EPiC Series in Computing

Volume 61, 2019, Pages 62–102

ARCH19. 6th International Workshop on Applied Verification of Continuous and Hybrid Systems

ARCH-COMP19 Category Report: Stochastic Modelling

Alessandro Abate[1], Henk Blom[2], Nathalie Cauchi[1], Kurt Degiorgio[4], Martin Fränzle[5], Ernst Moritz Hahn[6], Sofie Haesaert[3], Hao Ma[2], Meeko Oishi[12], Carina Pilch[9], Anne Remke[9], Mahmoud Salamati[10], Sadegh Soudjani[7], Birgit van Huijgevoort[3], and Abraham P. Vinod[11]

[1] University of Oxford, Department of Computer Science, Oxford, UK `name.surname@cs.ox.ac.uk`
[2] Delft University of Technology, Delft, The Netherlands and National Aerospace Laboratory, Amsterdam, The Netherlands {`Henk.Blom,Hao.Ma`}`@nlr.nl`
[3] TU Eindhoven, Eindhoven, The Netherlands {`S.Haesaert, b.c.v.huijgevoort`}`@tue.nl`
[4] Diffblue Ltd, Oxford, UK `kurt.degiorgio@diffblue.com`
[5] Oldenburg University, Oldenburg, Germany `Martin.Fraenzle@Informatik.Uni-Oldenburg.De`
[6] University of Liverpool, Liverpool, UK `e.hahn@qub.ac.uk`
[7] School of Computing, Newcastle University, UK, `Sadegh.Soudjani@ncl.ac.uk`
[8] University of Münster, Germany {`carina.pilch, anne.remke`}`@uni-muenster.de`
[9] Max Planck Institute for Software Systems, Germany `msalamati@mpi-sws.org`
[10] University of New Mexico, Department of Electrical and Computer Engineering, New Mexico, USA {`oishi`}`@unm.edu`
[11] The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, Texas, USA `aby.vinod@gmail.com`

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 2019.

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 gitlab.com/goranf/ARCH-COMP.

This report summarizes results obtained in the 2019 friendly competition of the ARCH workshop[1] for the *stochastic modelling* group. In this edition, we have divided our work over two targets:

1. the generation of new benchmarks, comprising different model structures and dealing with diverse applications and tasks to be solved; and

2. the friendly competition, run over previously identified benchmarks.

We have additionally initiated a discussion about setting up a formal language for stochastic models (or a subset thereof). This is specifically seen within the heated tank benchmarks, were different formalisms are collated and key differences are identified. We will leave this as future work (see relevant final section).

The tools and frameworks used are (in alphabetical order): (ϵ, δ) Abstraction, FAUST^2, HYPEG, LyapMMC, Modest Toolset, SReachTools, StocHy, SDCPN. We emphasise in particular the introduction of four new tools and/or frameworks. 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 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. Next, in Section 4 we present the results of the friendly competition where the participating tools or algorithmic frameworks that are used to solve instances of the benchmarks from last year's edition. We identify key challenges and discuss future plans in Section 5.

2 Participating Tools & Frameworks

The tools and frameworks used in the category *Stochastic Modelling* are introduced next, organised in alphabetical order.

2.1 Tools

FAUST^2 The tool *Formal Abstractions of Uncountable-STate STochastic processes* (FAUST^2) [48] 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 [36] or STORM [17]. Alternatively, it can handle internally

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

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 [46, 43, 47, 44, 45]. The toolbox is available at `https://sourceforge.net/projects/faust2/`

HYPEG The Java-based library HYPEG [38] is a simulator for hybrid Petri nets with general transitions (HPnGs) [27] 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 [39]. 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 [37], whereas piecewise-linear continuous behavior is simulated without approximation.

