheduled to be released this year will handle all omega-regular specifications including Linear Temporal Logic properties. Mascot-SDS is written in C++, and is an extension of Mascot [35]. The suffix SDS stands for Stochastic Dynamical Systems.

The tool is available at https://gitlab.mpi-sws.org/kmallik/mascot-sds.

The Modest Toolset The Modest Toolset [34] supports the modelling and analysis of hybrid, real-time, distributed and stochastic systems. At its core is the model of networks of stochastic hybrid automata (SHA) [33], which combine nondeterministic choices, continuous system dynamics, stochastic decisions and timing, and real-time behaviour, including nondeterministic delays. The Modest Toolset is a modular framework, supporting as input the high-level Modest modelling language [6, 33] and the Jani specification [9], and providing a variety of analysis backends for various special cases of SHA. In particular, the modes discrete-event simulator [8] supports SHA without nondeterminism and linear dynamics. It includes a highly-automated rare event simulation engine based on importance splitting [7], and provides statistical estimates with configurable error and confidence levels. The prohver tool [33] model-checks SHA with linear differential equations and inclusions, combining abstraction of continuous probability distributions with a non-stochastic hybrid automata reachability analysis (using PHAVer [25] as backend). It delivers guaranteed upper bounds on (time-bounded) reachability probabilities.

The Modest Toolset can can be obtained from http://www.modestchecker.net/.

ProbReach ProbReach [59] provides a set of algorithms for computing probabilistic bounded reachability in *parametric* stochastic hybrid systems (PSHS) with nonlinear continuous dynamics (i.e., defined by nonlinear ODEs). The parameters can be random (continuous and discrete) and/or nondeterministic. ProbReach features a formal approach [61] for computing numerically sound probability intervals that are formally guaranteed to contain the exact value of the bounded reachability probability. For PSHSs containing random parameters only, the size of such interval can be made arbitrarily small. Also, ProbReach implements Monte Carlo algorithms [60] for computing confidence intervals for the bounded reachability probability with rigorous (i.e., numerically sound) sampling. ProbReach uses the Probabilistic Delta-ReacHability (PDRH) format for encoding PSHSs, and it is available at https://github.com/dreal/probreach.

SReachTools SReachTools [74] is an open-source, repeatability-evaluated, MATLAB toolbox for tackling the problem of stochastic reachability of a target tube [78]. This problem subsumes terminal hitting-time stochastic reach-avoid and stochastic viability problems (guaranteeing safety in stochastic systems) [72, 3]. SReachTools handles linear, discrete-time, continuous-state dynamical systems with additive stochastic disturbance. The dynamics and the safety constraints can be time-varying, and the disturbance may be Gaussian or non-Gaussian. It relies on approaches drawn from convex optimization, Fourier transforms, scenario-based optimization, and computational geometry for a grid-free and scalable computation of the stochastic reach sets as well as controller (open-loop, affine feedback, and set-based) synthesis [28, 75, 76, 79, 56].

SReachTools is available at https://sreachtools.github.io. The code for solving the benchmarks presented in this report is available as a Code Ocean capsule at https://doi.org/10.24433/CO.5339956.v1.

StocHy StocHy [13] is a software tool for the quantitative analysis of discrete-time *stochastic hybrid systems* (SHS) accepts a high-level description of stochastic models and constructs an equivalent SHS model. In comparison with the other tools in the stochastic modelling category, StocHy is the only tool that provides exact (i.e. not via statistical means) errors/guarantees on the obtained solution [14, 30]. The tool enables users to (i) simulate the SHS evolution over a given time horizon; and to (ii) to automatically construct formal abstractions of the SHS using either abstractions taking the form of Markov Decision Processes (MDP) or grounded on interval MDPs (IMDP) [14]. The abstractions are then employed for (ii) formal verification or (iii) control (policy, strategy) synthesis. StocHy allows for modular modelling, and has separate simulation, verification and synthesis engines, which are implemented as independent libraries. The tool is implemented in C++ and employs manipulations based on vector calculus,sparse matrices, symbolic construction of probabilistic kernels, and multi-threading. The tool can be obtained from https://github.com/natchi92/stochy.

2.1 Frameworks

In contrast to complete tools, frameworks usually provide a collection of algorithms and data structures or collect several tools for different sub-problems into one library.

SDCPN & IPS This is a reach probability modelling and estimation framework that has been developed for the evaluation of multi-actor air traffic designs on mid-air collision risk. Because this air traffic application domain is very demanding, the selected mathematical setting is Generalized Stochastic Hybrid Processes (GSHP) [10]. GSHP incorporates Brownian motion in continuous-time Piecewise Deterministic Markov Processes [18]. Because a direct specification of a large GSHP model does not work, Stochastically and Dynamically Coloured Petri Nets (SDCPN) are used for the compositional modelling of a GSHP [21, 22, 23, 24]. For the acceleration of MC simulation of rare events, the Interacting Particle Systems (IPS) approach for GSHP is used [15, 5]. The SDCPN & IPS framework has been applied to the Heated Tank benchmark.

(ϵ, δ) **Abstraction** Based on the papers [30, 32, 58], this software library uses code snippets and algorithms to compute two precision parameters (ϵ, δ), which allow bounding the deviations between models in both the output signals (ϵ) and the transition probabilities (δ). The obtained abstract models, either with deterministic continuous states or with stochastic finite states, are then employed in probabilistic model checking.

3 Established benchmarks, revisited

Benchmarks already established allow to visualise the development progress that a tool makes during its live-cycle. Therefore, the collection of benchmarks used for evaluation does not only consider new benchmarks but also re-uses existing benchmarks to allow tool developers to improve their results from previous years. In this section we give a short description of established benchmarks that have been used in the evaluation with references to their original sources (and previous reports in ARCH) for further details.

3.1 Automated Anesthesia benchmark

The automated anesthesia delivery system benchmark, introduced in ARCH 2018 [2], seeks a safe automated delivery system, with and without the human-in-the-loop (anestheologist). The safety specification requires the depth of hypnosis of a patient to stay within pre-specified safe bounds for a given period of time. The continuous-state system model for the patient's depth of hypnosis is based on a three-compartment pharmacokinetic system (LTI system). We account for the variation in the system model for different patients using an additive Gaussian disturbance. We obtain a stochastic hybrid system model when the inputs from the anestheologist is also modeled.

3.2 Building Automation Systems benchmark

The Building Automation System (BAS) benchmark, also introduced in ARCH 2018, is built upon the library of stochastic models presented in [12, 11] and based on earlier work [65, 63]. This library allows us to generate benchmarks with different levels of complexities which can take the form of the general framework of stochastic hybrid systems. In this instance we construct two different benchmarks where the focus is on synthesis and on the number of continuous dimensions being modeled. For each (i) we provide a high-level description of the models and the specification of interest:

Thermal model with 4-dimensions: This model is a two zone thermal model consisting of a single discrete location and 4 continuous variables which evolve according to a stochastic difference equation. The model details can be found in [2]. We are interested in generating a control policy which ensures the comfort range of the temperature in each zone is kept within the range [19.5 20.5] , when using a control input signal which lies within the range of [15 22], for a specific time horizon for 1.5 hours (i.e. 6 time steps), with a minimum likelihood of 0.8.

Thermal model with 7-dimensions: The model is a discrete-time Gaussian-perturbed stochastic linear system with 7-dimensional state, 1-dimensional control input, and a 6 - dimensional Gaussian disturbance vector. The model details can be found in [2]. We would like to synthesise a policy ensuring that the temperature within zone 1 does not deviate from the set point by more than 0.5 °C over a time horizon equal to 1.5 hours.

3.3 Heated Tank

The Heated Tank benchmark stems from safety literature, where it is a well-known example of a Piecewise Deterministic Markov Process (PDMP) [18]. This made the Heated Tank benchmark a logical candidate for inclusion in the set of ARCH stochastic models [2, 1].

The heated tank system consists of a tank containing liquid whose level is influenced by two pumps and one valve managed by a controller. The purpose of the liquid in the tank is to absorb and transport the heat from a heat source; this means that under nominal conditions one of the pumps produces a constant inflow of cool liquid, and a similar flow of heated liquid leaves the tank through the valve. The Euclidean valued state components are the height $x_{H,t}$ and temperature $x_{T,t}$ of the liquid in the tank at moment t. Pumps and Valve may fail, and a Controller switches Pumps or Valve if the height of the liquid becomes too high or too low. The rare event probabilities, to be estimated on time interval $[t_0, t_{end}]$, are: Dryout probability, Overflow probability, and Overheating probability. In literature the heated tank benchmark has five versions: 1) Pumps and Valve have constant failure rates; 2) Pumps and Valve have mode dependent failure rates; 3) In version 1, Controller may fail to implement its switching decision; 4) In version 1, Pumps and Valve are repaired; and 5) In version 1, failure rates depend on the liquid temperature.

Of these five versions, version 4 is the only one where the Dryout probability reaches small values. This makes version 4 (= 4.0) the best choice for a rare event estimation benchmark. Version 4.0 has been described in words [2], and in a formal way using formal models in Modest [33] and from SDCPN and HYPEG [1]. In [1] Heated Tank version 4.0 has been enriched with the following five extensions:

4.I. Discrete-valued process influences repair rates or failure rates (e.g. in version 2).

4.II. Probability of non-communicating a decision made to an applicable entity (e.g. in version 3).

4.III. Continuous-valued process influences repair or failure rates (e.g. in version 5).

4.IV. Non-exponentially distributed duration of working and/or repair of system components.

4.V. Brownian motion in a Euclidean state component, e.g. in the heat source.

In 2018 and 2019, benchmark version 4.0 only has been evaluated.

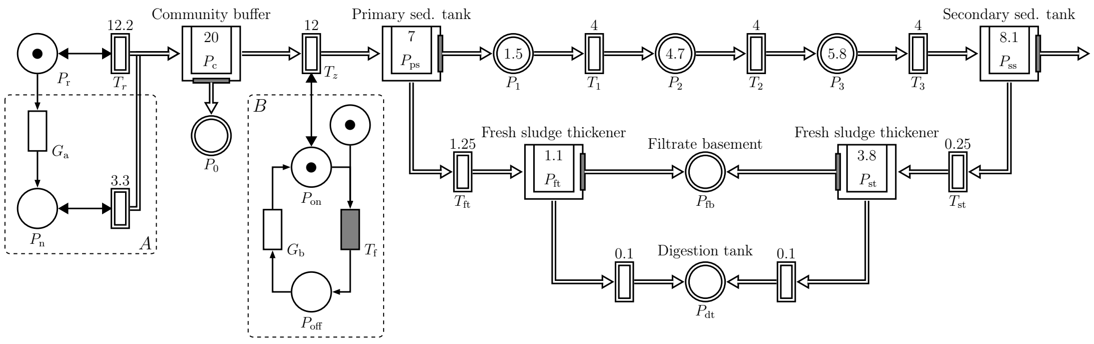

Figure 1: Water sewage facility with extensions A and B. The unit of all capacities for the continuous and overflow places is 10^6 l and the unit of all continuous transition rates is 10^6 l/h.

3.4 Water Sewage Facility

The water sewage benchmark models a water sewage treatment plant situated in Enschede as a hybrid Petri net with general transitions (HPnG). It has been proposed and thoroughly analyzed in [26] and was already included in ARCH2019 [1].

The model consists of a cleaning street with multiple so-called *overflow places*, which are modeled as continuous variables with a predefined upper bound. When the overflow place reaches its upper bound it releases the excess fluid to another place via a designated transition. The model makes excessive use of *rate adaptation*, which is a control mechanism unique to hybrid Petri nets and ensures that the inflow / outflow of a place that is at either capacity is adapted in order to prevent over- or underflow. This leads to a large number of state changes when executing the benchmark.

The report from 2019 already featured two different failure scenarios for the water sewage facility. Here, we want to formalize these scenarios A and B and introduce properties for model checking the survivability of the system [27]. Note that, *survivability evaluation* is a measure of dependability, which quantifies the ability of a system to recover after the occurrence of a disaster to a predefined level of service in a timely manner [16].

Figure 1 gives an illustration of the hybrid Petri net model for the water sewage with extensions A and B. Initially, all finite places of the water sewage facility are considered to be full except for the community buffer P_c.

Extension A initializes place P_r with one token, modeling heavy rain. The duration of this heavy rain is modeled by general transition G_a, which moves the token to place P_n and hence disables the continuous transition T_r with rate $12.2 \cdot 10^6$ liters per hour and enables the continuous transition with rate $3.3 \cdot 10^6$ liters per hour, effectively decreasing the input to the system. The general transition G_a follows a folded normal distribution $\mathcal{N}_f$ with mean μ and standard deviation $\sigma = 1\,\text{h}$. For the analysis, μ as well as the capacity of the community buffer P_c, are parametrized.

Extension B models the failure of the input transition T_z to the water sewage facility, which forms the bottleneck of the system. After a deterministic time delay α, the pump fails, after which repair is initiated. The delay for that is modeled with a random variable which is exponentially distributed with rate $\lambda = 2$ per hour. For the analysis of the system, the deterministic time delay α, as well as the rate of rainfall to the water sewage facility are parametrized.

For each extension dedicated properties have been formulated in STL. Both properties ensure that the community buffer is not exceeded and have been discussed in [1].

Overflow initiated by heavy rain (φ_A) This property ensures that the community buffer P_c does not overflow before heavy rain stops. Hence, property φ_A states that the overflow of the community buffer P_c is below a predefined small value until the goal state is reached after at most 30 hours and the token has moved to place P_n:

$$\varphi_A := x_{P_0} \leq 0.1 \;\; \mathcal{U}^{[0,30]} \;\; m_{P_\text{n}} = 1.$$

Overflow initiated by pump failure (φ_B) This property states that the overflow of the community buffer is below a given threshold until the input transition is repaired. The interval of the until operator starts with α, which indicates the time point of the failure of the input transition. The system is then required to become operational again within 30 hours without spilling fluid from the community buffer. This is formalized as φ_B:

$$\varphi_B := x_{P_0} \leq 0.01 \;\; \mathcal{U}^{[\alpha,\alpha+30]} \;\; m_{P_\text{on}} = 1.$$

Transformation To make the benchmark accessible for tools that operate on stochastic hybrid automata, e.g. modes and prohver, the HPnG has been transformed into a hybrid automaton with stochastic resets, according to the transformation proposed in [52]. This transformation constructs a hybrid automaton building on the symbolic state space, which is represented as a finite parametric location tree [37]. The parametric location tree, which also forms the basis for the analytical model checker hpnmg, is constructed up to a certain time t. Note that, for this model depending on the parameter setting, the state space is finite as after a certain time, no state changes occur. We have chosen the time t for each parameter setting, such that the full state space is included in the transformation.

Figure 2 shows an example for the transformed automaton for extension A with a specific parameter setting. Property φ_A is fulfilled as soon as location l_2 is reached. However, if too much time is spent in location l_5, the property is violated.

4 New Benchmarks

Tool development strongly profits from large sets of benchmarks for evaluation of the implemented approach. Additionally to the already existing benchmarks which have been proposed in the last years [1, 2], in this section we present new benchmarks (in alphabetical order) which have been proposed this year for evaluation and introduce new challenges that are not currently handled by existing benchmarks.

4.1 Stochastic Van der Pol Oscillator

The state evolution of the oscillator is given by:

$$\begin{aligned} x_1(k+1) &= x_1(k) + x_2(k)\tau + w_1(k) \\ x_2(k+1) &= x_2(k) + (-x_1(k) + (1 - x_1(k)^2)x_2(k))\tau + w_2(k), \end{aligned} \tag{1}$$

where the sampling time τ is set to $0.1s$ and $(w_1(k), w_2(k))$ is a pair of stochastic noise signals at time k drawn from a uniform density function with a compact support $D = [-0.02, 0.02] \times [-0.02, 0.02]$.

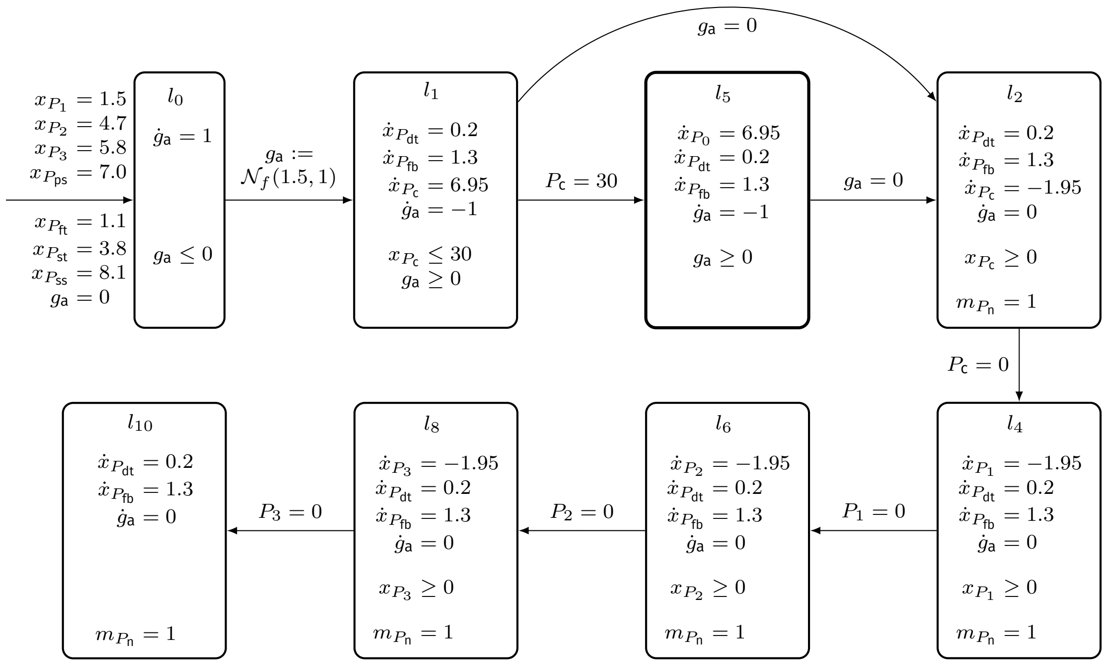

Figure 2: Transformed automaton for the water sewage facility with extension A and $\mu = 1.5$h and a capacity of $30 \cdot 10^6$l for the community buffer P_c. The random variable that describes the duration of the heavy rain is represented by the stochastic clock g_a. Note that all values not shown in the figure are zero.

Consider a safety specification $\square A$ for staying within the working area $A = [-5, 5] \times [-5, 5]$, as well as a Büchi specification $\square\lozenge B$ for repeatedly reaching the target set $B = [-1.2, -0.9] \times [-2.9, -2]$.

Problem 1 (Qualitative Verification)**.** *Compute the set of initial states from which the probability of satisfying the specification $\square A \wedge \square\lozenge B$ under dynamics (1) is equal to one.*

Problem 2 (Quantitative Verification)**.** *Compute the probability of satisfying the specification $\square A \wedge \square\lozenge B$ under dynamics (1) as a function of initial state.*

Since some of the tools are not able to handle $\square\lozenge B$, the following modified dynamical system can be used together with a reachability specification that gives an upper-bound for probability of satisfying $\square A \wedge \square\lozenge B$. Let us denote the right-hand side of (1) by $f(x(k)) + w(k)$. Define a new dynamical system with state space $A \cup \{\varphi_1, \varphi_2\}$ such that φ_1 and φ_2 are sink states and

$$x(k+1) = \begin{cases} f(x(k)) + w(k) & \text{if } w(k) \in A \backslash f(x(k)) \text{ and } x(k) \notin B \\ \varphi_1 & \text{if } w(k) \notin A \backslash f(x(k)) \text{ and } x(k) \notin B \\ f(x(k)) + w(k) & \text{if } w(k) \in A \backslash f(x(k)) \text{ and } x(k) \in B \text{ and } \nu(k) = 0 \\ \varphi_1 & \text{if } w(k) \notin A \backslash f(x(k)) \text{ and } x(k) \in B \text{ and } \nu(k) = 0 \\ \varphi_2 & \text{if } w(k) \in A \backslash f(x(k)) \text{ and } x(k) \in B \text{ and } \nu(k) = 1 \\ \varphi_2 & \text{if } w(k) \notin A \backslash f(x(k)) \text{ and } x(k) \in B \text{ and } \nu(k) = 1, \end{cases} \tag{2}$$

where $\nu(k)$ are independent and identically distributed Bernoulli random variables with success probability $(1-\zeta)$.

Problem 3 (Quantitative Reachability)**.** *Compute the probability $\Diamond\varphi_2$ under dynamics* (2)*.*

The solution of Problem 3 is an upper bound for Problem 2. Moreover, it converges to the solution of Problem 2 when $\zeta \to 1^-$.

The dynamics in (1) can be extended to include inputs for shaping the limiting behaviour of the system. Consider the non-autonomous version of the oscillator dynamics given by:

$$\begin{aligned} x_1(k+1) &= x_1(k) + x_2(k)\tau + w_1(k) \\ x_2(k+1) &= x_2(k) + (-x_1(k) + (1-x_1(k)^2)x_2(k))\tau + u(k)w_2(k). \end{aligned} \tag{3}$$

Problem 4 (Quantitative Synthesis)**.** *Compute a policy for dynamical system* (3) *that maximises the probability of satisfying $\Box A \wedge \Box\Diamond B$.*

4.2 Integrator-Chain (for scalability comparison)

Consider a chain of integrators,

$$x_{k+1} = \begin{bmatrix} 1 & N_s & \frac{N_s^2}{2} & \cdots & \frac{N_s^{n-1}}{(n-1)!} \\ 0 & 1 & N_s & \cdots & \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{N_s^n}{n!} \\ \frac{N_s^{n-1}}{(n-1)!} \\ \vdots \\ N_s \end{bmatrix} u_k + w_k \tag{4}$$

with state $x_k \in \mathbb{R}^n$, input $u_k \in \mathcal{U} = [-1,1]$, a Gaussian disturbance $w_k \sim \mathcal{N}(\overline{0}_2, 0.01 I_2)$, sampling time $N_s = 0.1$, and time horizon $N = 5$. Here, I_n refers to the n-dimensional identity matrix and $\overline{0}_n$ is the n-dimensional zero vector. Denote the set of Markov policies as $\mathcal{M}$. For any policy $\pi \in \mathcal{M}$ and an initial state $x_0 \in \mathbb{R}^n$, $\mathbb{P}_X^{\pi,x_0}$ denotes the probability measure of the resulting trajectory (random) vector $X = [x_1\ x_2\ \dots\ x_N] \in \mathbb{R}^{Nn}$ of the system (4).