LyapMMC Using the tool Lyapanov-based Markov Model Checker (LyapMMC), verification problems over continuous-time Markov chains (CTMC) and continuous-time Markov decision processes (CTMDP) can be solved efficiently([40]). The core verification problem for CTMCs and CTMDPs is time-bounded reachabilty. It can be computed by numerically solving a characteristic linear dynamical system but the procedure is computationally expensive. We take a control-theoretic approach and propose a reduction technique that finds another dynamical system of lower dimension (number of variables), such that numerically solving the reduced dynamical system provides an approximation to the solution of the original system with guaranteed error bounds.

Modest Toolset The Modest Toolset [32] supports the modelling and analysis of hybrid, real-time, distributed and stochastic systems. A modular framework centred around the stochastic hybrid automata formalism [31] and supporting the JANI specification [9], it provides a variety of input languages and analysis backends. The modelling formalism at the core of the Modest Toolset is the model of networks of stochastic hybrid automata (SHA), which combine nondeterministic choices, continuous system dynamics, stochastic decisions and timing, and real-time behaviour, including nondeterministic delays. A wide range of well-known and extensively studied formalisms in modelling and verification can be seen as special cases of SHA e.g. STA (stochastic timed automata), PTA (probabilistic timed automata) and MDP (Markov decision processes). The toolset can can be obtained from `http://www.modestchecker.net/`. For the experiments on the Heated Tank benchmark, we have used the Modest Toolset's simulator "modes" [8] and its support for rare event simulation based on importance splitting with the fixed effort method (using 64 child runs for each fixed effort run) [7].

SReachTools [52] An open-source MATLAB toolbox to tackle the problem of stochastic reachability of a target tube [55]. This problem subsumes terminal hitting-time stochastic reach-avoid and stochastic viability problems (guaranteeing safety in stochastic systems) [49, 2]. SReachTools handles discrete-time continuous-state dynamical systems that are linear and have an 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 [25, 53, 54, 56, 41]. The toolbox is available at https://unm-hscl.github.io/SReachTools/.

StocHy [12] is a software tool for the quantitative analysis of discrete-time *stochastic hybrid systems* (SHS). StocHy accepts a high-level description of stochastic models and constructs an equivalent SHS model. The tool allows to (i) simulate the SHS evolution over a given time horizon; and to automatically construct formal abstractions of the SHS- these are grounded on interval MDPs [13] and show the benefit of providing scalability and tighter error bounds than cognate approaches [11]. 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. This allows for libraries to be easily used and for extensions to be easily built. The tool is implemented in C++ and employs manipulations based on vector calculus, the use of sparse matrices, the symbolic construction of probabilistic kernels, and multi-threading. StocHy is available at www.gitlab.com/natchi92/StocHy.

2.2 Frameworks

(ϵ, δ) Abstraction Based on the papers [28, 29, 42], 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.

SDCPN modelling & Rare Event simulation Stochastically and Dynamically Coloured Petri Nets (SDCPN) [18, 19, 20, 21] have been developed in support of the compositional modelling of continuous-time Piecewise Deterministic Markov Processes [16] and Generalized Stochastic Hybrid Processes [10]. In combination with conditional and rare event MC simulation, SDCPN modelling has successfully been applied for the quantitative risk modelling and assessment of future air traffic management designs, e.g. [4]. The SDCPN framework is suitable for all versions of the Heated Tank benchmark. For rare event simulation it can conduct straightforward Monte Carlo (MC) simulation as well as the Interacting Particle System (IPS) acceleration approach [4, 5, 14] for reach probability evaluation of Generalised Stochastic Hybrid Processes.

3 New benchmarks

We present a set of new benchmarks (alphabetically ordered), in this section. The benchmarks highlight different tasks and modelling complexities that are not currently handled within the current benchmarks. For additional benchmarks please refer to last year's report [1] and to [42].