We wish to solve the stochastic reach-avoid problem for the system (4) with safe set $\mathcal{S} = [-10,10]^n$ and target set $\mathcal{T} = [-8,8]^n$. Specifically, we compute the stochastic reach-avoid set $\mathcal{L}(\alpha) \subseteq \mathcal{S}$ for some $\alpha \in [0,1]$,

$$\mathcal{L}(\alpha) = \{x_0 \in \mathcal{S} : \exists \pi \in \mathcal{M}, \mathbb{P}_X^{\pi,x_0}\{x_1 \in \mathcal{S}, \dots, x_{N-1} \in \mathcal{S}, x_N \in \mathcal{T}\} \geq \alpha\}. \tag{5}$$

4.3 7-Dimensional BMW 320i

We consider a vehicle described by the following hybrid 7-dimensional *nonlinear* single track (ST) model of a BMW 320i car [4, Section 5.1] by including the stochasticity inside the dynamics as the additive noise:
For $|x_4(k)| < 0.1$:

$$\begin{aligned} x_i(k+1) &= x_i(k) + \tau a_i(k) + R_i\varsigma_i(k), \quad i \in \{1,\dots,7\}\backslash\{3,4\}, \\ x_3(k+1) &= x_3(k) + \tau \mathrm{Sat}_1(\nu_1) + 0.2\varsigma_3(k), \\ x_4(k+1) &= x_4(k) + \tau \mathrm{Sat}_2(\nu_2) + 0.1\varsigma_4(k), \end{aligned}$$

for $|x_4(k)| \geq 0.1$:

$$\begin{aligned}
x_i(k+1) &= x_i(k) + \tau b_i(k) + R_i \varsigma_i(k), \quad i \in \{1,\dots,7\} \backslash \{3,4\},\\
x_3(k+1) &= x_3(k) + \tau \mathrm{Sat}_1(\nu_1) + 0.2\varsigma_3(k),\\
x_4(k+1) &= x_4(k) + \tau \mathrm{Sat}_2(\nu_2) + 0.1\varsigma_4(k),
\end{aligned}$$

where,

$$\begin{aligned}
&R_1 = R_2 = 0.25, \quad R_5 = R_6 = R_7 = 0.2, \quad a_1 = x_4 \cos(x_5(k)), \quad a_2 = x_4 \sin(x_5(k)),\\
&a_5 = \frac{x_4}{l_{wb}} \tan(x_3(k)), \quad a_6 = \frac{\nu_2(k)}{l_{wb}} \tan(x_3(k)) + \frac{x_4}{l_{wb} \cos^2(x_3(k))} \nu_1(k), \quad a_7 = 0,\\
&b_1 = x_4(k) \cos(x_5(k) + x_7(k)), \quad b_2 = x_4(k) \sin(x_5(k) + x_7(k)), \quad b_5 = x_6(k),\\
&b_6 = \frac{\bar{\mu} m}{I_z(l_r + l_f)} (l_f C_{S,f}(g l_r - \nu_2(k) h_{cg}) x_3(k) + (l_r C_{S,r}(g l_f + \nu_2(k) h_{cg}) - l_f C_{S,f}(g l_r\\
&\qquad - \nu_2(k) h_{cg})) x_7(k) - (l_f^2 C_{S,f}(g l_r - \nu_2(k) h_{cg}) + l_r^2 C_{S,r}(g l_f + \nu_2(k) h_{cg})) \frac{x_6(k)}{x_4(k)}),\\
&b_7 = \frac{\bar{\mu}_f}{x_4(k)(l_r + l_f)} (C_{S,f}(g l_r - \nu_2(k) h_{cg}) x_3(k) + (C_{S,r}(g l_f + \nu_2(k) h_{cg}) + C_{S,f}(g l_r\\
&\qquad - \nu_2(k) h_{cg})) x_7(k) - (l_f C_{S,f}(g l_r - \nu_2(k) h_{cg}) - l_r C_{S,r}(g l_f + \nu_2(k) h_{cg})) \frac{x_6(k)}{x_4(k)}) - x_6(k).
\end{aligned}$$

Here, $\mathrm{Sat}_1(\cdot)$ and $\mathrm{Sat}_2(\cdot)$ are input saturation functions introduced in [4, Section 5.1], x_1 and x_2 are the position coordinates, x_3 is the steering angle, x_4 is the heading velocity, x_5 is the yaw angle, x_6 is the yaw rate, and x_7 is the slip angle. Variables ν_1 and ν_2 are inputs and they control the steering angle and heading velocity, respectively.

The model takes into account the tire slip making it a good candidate for studies that consider planning of evasive maneuvers that are very close to physical limits. We consider an update period $\tau = 0.1$ seconds and the following parameters for a BMW 320i car: $l_{wb} = 2.5789$ as the wheelbase, $m = 1093.3$ [kg] as the total mass of the vehicle, $\bar{\mu} = 1.0489$ as the friction coefficient, $l_f = 1.156$ [m] as the distance from the front axle to the center of gravity (CoG), $l_r = 1.422$ [m] as the distance from the rear axle to CoG, $h_{cg} = 0.6137$ [m] as the height of CoG, $I_z = 1791.6$ [kg m^2] as the moment of inertia for entire mass around z axis, $C_{S,f} = 20.89$ [1/rad] as the front cornering stiffness coefficient, and $C_{S,r} = 20.89$ [1/rad] as the rear cornering stiffness coefficient.

To construct a finite MDP $\widehat{\Sigma}$, we consider a bounded version of the state set $X := [-10.0, 10.0] \times [-10.0, 10.0] \times [-0.40, 0.40] \times [-2, 2] \times [-0.3, 0.3] \times [-0.4, 0.4] \times [-0.04, 0.04]$, a state discretization vector $[4.0; 4.0; 0.2; 1.0; 0.1; 0.2; 0.02]$, an input set $U := [-0.4, 0.4] \times [-4, 4]$, and an input discretization vector $[0.2; 2.0]$.

We are interested in an autonomous operation of the vehicle. The vehicle should park itself automatically in the parking lot located in the projected set $[-1.5, 0.0] \times [0.0, 1.5]$ within 32 time steps. The vehicle should avoid hitting a barrier represented by the set $[-1.5, 0.0] \times [-0.5, 0.0]$.

5 Friendly Competition – Setup and Outcomes

The results observed from the execution of the previously described benchmarks with the participating tools are presented in this section, an overview on which tool was used on which benchmark is given in Table 1.

Table 1: Tool-benchmark matrix: We indicate the year a tool was first applied to a given benchmark. Shortkeys: automated anesthesia (AS), building automation (BA), heated tank (HT), water sewage (WS), stochastic Van der Pol (VP), integrator chain (IC), and autonomous vehicle (AV).

Tool	Benchmarks						
	AS	BA	HT	WS	VP	IC	AV
FAUST[2]	2018	2018				2020	
StocHy	2019	2019				2020	
SReachTools	2018	2018				2020	
AMYTISS	2020	2020			2020	2020	2020
hpnmg				2020			
HYPEG			2019	2020			
Mascot-SDS					2020		
modes			2018	2020			
ProbReach				2020			
prohver			2020	2020			
SDCPN&IPS			2019				
(ϵ, δ) Abstraction	2019	2019					

Since the start of the ARCH stochastic modelling group in 2017, relevant benchmarks have slowly but steadily been developed, as the stochastic model area is very large the variation in benchmarks also is broad. In order to get grip on this, in Table 2 we have given an overview of the different aspects that are addressed by the set of established benchmarks that have been evaluated within the group (indicated in Table 1).

As we aimed for a centralized evaluation, the used machines for evaluation are described in Section 5.1 followed by detailed results for each benchmark.

5.1 Platforms

In contrast to previous years, this year we targeted a centralized execution of the benchmarks to obtain comparable results. This development follows the general tendency of the ARCH-COMP where centralized execution via Docker-Containers is used this year.

To allow comprehensive comparison of results which also allow tools designed for multi-core architectures to highlight their capabilities, we were provided three machines from the Amazon Web Services (AWS) cloud. **We are grateful that the repeatability experiment was funded by Amazon Web Services with a donation of $5000 provided as computing time on the AWS machines described below.**

All participants were provided with access to three machines from Amazon AWS where each of the used machines belongs to a different computing class. The first machine (labeled CPU_1 in the tables) belongs to the so-called class M4, which provides a balance of compute, memory, and network resources, and it is a good choice for many small-sized applications. The used machine has an Intel CPU E5-2686 v4 processor with 8 CPU cores running on a frequency of 2.30 GHz, and 32 GB memory. The second machine (labeled CPU_2 in the tables) belongs to the class C5, which delivers high performance computing (HPC) for applications running advanced compute-intensive workloads. The machine has an Intel Xeon Platinum 8000 processor with 72 CPU cores running on a frequency of 3.60 GHz, and 144 GB memory. The last machine (labeled GPU_1 in the tables) belongs to class P3, which is based on NVIDIA Tesla GPUs

Table 2: Overview of benchmark properties. Shortkeys: Time horizon: Finite (F) or Infinite (I); Type of control: Switching (S), Drift (Dr), or Multiple (M); Time line: Discrete (D) or Continuous (C); State space: Continuous (C) or Hybrid (H); Drift in ODE/SDE: Linear (L), Piecewise Linear (pL), or Nonlinear (NL); Noise : Brownian motion (BM) or independently and identically distributed (iid)

Aspect	Benchmarks						
	AS	BA	HT	WS	VP	IC	AV
Liveness/deadlock					✓		
Prob. reachability	✓	✓	✓	✓		✓	✓
Control synthesis	✓	✓				✓	✓
Min-max		✓					✓
Time horizon	F	F			I	F	F
Type of control	S	M			Dr	Dr	M
Time line	D	D	C	C	D	D	D
State space	C	H	H	H	C	C	C
Drift in ODE/SDE	pL	NL	NL	pL	NL	L	NL
Noise in SDE	Fixed	Fixed			Fixed	Fixed	Fixed
Noise: BM or i.i.d.	iid	iid			iid	iid	iid
Guards		✓	✓	✓			✓
Rate spontaneous jumps	Fixed		Fixed	Fixed		Fixed	
Size spontaneous jumps	Fixed		Fixed	Fixed		Fixed	
Environment		✓		✓			✓
Subsystems		✓	✓	✓			
Concurrency			✓	✓			
Synchronization			✓	✓			
Shared variables		✓		✓			
# discrete states		5	576	35			
# continuous variables	3	7	2	11	2	50	7
# model parameters	24	19	15	36	3	8	11

and can deliver up to one petaflop of mixed-precision performance to significantly accelerate massively-parallelized HPC applications. This machine has one NVIDIA Tesla V100 GPU with 5120 CUDA cores, and 61 GB memory. To manage the selected machines, we developed a custom web-based interface that uses the Amazon AWS API to interact with the selected three machines (start, stop, or read status), while providing a level of user-access control in form of a pairs of user names and passwords, that are distributed to all participants of the ARCH competition. The web-based interface is developed using ASP.NET, a programming language for web applications, and hosted on a separate AWS machine that is continuously running.

Independent from the AWS-based evaluation, tool developers could provide repeatability evaluation packages in form of Docker containers and Code Ocean capsules to the repeatability committee of ARCH. See https://gitlab.com/goranf/ARCH-COMP/tree/master/2020/SM for more details on the repeatabilty evaluation.

5.2 Anesthesia benchmark results

Table 3 compares the performance of the tools based on their run time and the highest maximum stochastic reach probability starting from any state in the initial safe set for the Anesthesia benchmark. Anesthesia benchmark defines a stochastic viability problem for a three-dimensional Gaussian-perturbed LTI system model.

Table 3: Run times and maximum reach probability on the Anesthesia benchmark.

Property	FAUST^2	StocHy	SReachTools	AMYTISS
Run time on CPU_2 (sec)	1484	74.6	7.87	< 1
Run time on GPU_1 (sec)	Did not run the benchmark			< 1
Maximum reach probability	≈ 0.93	$\geq 0.99 \pm 0.02$	≥ 0.99	≈ 0.99

AMYTISS has the fastest run time, while producing among the highest stochastic reach probability along with StocHy and SReachTools.

SReachTools computes a three-dimensional convex polytope, which is a strict subset of the initial safe set, from which there is an open-loop controller that can satisfy the specification. We found the ratio of the volume of the underapproximative 0.99-stochastic reach set to the volume of the safe initial set to be 0.7. This indicates that the controller synthesized by SReachTools can keep the system safe with 0.99 probability, when starting from a significantly large subset of the state space.

StocHy computes the optimal control policy which maximises the probability of satisfaction given any initial condition within the input set and provides exact guarantees on the resulting solution. For this case study, we see that StocHy can keep the system safe with 0.99 probability with a maximum abstraction error of 0.02.

5.3 Building automation benchmark results

Table 4 compares the performance of the tools based on their run time and the highest stochastic reach probability starting from any state in the initial safe set for the building automation benchmark. The benchmark defines a stochastic viability problem for a four-dimensional and seven-dimensional Gaussian-perturbed LTI system model (see Section 3.2).

Table 4: Run times on the Building Automation System.

Property	FAUST^2	StocHy	SReachTools	AMYTISS
Case 1, 4-dimensional system				
Run time on CPU_2 (sec)	2689	111.13	2.44	< 1
Run time on GPU_1 (sec)	Did not run the bencmark			< 1
Maximum reach probability	≈ 0.80	$\geq 0.99 \pm 0.05$	≥ 0.99	≈ 0.99
Case 2, 7-dimensional system				
Run time on CPU_2 (sec)	-	3910.41	1.33	2.9
Run time on GPU_1 (sec)	Did not run the bencmark			< 1
Maximum reach probability	-	$\geq 0.8 \pm 0.23$	≥ 0.99	≈ 0.8

AMYTISS has the fastest run time for Case 1, while SReachTools has the fastest run time for Case 2 on CPU_2. AMYTISS also took less than a second to run this benchmark on GPU_2.

SReachTools reports the highest maximum stochastic reach probability of 0.99 for both of the cases. It also computed a two-dimensional and one-dimensional convex, polytopic slices of the 0.8-stochastic viability set for Cases 1 and 2 respectively. The ratio of the volume of the underapproximative reach sets to the volume of the safe initial set was 0.97 (Case 1) and 0.87

(Case 2). This indicates that the controller synthesized by SReachTools can keep the system safe with 0.8 probability, when starting from a significantly large subset of the state space.

StocHy computes the control policy with the second highest probability of satisfaction for Case 1 with 0.99 probability with a maximum abstraction error of 0.05. However, for Case 2, the underlying abstraction for tight exact error bounds required finer gridding, and hence required more states and computational time.

FAUST^2 did not finish running Case 2.

5.4 Heated Tank benchmark results

Both in ARCH2018 and ARCH2019 the focus has been on the estimation of the dryout probability for Heated Tank version 4.0 [2, 1]. In addition to results from literature, the modes and HYPEG tools have been applied, as well as the framework SDCPN&IPS. Table 5 shows the P_{Dryout} estimation results obtained by the different methods; i.e. SAN&MC, Restart, modes, SDCPN&MC, HYPEG and SDCPN&IPS respectively. Of these methods, Restart, modes and SDCPN&IPS have in common of using splitting techniques in MC simulation of rare events. The importance functions that are used by Restart and modes count the number of component failures that are relevant for dry-out. The importance function that is used by SDCPN&IPS is the distance between liquid height $x_{H,t}$ and H_{Dryout}.

Table 5: P_{Dryout} for version 4.0: estimated by two methods from safety literature (SAN&MC and Restart), and by four ARCH2018/2019 methods (modes, SDCPN&MC, HYPEG and SDCPN&IPS).

Method / Measure	SAN&MC[17]	Restart[73]	modes	SDCPN&MC	HYPEG	SDCPN&IPS
Importance splitting	No	Yes	Yes	No	No	Yes
Estimated P_{Dryout}	5.1×10^{-4}	5.4×10^{-4}	5.09×10^{-4}	5.3×10^{-4}	4.84×10^{-4}	5.27×10^{-4}
Simulation effort	100,000 MC runs	-	-	100,000 MC runs	183,773 MC runs	100,000 particles
95% confidence interval	$\pm 1.4 \times 10^{-4}$	$\pm 3.4 \times 10^{-4}$	$\pm 0.26 \times 10^{-4}$	$\pm 1.4 \times 10^{-4}$	$\pm 1.0 \times 10^{-4}$	$\pm 0.24 \times 10^{-4}$

For ARCH2020 the selected objective was to apply novel tools prohver as well as Formal-verifier of ProbReach to the Heated Tank benchmark version 4.0. In contrast to the statistical methods that are shown in Table 5, these novel tools do not make use of statistical methods. Both prohver and ProbReach ran into difficulties in handling Heated Tank version 4.0. A common problem for both tools is the repeated occurrence of transitions' resets (e.g., unit failure $\rightarrow$ unit repair $\rightarrow$ unit failure $\rightarrow$ unit repair) during the benchmark time period of 500 hours ($t_{end} = 500\,\text{h}$). Although component failures are moderately rare, during the 500 hour period the tank level oscillates between the lower and upper control thresholds, and in rare cases even reach Dryout. By shortening the time period to 20 hours ($t_{end} = 20\,\text{h}$) the need to consider repeating of transition resets is much reduced. We refer to the corresponding Dryout probability as $P^{\leq 20}_{Dryout}$. In combination with some further model adaptations (which are described below), prohver was able to compute the results for $P^{\leq 20}_{Dryout}$ that are presented in Table 6. Table 6 also presents results obtained by modes via simulation on the same models for comparison.

Applying prohver to the Heated Tank benchmark. The prohver safety model checker for stochastic hybrid automata (SHA) can compute a guaranteed upper bound on the maximum

Table 6: prohver and modes estimated $P^{\leq 20}_{Dryout}$ for Heated Tank version 4.0 during 20 hours instead of the 500 hours in Table 5

Tool	prohver			modes		
	$p_I = 0.5$	$p_I = 0.\overline{3}$	$p_I = 0.25$	SMC	RES/count	RES/level
$P^{\leq 20}_{Dryout}$, race model	0.0266 upper bound 77k states 43 m	(timeout)	(timeout)	$1.95 \cdot 10^{-7}$ 95% in ±10% 4.3G runs 2 h	$1.98 \cdot 10^{-7}$ ≈ 95% in ±10% 823k×64 runs 5 min	$1.84 \cdot 10^{-7}$ ≈ 95% in ±10% 3.5M×64 runs 22 min
$P^{\leq 20}_{Dryout}$, combined model	0.0098 upper bound 4169 states 24 s	0.0044 upper bound 35k states 10 min	0.0024 upper bound 175k states 5 h	$1.93 \cdot 10^{-7}$ 95% in ±10% 4.3G runs 1 h	$1.97 \cdot 10^{-7}$ ≈ 95% in ±10% 214k×64 runs 1 min	$1.99 \cdot 10^{-7}$ ≈ 95% in ±10% 3.2M×64 runs 10 min

probability of reaching a set of (unsafe) goal states. Its input is a parallel composition of multiple *symbolic* stochastic hybrid automata, i.e. with discrete variables in guards and updates, in the Modest [33] or Jani [9] formats. prohver works in four steps (see [33, Fig. 9]):

1. Syntactically abstract continuous probability distributions into discrete distributions over intervals that cover the distribution's support. The result is a parallel composition of probabilistic hybrid automata (PHA) with discrete variables.

2. Compute the "discretely-reachable" discrete locations (by, starting from the initial state, following transitions and executing updates assuming that the continuous variables can always take any possible value) to obtain one flat PHA.

3. Remove probabilities from the PHA and use the non-stochastic PHAVer [25] reachability tool to compute the labelled transition system (LTS) of reachable concrete state sets.

4. Reintroduce the probabilities into the LTS to obtain a Markov decision process; use value iteration to compute the maximum probability to reach a goal state.

There are two sources of *over*approximation error in this process: the abstraction of continuous distributions, and the reachability computation by PHAVer in case of non-linear dynamics. For assessing Dryout probability for Heated Tank version 4.0 the latter does not apply.

The first attempt was to run prohver on a minimally updated version of the existing Modest heated tank model first used in [2] and included in [1]. It gave rise to a PHA with 5440 locations in step 2, on which PHAVer did not manage to terminate within 24 h of computation time for P_{Dryout}. For the second attempt prohver was run on a new model that pays attention to avoid any unnecessary intermediate transitions that would give rise to extra locations. For this model the SDCPN description of the heated tank benchmark 4.0 [1, Fig. 22] was closely followed. As a result, the PHA had only 82 discretely-reachable locations. This already sped up simulations using modes significantly. Using two intervals of equal probability $p_I = 0.5$ in step 1, we were able to obtain a first upper bound of 0.0266 for $P^{\leq 20}_{Dryout}$ after 43 minutes of computation time[1]. As soon as we used more intervals, or P_{Dryout}, PHAVer still timed out.

The system is always in a race between three exponentially-distributed events: either failure or repair of pump 1, pump 2, or the valve, each at a specific rate. Each of these gives rise to

[1]All experiments using modes and prohver on the Heated Tank benchmark were performed on a 64-bit Ubuntu 18.04 system with an Intel Core i7-4790 CPU (3.6-4.0 GHz, 4 physical and 8 logical cores) and 8 GB of RAM. prohver uses a single core only, while modes employs 7 parallel simulation threads on this machine.

a continuous variable representing the countdown timer until the event occurs in the PHAVer evaluation. Therefore in a third attempt, a further reduced "combined" Modest model has been identified by exploiting the special properties of the exponential distribution to replace the three concurrent events by a single one distributed exponentially with the *sum of the rates* of the individual events. This reduces the continuous dimensions from 4 (three countdown timers plus tank level) to 2. On this model, PHAVer and prohver finally manage to compute upper bounds for $P^{\leq 20}_{Dryout}$ using two, three, and four intervals as shown in Table 6. With 2 intervals, both overapproximation error and runtime are significantly lower than for the "racing" model. We see that the error decreases somewhat whereas computation time explodes as we increase the number of intervals. More intervals lead to a finer partitioning of the continuous state space, and consequently abstraction LTS (from step 3) with more states (the number of states being listed with each prohver result in Table 6). P_{Dryout}, however, still remains out of reach.

For comparison, we also ran modes using standard Monte Carlo simulation (SMC) and rare event simulation (RES) using two different importance functions for fixed-effort splitting (with 64 child runs on each importance level) to estimate $P^{\leq 20}_{Dryout}$. The results are also part of Table 6). Column "RES/count" shows the results with the importance function counting the number of stuck (failed) components, which was also used in the evaluation for Table 5; "RES/level" uses a discretisation with intervals of width up to 1 of the tank level for the importance function. modes consistently estimates the actual value to be around $1.94 \cdot 10^{-7}$ with an error of $\pm 10\,\%$ and $95\,\%$ confidence on the same hardware. Using SMC requires 4.3 billion individual simulation runs and at least one hour of computation time; rare event simulation brings significant speedup, with the "count" importance function working best here. The modes-estimated probability reflects that it is extremely rare that Dryout happens during the first 20 hours of the Heated Tank version 4.0. This confirms that the best prohver-estimated probability of $2.4 \cdot 10^{-3}$ is a significant overapproximation of $P^{\leq 20}_{Dryout}$.

5.5 Water sewage facility benchmark results

Within the competition, the water sewage facility benchmark has been evaluated by the analytical model checkers hpnmg, ProbReach and prohver and the statistical model checkers HYPEG and modes. The property φ_A has been model checked for the water sewage facility with extension A and accordingly property φ_B has been model checked for extension B. The model has been parametrized for the evaluation. From the results, a selection of parameter sets are presented in this report.