3.1 Networked automation system [26]

Figure 1 provides a schematic overview of the *networked automation system* (NAS) studied in [26, 50]. As a typical NAS, it involves networked control by programmable logic controllers (PLCs) connected to several sensors and actuators via wired and wireless networks. Its objective is to transport a workpiece from its initial position to the drilling position by means of a

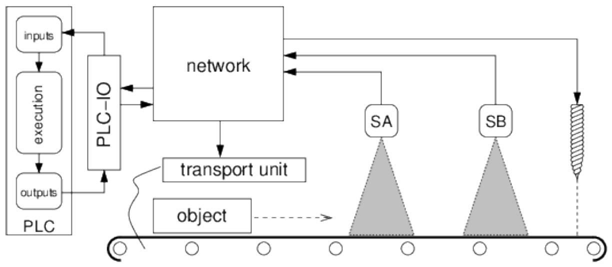


Figure 1: A networked automation system from [26].

transportation unit which controls the speed of the conveyor belt transporting the workpiece. The PLC can set the deceleration of the belt via network messages to the transportation unit, but cannot determine the position of the object unless it hits two sensors SA and SB close to the drilling position. The sensors are connected to the IO card of the PLC over the network. When the object reaches sensor SA, the PLC reacts with sending a command to the transportation unit that forces it to decelerate to slow speed. Likewise, the transportation unit is asked to decelerate to stand-still when the PLC notices that SB has been reached. The goal is that the object halts close to the drilling position despite the uncontrollable latencies in the communication network. The parameters of the system are adopted from [26] as far as indicated. Thus, one length unit (lu) is 0.01 mm, and one time step (ts) is 1 ms. The positions of SA and SB are 699 lu and 470 lu, respectively, while the acceptable drilling position are between 100 lu and 0 lu. The initial speed of the object is 24 lu/ts and the slow speed is 4 lu/ts; the decelerations for the two types of speed changes at SA and SB are 2 lu/ts^2 and 4 lu/ts^2, respectively. The network routing time is determined stochastically, needing 1 ts for delivery with probability 0.9 and 2 ts with probability 0.1. The cycle time of the PLC-IO card is 10 ts, while the cycle time of the PLC itself is 7 ts. Please note that these cycle times are not multiples of each other (in fact, even co-primal), resulting in a systematic cyclic variation of end-to-end response time of the PLC setup. The minimum sampling interval is 1 ts. Due to the initial speed of 24 lu/ts, the initial position of the object is thus equally distributed over 24 neighboring values, namely the range between 999 lu and 976 lu.

Task: Compute the probability of stopping within the region $[100\,lu, 0\,lu]$ of acceptable drilling positions.

3.2 Scheduling

Imagine a system of 16 structurally identical tasks running in sequence. Each task has a worst-case execution time (WCET) of 1 ms, with their actual run times being independent random variables featuring a triangular density, as depicted in Figure 2. Each task hence has 0.25% probability of exceeding 95% of its WCET.

Task: Compute (a conservative approximation of) the probability for the 16 tasks together exceeding 95% of their accumulated WCET of 16 ms.

3.3 Tandem Network

We introduce the *tandem network* benchmark [34] in this section. The tandem network is a queuing network system shown in Figure 3 and consists of a $M/Cox2/1$ queue composed with a $M/M/1$ queue.

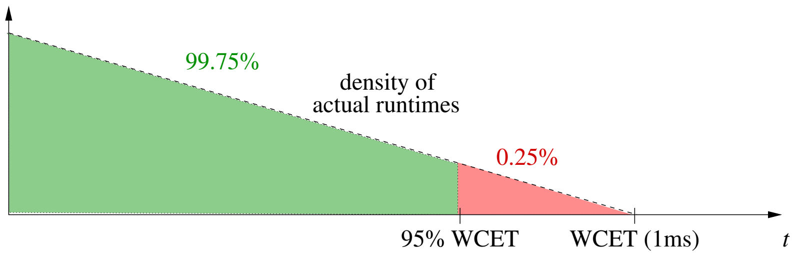


Figure 2: Runtime distribution for a single task

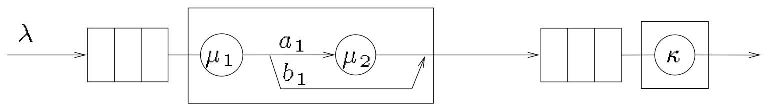


Figure 3: A typical tandem network ([34])

In Figure 3, both queuing stations have a capacity denoted by **cap**. The first queuing station has two phases for processing jobs, while the second queuing station has only one phase. In Figure 3, different phases are indicated by circles, with their corresponding rates written inside them. Jobs arrive at the first queuing station with rate λ and are processed in the first phase with rate μ_1. After this phase, jobs are passed through the second phase with probability a_1 and will be processed with the rate μ_2 before passing to the second station. Alternatively, jobs will be sent directly to the second queuing station with probability b_1. Processing in the second station is done with the rate κ.

The tandem network can be modelled as a continuous-time Markov chain (CTMC) with a state space of size determined by **cap**. Given a time bound T, we seek computing the probability of reaching to the configurations in which both stations are at their full capacity (*blocked state*) starting from a configuration in which both stations are empty (*empty state*). Therefore, *blocked state* is chosen as the **good** (absorbing) state. First, we need to model the given tandem network as a CTMC. Figure 4 demonstrates the corresponding CTMC for the case that **cap** $= 1$.

The states are of the form (l_1, p, l_2), where l_1 is the number of jobs in first station, $p \in \{0, 1, 2\}$ is the status of servicing in the first station (0 means that no service is going on, 1 means that a job is in the first phase,and 2 means that a job is in the second phase) and l_2 indicates the number of jobs in the second station. Given higher values for **cap**, one can compute the corresponding CTMC, modelling the original tandem network by applying the same procedure.

3.4 Water Sewage Treatment Plant

We provide a high level description of the sewage treatment facility in Enschede, NL based on Hybrid Petri nets [27], using the graphical components given in Figure 6a. As can be seen we have four main components: tanks, overflow tanks, pumps, and pipes. These components suffice to model the water treatment facility at a certain level of abstraction. Tanks with a certain capacity are used to store sewage during the various cleaning stages, which sometimes require additional chemicals and certain bacterias, which we however abstract from. Note that due to its finite capacity, the semantics of such a tank is that its input is reduced to match its output whenever it is full and vice versa in case it is empty. This concept is denoted *rate*

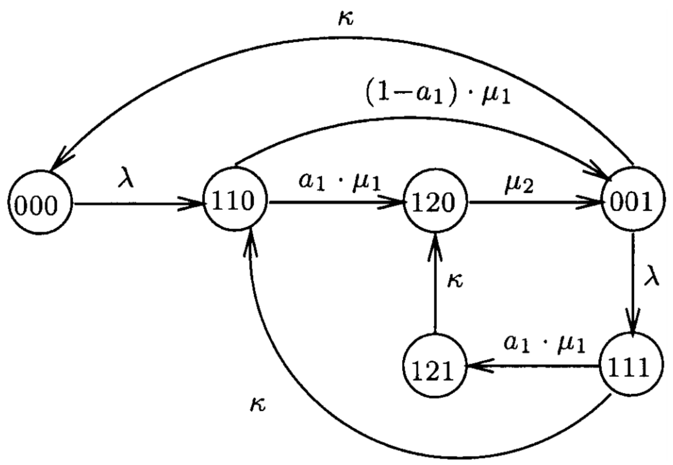


Figure 4: CTMC for a tandem network with **cap** = 1 [35]

adaptation in [27]. Pumps are used to transport sewage from one tank to another and are associated with a predefined nominal rate and can be reduced for rate adaptation purposes. Pipes connect pumps and tanks and vice versa. Note that we abstract from further parameters, like e.g. the diameter of pipes and the pressure in tanks and pipes.