Setup For extension A, the mean of the random variable modeling the duration of heavy rain takes values between 1.5h and 4h and the capacity of the community buffer P_c is varied between 5 and 30 (in 10^6 liters). For extension B and formula φ_B, the time of failure α takes values between 1h and 5h, and the rate of rainfall modeled by transition T_{r} is parametrized between 6 and 10 (in 10^6l/h). These parameter combinations have been chosen in order to illustrate the full range of probabilities for each variant.

For hpnmg, the errors are caused by numerical methods used for multi-dimensional integration. Using the Monte Carlo Vegas integration method [49] with default settings leads to estimated errors smaller than 10^{-6}.

The requested precision (size of the probability interval) of 10^{-12} was not met by ProbReach due to a bug in the software. Nevertheless, the probability intervals (presented in the form of the intervals' mid points and the intervals' sizes) reported in Tables 7 and 8 are numerically sound, in the sense that they are guaranteed to contain the true probability value.

We configured the Monte Carlo simulation-based tools—HYPEG and modes—to sample a number of runs sufficient to obtain a confidence level of 99 % with a confidence interval half-width of 0.0025. In prohver, error is introduced by abstracting continuous distributions into discrete intervals, and by overapproximation of reachable state sets in the non-stochastic hybrid reachability computation. Since all derivatives are constant per location, the latter can actually be performed without errors. We abstracted the continuous distributions into intervals of width at most 0.0025; since the model is acyclic and only samples a continuous probability distribution once, this should lead to results at most 0.0025 *higher* than the true values.

Results Table 7 presents the results for extension A. The combination of a large mean and a small community buffer results in a very low probability of survivability, whereas a small mean and a large community buffer result in almost sure survivability. Results for extension B are summarized in Table 8. Early failure times increase the probability of survivability, whereas later failure times in combination with high rates of rainfall decrease the survivability.

The hpnmg tool and HYPEG take the original hybrid Petri net model of this benchmark as input, whereas modes, ProbReach and prohver operate on the transformed stochastic hybrid automata (SHA).

Table 9 contains the computation times required for the construction of the parametric location tree as well as for the transformation into a SHA. The resulting computation times are considerably slower for extension B than for extension A. This is due to the size of the underlying state space which directly influences the size of the resulting automaton. For extension A the SHA has 8 locations, depending on the chosen parameter setting. For extension B the resulting automata have between 32 and 35 locations.

The tools operating on the transformed SHA, hence have an advantage over the tools which operate on the hybrid Petri net model, as they do not need to construct the full state-space anymore. This could explain the difference between the computation times, especially between the two statistical model checkers HYPEG and modes: while HYPEG needs to compute the specific next states and events, modes can mainly rely on the state-space information included in the transformed automaton model. Note that, the computation times of hpnmg include the construction of the state space (as parametric location tree).

Outlook We see that, in variant A, the complement of the property we check (in the first three parameter valuations) describes a moderately rare event. Variant B shows a similar tendency. Accordingly, it is possible to formulate properties which focus on the evaluation of safety and describe rare events. Also, performing the analysis with a relative-error criterion—instead of the absolute error of 0.0025—may thus deliver more precise results. However, currently only modes supports statistical evaluation methods for relative errors; we thus plan such a comparison as future work. For sufficiently low probabilities, we may then need rare event simulation techniques—such as the importance splitting implemented in modes [7]—to achieve acceptable computation times. Note that hpnmg already delivers absolute errors that correspond to very small relative errors for the current probabilities; here, obtaining even more precise results simply requires performing the Monte Carlo integration with higher precision.

Platforms Note, that the water sewage facility has not been executed on the AWS cloud. Instead, local machines have been used. HYPEG has been executed on a machine with an Intel i5-5257U processor with 2 CPU cores running on a frequency of 2.70 GHz, and 8 GB memory. hpnmg was executed on a Intel i5-7200U processor with 2 CPU cores running on a frequency of 2.712 GHz and 16 GB memory. modes and prohver ran on a machine with an Intel i5-6600T

Table 7: Probabilities for $\varphi_A = x_{P_0} \leq 0.1 \ \mathcal{U}^{[0,30]} \ m_{P_n} = 1$, error resp. confidence interval and run time for the given parameters μ and the capacity of the community buffer P_c for the water sewage facility benchmark with extension A. Note, that the values for μ are given in hours and the capacity of the community buffer P_c is given in 10^6 liters.

Param.		Tools				
μ	P_c	hpnmg	HYPEG	modes	prohver	ProbReach
1.5	30	0.9975730 $5.56 \cdot 10^{-7}$ 0.183s	0.9972000 99% in ± 0.0025 21.526s	0.99794 99% in ± 0.0025 0.1s	1.00000 −0.0025 15.5s	0.9976793 $2.75 \cdot 10^{-5}$ 188.731s
2	25	0.9448810 $5.08 \cdot 10^{-7}$ 0.226s	0.9447855 99% in ± 0.0025 16.322s	0.94692 99% in ± 0.0025 0.2s	0.94750 −0.0025 14.5s	0.9464662 $2.75 \cdot 10^{-5}$ 173.262s
2.5	30	0.9653570 $7.2 \cdot 10^{-7}$ 0.253s	0.9668666 99% in ± 0.0025 182.707s	0.96734 99% in ± 0.0025 0.2s	0.96750 −0.0025 14.6s	0.9664448 $2.75 \cdot 10^{-5}$ 182.653s
3	10	0.0592395 $3.46 \cdot 10^{-8}$ 0.240s	0.0602694 99% in ± 0.0025 273.123s	0.06067 99% in ± 0.0025 0.2s	0.06250 −0.0025 13.7s	0.0609557 $2.75 \cdot 10^{-5}$ 184.924s
3.5	10	0.0196439 $1.36 \cdot 10^{-8}$ 0.240s	0.0217189 99% in ± 0.0025 106.062s	0.02069 99% in ± 0.0025 0.1s	0.02250 −0.0025 14.1s	0.0203402 $2.75 \cdot 10^{-5}$ 186.299s
4	5	0.0005168 $2.47 \cdot 10^{-10}$ 0.198s	0.0012000 99% in ± 0.0025 20.782s	0.00030 99% in ± 0.0025 0.1s	0.00250 −0.0025 14.2s	0.0005439 $2.75 \cdot 10^{-5}$ 176.875s

processor with 4 physical CPU cores running at a frequency of 2.7–3.5 GHz, and 16 GB memory. ProbReach was executed on a machine with a 2.2 GHz Intel i7-4702MQ processor with 8 logical CPU cores (4 physical CPU cores), and 8 GB of RAM.

5.6 Van der Pol Oscillator benchmark results

We have applied Mascot-SDS on this benchmark for solving Problem 1. An over- and under-approximation of the winning region is computed and plotted in Figure 3. Note that when the noise is treated as worst case, then there exists a deterministic value of the noise for which the oscillator trajectory never reaches the target B from all the initial states inside the domain, thus violating the specification. So the winning region is empty if the noise is treated as worst case. A trajectory with a fixed deterministic perturbation that misses the target all the time is shown in black in Figure 3. On the other hand, when the noise is treated as stochastic, then there are initial states from where the perturbed trajectory visits the target set B repeatedly. A trajectory with stochastic perturbation and the initial state I is also shown in the figure. The computation times are 417 sec for the over-approximation and 15 415 sec for the under-approximation.

Tool AMYTISS was also applied to Problem 3. It solves the problem in around 416 sec using CPU_1, 58 sec using CPU_1 and 4.7 sec using GPU_1.

Table 8: Probabilities for $\varphi_B = x_{P_0} \leq 0.01\ \mathcal{U}^{[\alpha,\alpha+30]}\ m_{P_{\text{on}}} = 1$, error resp. confidence interval and run time for the given parameters for the water sewage facility benchmark with extension B. Note that the formula has been modified to be model checked with hpnmg, prohver and modes (to $\varphi_B = x_{P_0} \leq 0.01\ \mathcal{U}^{[0,30]}\ (x_{P_{\text{time}}} > \alpha \wedge m_{P_{\text{on}}} = 1)$ with an additional continuous place/clock to model the time). Here, the values for the parameter α modelling the repair time are given in hours and the rate of the continuous transition T_r is given in 10^6 liters per hour.

Param.		Tools				
T_r	α	hpnmg	HYPEG	modes	prohver	ProbReach
6	1	0.998367 $9.56 \cdot 10^{-7}$ 1.047s	0.9988000 99% in $\pm$ 0.0025 55.806s	0.99908 99% in $\pm$ 0.0025 0.1s	1.00000 -0.0025 8.5s	0.9983796 $6.6 \cdot 10^{-3}$ 296.559s
7	5	0.959817 $6.63 \cdot 10^{-7}$ 0.782s	0.9591547 99% in $\pm$ 0.0025 554.432s	0.96061 99% in $\pm$ 0.0025 0.3s	0.96000 -0.0025 8.7s	0.9602791 $6.6 \cdot 10^{-3}$ 275.235s
8	4	0.894601 $4.73 \cdot 10^{-7}$ 0.642s	0.8955851 99% in $\pm$ 0.0025 1283.087s	0.89538 99% in $\pm$ 0.0025 0.6s	0.89500 -0.0025 7.7s	0.8948937 $6.6 \cdot 10^{-3}$ 272.095s
9	4	0.670807 $1.88 \cdot 10^{-7}$ 0.579s	0.6721797 99% in $\pm$ 0.0025 394.829s	0.67174 99% in $\pm$ 0.0025 0.9s	0.67250 -0.0025 7.5s	0.6715376 $6.6 \cdot 10^{-3}$ 339.147s
10	4	0.181269 $9.39 \cdot 10^{-9}$ 0.802s	0.1823443 99% in $\pm$ 0.0025 2253.333s	0.18367 99% in $\pm$ 0.0025 0.6s	0.18500 -0.0025 7.0s	0.1866682 $6.6 \cdot 10^{-3}$ 224.552s

Table 9: Computation times for (i) the construction of the state space (as parametric location tree), denoted as *plt*, and (ii) the transformation of the hybrid Petri net into a stochastic hybrid automaton, denoted as *transf.*, for the water sewage facility with extensions A and B.

(a) Extension A.

Parameters		computation time	
μ (h)	P_c (10^6 l)	plt	transf.
1.5	30	34ms	3ms
2	25	84ms	6ms
2.5	30	35ms	3ms
3	10	60ms	3ms
3.5	10	41ms	4ms
4	5	40ms	3ms

(b) Extension B.

Parameters		computation time	
T_r (10^6 l/h)	α (h)	plt	transf.
6	1	199ms	210ms
7	5	203ms	99ms
8	4	104ms	126ms
9	4	109ms	113ms
10	4	126ms	217ms

5.7 Integrator-chain benchmark results

Figure 4 describes the results of the scalability comparison of the participating tools on a n-dimensional integrator chain. SReachTools is the most scalable tool, with the ability to analyze 100-dimensional system in less than 5 minutes, while FAUST^2, StocHy, and AMYTISS can solve the benchmark up to dimensions 3, 10, and 20 respectively. AMYTISS based on GPU and coarse-gridding was the fastest approach for low-dimensional systems $n \leq 10$.

SReachTools, FAUST^2, and AMYTISS successfully identified a safe initial state with reach

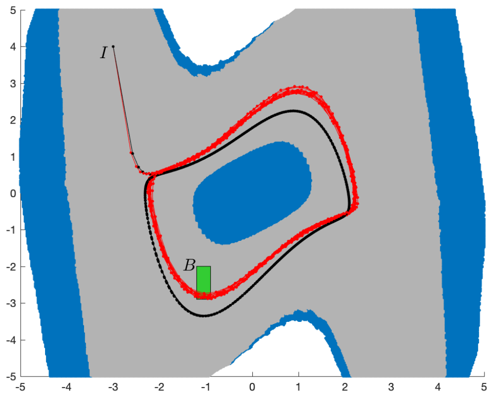

Figure 3: The Van der Pol oscillator example: The set B (green box) is the target that should be visited infinitely often. The under-approximation of the winning region is in grey. The over-approximation of the winning region is the union of blue and grey areas. I is the initial state for simulation. The trajectory with stochastic perturbation is shown in red, and the trajectory with a fixed deterministic perturbation that always misses the target is shown in black.

probability ≈ 1 for $n \leq 8$, while StocHy returned initial states with decreasing reach probability with increasing problem dimension.

SReachTools consistently generated non-trivial underapproximation of 0.8-stochastic reach set sets compared to FAUST^2 and AMYTISS even for high dimensional problems. This observation may be attributed to the fact that SReachTools is abstraction-free. We report the volumes for 2D slices with $x_3 = \ldots = x_n = 0$, to facilitate the computation of the volumes for large n.

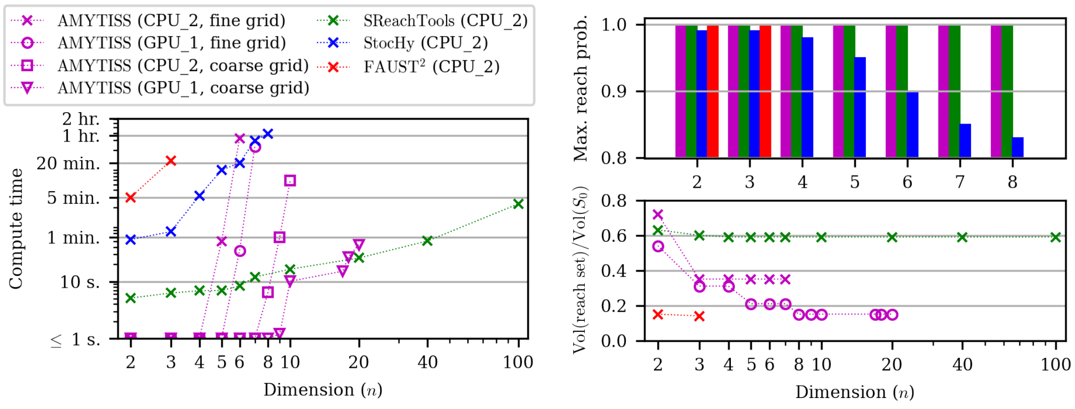

Figure 4: Integrator-chain benchmark for scalability comparison. (left) SReachTools can analyze the highest dimensional system ($n = 100$), while AMYTISS is the fastest tool for low-dimensional problems. (right, top) FAUST^2 (for $n \leq 3$), AMYTISS, and SReachTools computes maximum stochastic reach probability of ≈ 1, while StocHy returns lower maximum reach probability as dimension increases. (right, bottom) SReachTools provides non-trivial underapproximative stochastic reach sets even for high-dimensional problems.

5.8 7-Dimensional BMW 320i benchmark results

In this benchmark, we are interested in an autonomous operation of the vehicle to satisfy a reach-avoid property. In particular, the vehicle should park itself automatically in a parking lot located in the projected set $[-1.5, 0.0] \times [0.0, 1.5]$ within 32 time steps, while avoids hitting a barrier represented by the set $[-1.5, 0.0] \times [-0.5, 0.0]$. AMYTISS is the only tool presented in this report capable of handling this benchmark, and solved the problem in 825 sec. Since the dimension of the system is seven and relatively large for discretization-based techniques (the number of transitions in MDPs $|\hat{X} \times \hat{U}|$ is 3,937,500), the required memory for constructing the finite MDP would be very huge (and it is impossible in practice to construct such an abstraction due to the memory limitation). In order to handle this benchmark, we employ the *on-the-fly abstraction* technique as described in Section 2 to significantly reduce the required memory for constructing our finite MDP. We refer the interested reader to [43] for more details on the OFA technique.

6 Conclusions

The achieved results this year are manifold. Three novel benchmarks with interesting properties and challenges were added to the collection of benchmarks for stochastic hybrid models. Eleven tools were successfully evaluated on subsets of those collected benchmarks. Additionally, a first-time attempt to classify benchmarks according to relevant properties has been made with interesting results.

Based on these results we develop two types of conclusions. The first is what the tool set owners learned from the benchmarking regarding their future tool development; this is addressed in Section 6.1. The second conclusion concerns identification of possible directions for further benchmark development; this is addressed in Section 6.2.

6.1 Further tool development

While part of the future work of this group may focus on the further development of novel benchmarks, it is of primary importance to learn from the results obtained by the various tools on the current benchmarks, and to use this for the further development of the tools.

6.1.1 Further development of FAUST^2

FAUST^2 is the first tool developed for performing automated verification and synthesis of discrete-time stochastic systems with guaranteed error bounds, and is used as the basis for comparison by other tools handling the same class of models and problems. The capabilities of FAUST^2 can be extended in the following directions: i) a better tradeoff between sequential and parallel computations; ii) handling continuous-time stochastic systems; iii) handling game settings with multiple players; and iv) implementing randomised methods for computation of quantities of interest.

6.1.2 Further development of StocHy

StocHy provides exact errors/guarantees on the obtained solution and while it performed well in this category further work is needed to reduce the trade-off between the computational time to obtain the solution and the number of states generated for the underlying abstraction. We will further extend StocHy to also (i) parallelise the abstraction process as much as possible,

(ii) generate tighter bounds with smaller number of states and (iii) handle continuous-time continuous state SHS.

6.1.3 Further development of SReachTools

We will develop SReachTools further to provide abstraction-free, convex-optimization-based (underapproximative) verification of discrete-time stochastic switched systems. We will incorporate recent results in efficient, abstraction-free, forward stochastic reachability analysis of Markov jump affine systems using Fourier transformations and convex optimization [77]. We will also extend SReachTools to admit correct-by-construction interpolation of stochastic reach sets [78].

6.1.4 Further development of AMYTISS

AMYTISS provides a cutting probability threshold $\gamma \in [0, 1]$ to control how many partition elements around the mean value μ should be stored. For a given mean value μ, a covariance matrix Σ and a cutting probability threshold γ, $x \in X$ is called a PDF (probability density function) cutting point if $\gamma = \text{PDF}(x|\mu, \Sigma)$. Since Gaussian PDFs are symmetric, by repeating this cutting process dimension-wise, we end up with a set of points forming a hyper-rectangle in X, which is called the cutting region. Any partition element outside the cutting region is considered to have zero probability of being reached. Such approximation allows controlling the sparsity of the columns of the finite MDP matrix. At the moment, this feature is available only for systems with additive noises. We plan to include this feature for also multiplicative noises in a future update of AMYTISS. In addition, AMYTISS provides automated verification and synthesis for large-scale *discrete-time* stochastic systems. Providing a similar tool but in the *continuous-time* setting is under investigation as a future development.

6.1.5 Further development of modes and prohver

The modes simulator has worked very well—delivering results with good precision quickly—for both the Heated Tank and the water sewage benchmarks. To estimate the probabilities of rare events, its importance sampling techniques for rare event simulation speed up the analysis for the Heated Tank, and quick tests showed them to also work for some of the water sewage models. As future extensions, we would like to add support for non-linear continuous dynamics in a way that properly tracks any numerical errors.

For fully stochastic models like these two benchmarks, an overapproximating model checker may not be the best choice, which is clearly evident in the Heated Tank results for prohver. On the water sewage case, it worked well enough, due to the simplicity of the continuous dynamics coupled with having only a single stochastic event in every system execution. The bottleneck in prohver's analysis is the non-stochastic hybrid reachability step currently implemented by calling an external tool, PHAVer. It caused timeouts (more than 24 h of computation time) for the Heated Tank benchmark that currently prevent the analysis of realistic properties with useful overapproximation errors. Our first priority in the future development of prohver will be to replace PHAVer either by an up-to-date tool such as SpaceEx (which we however may need to modify to deliver a reachability graph with appropriate labelling to be able to reintroduce the probabilities), or by a custom implementation of hybrid reachability based on HyPro.

6.1.6 Further development of HYPEG and hpnmg

Both tools are dedicated to the efficient evaluation of hybrid Petri nets with general transitions with respect to potentially nested timed properties. As a result of the underlying model checking procedure a probability is returned which quantifies the probability that the specific property holds at a certain point in time.

The analytical tool hpnmg relies on a multidimensional representation of the underlying statespace as convex polytopes and is currently limited by the number of random variables present in the system. In the future we aim at relieving this restriction by improving the geometric representation of the convex polytopes and geometric operations thereon.

HYPEG uses discrete-event simulation for the evaluation of the same class of hybrid Petri nets and as such does not suffer from the number of random variables in the system. However, it needs to explore different evolutions of the hybrid Petri net in different simulation runs in order to compute results with a reasonably low statistical error. To further improve the computation times of HYPEG, the tool will be parallelised in the future.

6.1.7 Further development of Mascot-SDS

Despite being powerful in approximating the solution of infinite-horizon specifications, the current implementation of Mascot-SDS can handle only specifications in the form of repeated reachability and it only computes approximately the set of initial states from which the system can satisfy the specification with probability one (qualitative satisfaction). Future extension of this tool includes handling all linear temporal logic or omega-regular specifications, and performing computations for the quantitative satisfaction (computing probability of satisfaction for any initial state).

6.1.8 Further development of ProbReach

During the application of ProbReach to the version 4.0 of the Heated Tank benchmark, it ran into difficulties due to the the repeated occurrence of transitions' resets. Further extension of the tool includes adapting the computations to the case of varying number of jumps.

6.1.9 Development and implementation of new approaches

We welcome new tools that implement recently developed innovative approaches. In particular, the recent efforts on using Barrier Certificates for formal verification and synthesis of stochastic systems [40] has not been included in this edition of the ARCH competition. The notion of approximate similarity relation based on coupling stochastic processes [31, 32] is able to handle systems with partial state observations but is not used by any of the tools in this year. We also welcome tools that are developed for specific sub-classes of stochastic systems having a particular property, e.g., mixed-monotonicity [20]. It is also of interest to explore analytic approaches (as opposed to those based on simulations) on version 1.0 of the Heated Tank benchmark.

6.2 Further benchmark development

Based on Table 2, we have identified three types of benchmark objectives:

i) To synthesize control policies that increase the probability of remaining in a given safe set;

ii) To assess Probabilistic Reachability, i.e. to assess the probability for remaining in a given safe set during a finite time period; and

iii) To assess if a given model satisfies the Liveness/Deadlock property.

At this moment there are four benchmarks of type i): the Anesthesia model, the Chain of Integrators, the Building Automation System, and the BMW model. From the application of various methods it has become clear that these benchmarks are at the edge of what current methods and tools can handle. There also are various directions for further development:

- Control synthesis for infinite horizon. The problem is that for infinite-horizon safety, the probability of reaching the unsafe set will typically go to one. In order to avoid this, the optimization objective could for example be formulated in terms of the allowed probability to reach the set of unsafe states per unit of time (the latter is the way it is typically done in safety critical operations). There is room to develop such a benchmark. For other infinite-horizon specifications that may result in non-trivial satisfaction probabilities (e.g., liveness), the controlled version of the Van der Pol Oscillator is a good benchmark, but there is room for developing a more complex benchmark.
- Continuous time line. Current type i) benchmarks consider discrete time only. There is room to define a type i) stochastic hybrid system benchmark on continuous time.
- Spontaneous jumps. Currently type i) benchmarks hardly involve spontaneous jumps, e.g., sudden failures. There is room to define a type i) stochastic hybrid system benchmark on continuous time.
- Including differential algebraic equations. Currently type i) benchmarks do not include differential algebraic equations. There is room for the development of a type i) benchmark that incorporates differential algebraic equations.