This system has previously been evaluated in [24], where the model is limited in the number of stochastic variables and restricted to linear continuous dynamics without diffusion. By using models that allow for diffusion stochastic differential equations, one might be able to incorporate the cleaning steps including chemicals and bacterias into the model.

Overflow tanks are used to model *sedimentation tanks* as can be seen in the bird's-eye view of a sewage treatment facility in Figure 5. A sedimentation tank physically separates suspended solids from water using gravity. The process is such that the contaminated sewage containing solids enters the tank and stays there for a certain period of time, during which the solids sediment to the ground. At a certain rate the so-called sludge is removed from the bottom of the tank, while at the top of the tank the cleaned water overflows. Overflow tanks, depicted separately in Figure 6b, have a certain capacity, one input and two outputs, where one corresponds to taking out the settled sludge and the other (indicated by a grey rectangle) models the overflow. The rate of the input and the sludge output are determined by the connecting pumps, while the output of the overflow is determined by the difference of the output and the input.

With these components, we can describe the simplified structure of the water treatment facility, as depicted in Figure 7. Volumes of tanks are indicated in 1000 m^3 and pump rates in 1000 m^3/h. The capacity of the community sewerage system is modelled by an overflow tank denoted P_c, which has input rates that depend on the weather conditions. From this tank the water is pumped into the treatment facility with a maximum rate of 12 and in case the input exceeds the capacity of the place and the intake of the treatment facility, the waste water flows into the (o̲verflow) place P_o which models the amount of water in the streets. The primary stage of the sewage treatment consists of two phases, namely the sand interceptor and the primary sedimentation tank. The first, as the naming suggests, is responsible for filtering solids like sand from the water. Then the sewage flows into a large tank, where the sludge settles and where the lighter material, like oils, rise to the surface and is removed, and the remaining cleaner water overflows. The speed of the sand interceptor is abstracted through pump T_z, and the primary sedimentation tank is modelled by the overflow tank P_{ps}.

While the dirt settles at the ground, cleaned water is forwarded to the second cleaning stage. This stage consists of several phases for removing chemical and biological contaminations, modelled by a sequence of pumps and tanks, before a second sedimentation tank separates the biological material from the now environment friendly sewage water, that can safely be disposed to surface water. The second sedimentation tank is modelled by overflow place P_{ss}. The sludge that settles at the primary and secondary sedimentation tank is accumulated and forwarded to the sludge treatment stage. There it is thickened to reduce its volume for easier off-site transport. The sludge from the primary tank is pumped out and forwarded to the fresh sludge thickener. This is also modelled by an overflow place, denoted P_{ft}. Sludge is pumped out of the tank with a small rate and discharged to the digestion tank which is considered a very large tank. The overflow is directed to the filtrate basement. The same procedure is repeated for the accumulated sludge in the second sedimentation tank. Note that we do not consider the digestion process and the transportation of the remaining material off site.

The model presented in Figure 7, is an abstraction of the real system, which consists of three parallel sand interceptor pumps and three parallel cleaning streets with several parallel primary and secondary sedimentation tanks. Moreover, fresh and surplus sludge treatment lines also consist of several parallel pumps, which are all abstracted into a single pump. Tables 1-7 provide the parameters for the entire system, where the capacity of the community buffer, the filtrate basement and the digestion tank is provided in Table 1 and the rates for the Sand interceptor pumps is given in Table 2. The parameters for the three different cleaning streets are provided in Tables 3-5 and Tables 6 and 7 depict the values for fresh sludge treatment and surplus sludge treatment, respectively.