At this moment there are two benchmarks of type ii): the Heated Tank and the Water Sewage models. From the application of various methods it has become clear that these benchmarks are at the edge of what current methods and tools can handle. There are various directions for further development:

- Unambiguous specification of a stochastic hybrid system. The key problem is currently each participant uses its own specification language. Transformations between these specification languages and formal mathematical models of generalised stochastic hybrid automata are needed. Because a similar problem has emerged in the recent past for (non-stochastic) hybrid benchmarks: we should learn from lessons learnt in that domain. In addition, we should take the broader spectrum of stochastic models into account. Maybe it is an idea to introduce a benchmark that can stimulate this development. Because such benchmark should include the full spectrum of behaviours of generalised stochastic hybrid automata, part of this benchmark development would be to define this full spectrum.
- Making the connection with safety risk analysis. From a mathematical perspective the probability that a stochastic system remains in the set of safe states during a given period, is the complement from the probability that a stochastic system enters the set of unsafe states. From a safety-risk perspective the common approach is to assess the latter only. Therefore it is recommended that the objective of safety risk directed type ii) benchmarks are formulated in terms of the probability of entering the set of unsafe states.
- Dependability analysis. More complex properties are used to quantify the probability that a system is dependable at a specific time or up to a point in time. These are often formulated using linear-time or branching time logics. Suitable properties should be formulated for benchmarks of type ii) and evaluated using probabilistic model checking.

- Sensitivity analysis and uncertainty quantification. Each benchmark involves multiple model parameters the setting of which may have significant influence on the assessed reach probabilities. Currently, the benchmarks of type ii) are defined in terms of assessing a point estimate of the reach probability for one or multiple sets of given model parameter values. Because the number of model parameters is typically large, this straightforward approach leads to a combinatorial explosion. To avoid the latter, advanced methods from sensitivity analysis and uncertainty quantification have to be integrated in the tools. In order to promote this type of tool development, it could be a good idea to include sensitivity analysis and/or uncertainty quantification questions in type ii) benchmarks.

- Including differential algebraic equations. Currently type ii) benchmarks do not include differential algebraic equations. There is room for the development of a type ii) benchmark that incorporates differential algebraic equations.

- Including Brownian Motion. Currently type ii) benchmarks do not include Brownian motion terms in differential equations. There is room to define a type ii) benchmark that includes Brownian motion.

- The rate of spontaneous jumps depends on the continuous-valued state. In the current type ii) benchmarks, the rates of spontaneous jumps depend on the discrete state components, but not on the continuous valued state components. It is recommended to define a type ii) benchmark for this phenomenon.

- In reality, safety critical operations are under the influence of environment and involve multiple agents that collaborate on the basis of partial observations and rely on different information patters. Even in a perfect safety-critical system this forces each of these agents to *make decisions under uncertainty*. In order to capture this effect there is room for a benchmark that include multiple agents and influences from environment uncertainties.

- Including nondeterminism. The current type ii) benchmarks are fully stochastic, i.e. all decisions are resolved in a probabilistic manner. The control actions in type i) benchmarks are fully synthesised and result in systems without any nondeterminism. When aiming to synthesise a control strategy, considering adversarial environments, modelling parameter ranges, or in case of absence of knowledge on event frequencies, we need to include continuous or discrete nondeterministic choices in our models in addition to *quantified* random uncertainties. Currently the prohver tool can verify the safety of nondeterministic and stochastic continuous-time hybrid models by upper-bounding the maximal probability to reach an unsafe state.

At this moment one benchmark is of type iii), namely the Van der Pol Oscillator. Because liveness/deadlock is a basic property, there seems to be room for novel benchmarks of this type. For models with purely discrete states, methods for the evaluation of liveness/deadlock property have been well developed, e.g. in case of Zeno-behaviour. However, for stochastic hybrid systems these challenges are of a more involved nature. Therefore, there is room for the further development of benchmarks that have as objective to assess liveness/deadlock property.

Observing the work of last years, the progress made on certain aspects within this research community is promising. The active participation makes us hope to welcome even more participants for the next years to contribute their views and expertise. We also wish to strengthen the link to other communities and especially groups within ARCH to establish valuable exchange, for instance on variations of benchmarks which can be shared. Furthermore, we are convinced that a lively exchange within the groups will improve the quality and analytical capabilities of tools and approaches.

References

[1] Alessandro Abate, Henk Blom, Nathalie Cauchi, Kurt Degiorgio, Martin Fraenzle, Ernst Moritz Hahn, Sofie Haesaert, Hao Ma, Meeko Oishi, Carina Pilch, Anne Remke, Mahmoud Salamati, Sadegh Soudjani, Birgit van Huijgevoort, and Abraham Vinod. ARCH-COMP19 category report: Stochastic modelling. In *ARCH19. 6th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 61 of *EPiC Series in Computing*, pages 62–102. EasyChair, 2019.

[2] Alessandro Abate, Henk Blom, Nathalie Cauchi, Sofie Haesaert, Arnd Hartmanns, Kendra Lesser, Meeko Oishi, Vignesh Sivaramakrishnan, Sadegh Soudjani, Cristian-Ioan Vasile, and Abraham P. Vinod. ARCH-COMP18 category report: Stochastic modelling. In *ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 54 of *EPiC Series in Computing*, pages 71–103. EasyChair, 2018.

[3] Alessandro Abate, Maria Prandini, John Lygeros, and Shankar Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11):2724–2734, 2008.

[4] Matthias Althoff. Commonroad: Vehicle models (version 2018a). Tech. rep. In *Technical University of Munich, 85748 Garching, Germany (October 2018)*, https://commonroad.in.tum.de. TUM, 2019.

[5] H.A.P. Blom, J. Krystul, G.J. Bakker, M.B. Klompstra, and B. Klein Obbink. Free flight collision risk estimation by sequential mc simulation. In C.G. Cassandras and J. Lygeros, editors, *Stochastic hybrid systems*, chapter 10, pages 249–281. Taylor & Francis/CRC Press, 2007.

[6] Henrik C. Bohnenkamp, Pedro R. D'Argenio, Holger Hermanns, and Joost-Pieter Katoen. MoDeST: A compositional modeling formalism for hard and softly timed systems. *IEEE Trans. Software Eng.*, 32(10):812–830, 2006.

[7] Carlos E. Budde, Pedro R. D'Argenio, and Arnd Hartmanns. Automated compositional importance splitting. *Sci. Comput. Program.*, 174:90–108, 2019.

[8] Carlos E. Budde, Pedro R. D'Argenio, Arnd Hartmanns, and Sean Sedwards. An efficient statistical model checker for nondeterminism and rare events. *Int. J. Softw. Tools Technol. Transf.*, 2020. to appear.

[9] Carlos E Budde, Christian Dehnert, Ernst Moritz Hahn, Arnd Hartmanns, Sebastian Junges, and Andrea Turrini. Jani: Quantitative model and tool interaction. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 151–168. Springer, 2017.

[10] M. L. Bujorianu and J. Lygeros. Toward a general theory of stochastic hybrid systems. In H.A.P. Blom and J. Lygeros, editors, *Stochastic hybrid systems*, pages 3–30. Springer, Berlin, 2006.

[11] Nathalie Cauchi. *Automatic verification of stochastic processes: certification of building automation systems.* PhD thesis, University of Oxford, 2019.

[12] Nathalie Cauchi and Alessandro Abate. Benchmarks for cyber-physical systems: A modular model library for buildings automation. In *IFAC Conference on Analysis and Design of Hybrid Systems*, 2018.

[13] Nathalie Cauchi and Alessandro Abate. StocHy: automated verification and synthesis of stochastic processes. In *25th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2019.

[14] Nathalie Cauchi, Luca Laurenti, Morteza Lahijanian, Alessandro Abate, Marta Kwiatkowska, and Luca Cardelli. Efficiency through uncertainty: Scalable formal synthesis for stochastic hybrid systems. In *22nd ACM International Conference on Hybrid Systems: Computation and Control (HSCC)*, 2019. arXiv: 1901.01576.

[15] F. Cérou, P. Del Moral, F. Legland, and P. Lezaud. Genetic genealogical models in rare event analysis. *Latin American J. of Probability and Mathematical Statistics*, 1:181–203, 2006.

[16] Lucia Cloth and Boudewijn R. Haverkort. Model checking for survivability! *Second International Conference on the Quantitative Evaluation of Systems (QEST'05)*, pages 145–154, 2005.

[17] D. Codetta-Raiteri. Modelling and simulating a benchmark on dynamic reliability as a stochastic activity network. In *23rd European Modeling and Simulation Symposium (EMSS)*, pages 545–554, 2011.

[18] M.H.A. Davis. *Markov models and optimization.* Chapman and Hall, London, 1993.

[19] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. A storm is coming: A modern probabilistic model checker. In *International Conference on Computer Aided Verification*, pages 592–600. Springer, 2017.

[20] Maxence Dutreix and Samuel Coogan. Specification-guided verification and abstraction refinement of mixed-monotone stochastic systems, 2019.

[21] M.H.C. Everdij and H.A.P. Blom. Piecewise deterministic Markov processes represented by dynamically coloured Petri nets. *Stochastics*, 77:1–29, 2005.

[22] M.H.C. Everdij and H.A.P. Blom. Bisimulation relations between automata, stochastic differential equations and Petri nets. In M. Bujorianu and M. Fisher, editors, *Workshop on Formal Methods for Aerospace (FMA), Electronic Proceedings in Theoretical Computer Science, EPTCS 20*, page 1–15, 2010.

[23] M.H.C. Everdij and H.A.P. Blom. Hybrid state Petri nets which have the analysis power of stochastic hybrid systems and the formal verification power of automata. In P. Pawlewski, editor, *Petri Nets*, chapter 12, pages 227–252. I-Tech Education and Publishing, Vienna, 2010.

[24] M.H.C. Everdij, M.B. Klompstra, H.A.P. Blom, and B. Klein Obbink. Compositional specification of a multi-agent system by stochastically and dynamically coloured Petri nets. In J. Lygeros H.A.P. Blom, editor, *Stochastic Hybrid Systems: Theory and safety critical applications*, pages 325–350. Springer, 2006.

[25] Goran Frehse. PHAVer: algorithmic verification of hybrid systems past HyTech. *Int. J. Softw. Tools Technol. Transf.*, 10(3):263–279, 2008.

[26] Hamed Ghasemieh, Anne Remke, and Boudewijn Haverkort. Analysis of a sewage treatment facility using hybrid Petri nets. In *Proceedings of the 7th International Conference on Performance Evaluation Methodologies and Tools.* ICST, 2014.

[27] Hamed Ghasemieh, Anne Remke, and Boudewijn R. Haverkort. Survivability Evaluation of Fluid Critical Infrastructures Using Hybrid Petri Nets. In *19th IEEE Pacific Rim International Symposium on Dependable Computing, PRDC 2013*, pages 152–161. IEEE, 2013.

[28] Joseph D. Gleason, Abraham P. Vinod, and Meeko M. K. Oishi. Underapproximation of reach-avoid sets for discrete-time stochastic systems via lagrangian methods. In *IEEE Conference on Decision and Control (CDC)*, pages 4283–4290, Dec 2017.

[29] Marco Gribaudo and Anne Remke. Hybrid Petri nets with general one-shot transitions. *Performance Evaluation*, 105:22–50, 2016.

[30] Sofie Haesaert, Nathalie Cauchi, and Alessandro Abate. Certified policy synthesis for general Markov decision processes: An application in building automation systems. *Performance Evaluation*, 117:75–103, 2017.

[31] Sofie Haesaert and Sadegh Soudjani. Robust dynamic programming for temporal logic control of stochastic systems. *IEEE Transactions on Automatic Control*, arXiv: abs/1811.11445, 2020.

[32] Sofie Haesaert, Sadegh Soudjani, and Alessandro Abate. Verification of general Markov decision processes by approximate similarity relations and policy refinement. *SIAM Journal on Control and Optimization*, 55(4):2333–2367, 2017.

[33] Ernst Moritz Hahn, Arnd Hartmanns, Holger Hermanns, and Joost-Pieter Katoen. A compositional modelling and analysis framework for stochastic hybrid systems. *Formal Methods Syst. Des.*,

43(2):191–232, 2013.

[34] Arnd Hartmanns and Holger Hermanns. The Modest Toolset: An integrated environment for quantitative modelling and verification. In *20th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 8413 of *Lecture Notes in Computer Science*, pages 593–598. Springer, 2014.

[35] Kyle Hsu, Rupak Majumdar, Kaushik Mallik, and Anne-Kathrin Schmuck. Multi-layered abstraction-based controller synthesis for continuous-time systems. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*, HSCC '18, page 120–129, New York, NY, USA, 2018. Association for Computing Machinery.

[36] Jannik Hüls, Henner Niehaus, and Anne Remke. Hpnmg: A C++ Tool for Model Checking Hybrid Petri Nets with General Transitions. In *12th International NASA Formal Methods Symposium, NFM 2020*. Springer, 2020.

[37] Jannik Hüls, Carina Pilch, Patricia Schinke, Joanna Delicaris, and Anne Remke. State-Space Construction of Hybrid Petri Nets with Multiple Stochastic Firings. In *16th International Conference on Quantitative Evaluation of Systems, QEST 2019*, volume 11785 of *LNCS*, pages 182–199. Springer, 2019.

[38] Jannik Hüls and Anne Remke. Model Checking HPnGs in Multiple Dimensions: Representing State Sets as Convex Polytopes. In *19th IFIP WG 6.1 International Conference on Formal Techniques for Distributed Objects, Components, and Systems, FORTE 2019*, volume 11535 of *LNCS*, pages 148–166, Cham, 2019. Springer.

[39] Jannik Hüls, Stefan Schupp, Anne Remke, and Erika Ábrahám. Analyzing Hybrid Petri nets with multiple stochastic firings using HyPro. In *11th EAI International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2017*, pages 178–185. ACM, 2018.

[40] Pushpak Jagtap, Sadegh Soudjani, and Majid Zamani. Formal synthesis of stochastic systems via control barrier certificates. *IEEE Transactions on Automatic Control*, arXiv: abs/1905.04585, 2020.

[41] M. Khaled and M. Zamani. `pFaces`: An acceleration ecosystem for symbolic control. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, New York, NY, USA, 2019. ACM.

[42] Marta Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Proc. 23rd International Conference on Computer Aided Verification (CAV'11)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[43] A. Lavaei, M. Khaled, S. Soudjani, and M. Zamani. AMYTISS: Parallelized automated controller synthesis for large-scale stochastic systems. In *Proc. 32nd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2020.

[44] A. Lavaei, S. Soudjani, and M. Zamani. From dissipativity theory to compositional construction of finite Markov decision processes. In *Proceedings of the 21st ACM International Conference on Hybrid Systems: Computation and Control*, pages 21–30, 2018.

[45] A. Lavaei, S. Soudjani, and M. Zamani. Compositional abstraction-based synthesis for networks of stochastic switched systems. *Automatica*, 114, 2020.

[46] A. Lavaei, S. Soudjani, and M. Zamani. Compositional abstraction-based synthesis of general MDPs via approximate probabilistic relations. *Nonlinear Analysis: Hybrid Systems*, 2020.

[47] A. Lavaei, S. Soudjani, and M. Zamani. Compositional abstraction of large-scale stochastic systems: A relaxed dissipativity approach. *Nonlinear Analysis: Hybrid Systems*, 36, 2020.

[48] A. Lavaei, S. Soudjani, and M. Zamani. Compositional (in)finite abstractions for large-scale interconnected stochastic systems. *IEEE Transactions on Automatic Control, DOI: 10.1109/TAC.2020.2975812*, 2020.

[49] G. Peter Lepage. A new algorithm for adaptive multidimensional integration. *Journal of Computational Physics*, 27(2):192–203, 1978.

[50] Rupak Majumdar, Kaushik Mallik, and Sadegh Soudjani. Symbolic controller synthesis for Büchi

specifications on stochastic systems. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, HSCC '20, New York, NY, USA, 2020. Association for Computing Machinery.

[51] Mathis Niehage, Carina Pilch, and Anne Remke. Simulating Hybrid Petri nets with general transitions and non-linear differential equations. In *13th EAI International Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS 2020*. ACM, 2020.

[52] Carina Pilch, Maurice Krause, Anne Remke, and Erika Ábrahám. A Transformation of Hybrid Petri Nets with Stochastic Firings into a Subclass of Stochastic Hybrid Automata. In *12th International NASA Formal Methods Symposium, NFM 2020*. Springer, 2020.

[53] Carina Pilch, Mathis Niehage, and Anne Remke. HPnGs go non-linear: Statistical dependability evaluation of battery-powered systems. In *Proceedings of the 26th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, MASCOTS 2018, pages 157–169. IEEE, 2018.

[54] Carina Pilch and Anne Remke. HYPEG: Statistical Model Checking for hybrid Petri nets: Tool Paper. In *Proceedings of the 11th EAI International Conference on Performance Evaluation Methodologies and Tools*, VALUETOOLS 2017, pages 186–191. ACM, 2017.

[55] Carina Pilch and Anne Remke. Statistical Model Checking for hybrid Petri nets with multiple general transitions. In *Proceedings of the 47th IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 475–486. IEEE, 2017.

[56] Hossein Sartipizadeh, Abraham P. Vinod, Behçet Açikmese, and Meeko Oishi. Voronoi partition-based scenario reduction for fast sampling-based stochastic reachability computation of LTI systems. In *Proceedings of the American Control Conference*, pages 37–44, 2019.

[57] Stefan Schupp, Erika Ábrahám, Ibtissem Ben Makhlouf, and Stefan Kowalewski. HyPro: A C++ Library of State Set Representations for Hybrid Systems Reachability Analysis. In Clark Barrett, Misty Davies, and Temesghen Kahsai, editors, *NASA Formal Methods*, volume 10227, pages 288–294. Springer, 2017.

[58] S.Haesaert, P. Nilsson, Cristian-Ioan Vasile, Rohan Thakker, Ali akbar Agha-mohammadi, Aaron D. Ames, and Richard M. Murray. Temporal logic control of POMDPs via label-based stochastic simulation relations. In *IFAC Conference on Analysis and Design of Hybrid Systems*, 2018.

[59] Fedor Shmarov and Paolo Zuliani. ProbReach: Verified probabilistic δ-reachability for stochastic hybrid systems. In *HSCC*, pages 134–139. ACM, 2015.

[60] Fedor Shmarov and Paolo Zuliani. Probabilistic hybrid systems verification via SMT and Monte Carlo techniques. In *HVC*, volume 10028 of *LNCS*, pages 152–168, 2016.

[61] Fedor Shmarov and Paolo Zuliani. SMT-based reasoning for uncertain hybrid domains. In *AAAI-16 Workhop on Planning for Hybrid Systems*, pages 624–630, 2016.

[62] S. Soudjani and A. Abate. Adaptive and sequential gridding procedures for the abstraction and verification of stochastic processes. *SIAM Journal on Applied Dynamical Systems*, 12(2):921–956, 2013.

[63] S. Soudjani and A. Abate. Aggregation of thermostatically controlled loads by formal abstractions. In *European Control Conference*, pages 4232–4237, Zurich, Switzerland, July 2013.

[64] S. Soudjani and A. Abate. Probabilistic reach-avoid computation for partially degenerate stochastic processes. *IEEE Transactions on Automatic Control*, 59(2):528–534, Feb 2014.

[65] S. Soudjani and A. Abate. Aggregation and control of populations of thermostatically controlled loads by formal abstractions. *IEEE Transactions on Control Systems Technology*, 23(3):975–990, 2015.

[66] S. Soudjani and A. Abate. Quantitative approximation of the probability distribution of a Markov process by formal abstractions. *Logical Methods in Computer Science*, 11(3):1–29, 2015. arXiv:1504.00039.

[67] Sadegh Soudjani and Alessandro Abate. Probabilistic invariance of mixed deterministic-stochastic dynamical systems. In *ACM Proceedings of the 15th International Conference on Hybrid Systems:*

Computation and Control, pages 207–216, Beijing, PRC, April 2012.

[68] Sadegh Soudjani, Alessandro Abate, and Rupak Majumdar. Dynamic Bayesian networks for formal verification of structured stochastic processes. *Acta Informatica*, 54(2):217–242, Mar 2017.

[69] Sadegh Soudjani, Caspar Gevaerts, and Alessandro Abate. FAUST^2: Formal Abstractions of Uncountable-STate STochastic processes. In *TACAS*, volume 15, pages 272–286, 2015.

[70] Sadegh Soudjani and Rupak Majumdar. Controller synthesis for reward collecting Markov processes in continuous space. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, HSCC '17, pages 45–54, New York, NY, USA, 2017. ACM.

[71] Sadegh Soudjani, Rupak Majumdar, and Alessandro Abate. Safety verification of continuous-space pure jump Markov processes. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 147–163. Springer, 2016.

[72] Sean Summers and John Lygeros. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem. *Automatica*, 46(12):1951–1961, 2010.

[73] P. Turati, N. Pedroni, and E. Zio. Advanced RESTART method for the estimation of the probability of failure of highly reliable hybrid dynamic systems. *Reliability Engineering & System Safety*, 154:117–126, 2016.

[74] Abraham P. Vinod, Joseph D. Gleason, and Meeko M. K. Oishi. SReachTools: A MATLAB Stochastic Reachability Toolbox. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pages 33 – 38, Montreal, Canada, April 16–18 2019.

[75] Abraham P. Vinod, Baisravan HomChaudhuri, and Meeko M. K. Oishi. Forward stochastic reachability analysis for uncontrolled linear systems using Fourier transforms. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control, HSCC*, pages 35–44, April, 2017.

[76] Abraham P. Vinod and Meeko M. K. Oishi. Scalable underapproximation for the stochastic reach-avoid problem for high-dimensional LTI systems using Fourier transforms. *IEEE Control Systems Letters*, 1(2):316–321, Oct 2017.

[77] Abraham P. Vinod and Meeko M. K. Oishi. Probabilistic occupancy via forward stochastic reachability for Markov jump affine systems. *IEEE Transactions on Automatic Control (accepted)*, 2018. Available online: https://arxiv.org/abs/1803.07180.

[78] Abraham P. Vinod and Meeko M. K. Oishi. Stochastic reachability of a target tube: Theory and computation. *Automatica (in review)*, 2018. Available online: https://arxiv.org/abs/1810.05217.

[79] Abraham P. Vinod and Meeko M. K. Oishi. Scalable underapproximative verification of stochastic LTI systems using convexity and compactness. In *Proceedings of the International Conference on Hybrid Systems: Computation and Control*, pages 1–10. ACM, April, 2018.