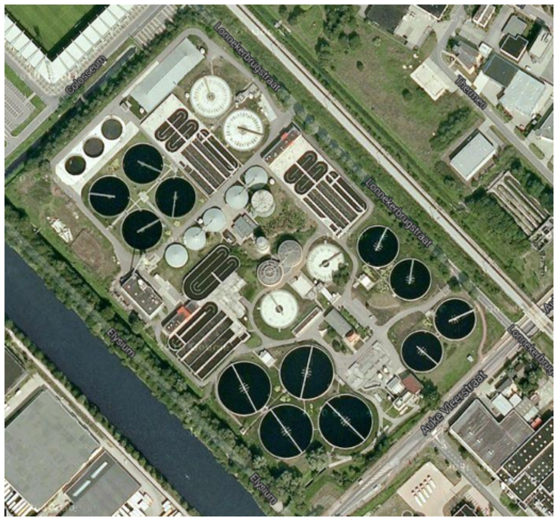
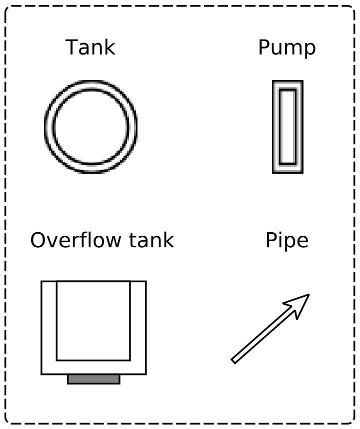
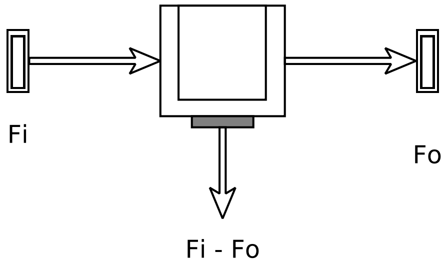


Figure 5: A bird's eye view picture of the sewage treatment facility in Enschede, the Netherlands. The picture is retrieved using Google Maps.



(a) Primitives used in the high level description.



(b) An overflow tank connected with two pumps.

Figure 6: Graphical representation.

Name	Symbol	Capacity
Community buffer	P_c	20
Filtrate basement	P_{fb}	no limit
Digestion Tank	P_{dt}	no limit

Table 1: Capacity of main tanks in the system.

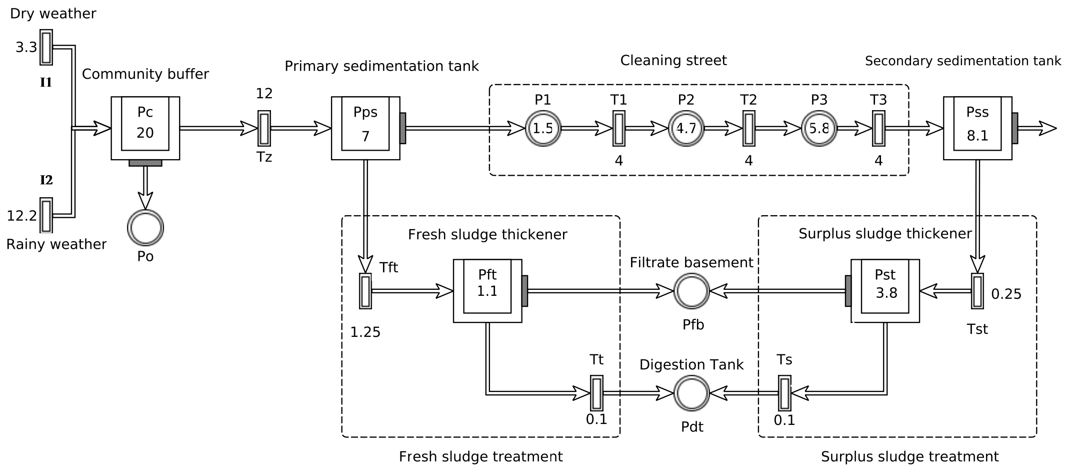


Figure 7: High level description of the water treatment facility in Enschede.

Name	Symbol	Capacity
Sand interceptor pump 1	T_z	4.7
Sand interceptor pump 2	T_z	4.7
Sand interceptor pump 3	T_z	2.6

Table 2: Rates of three different parallel sand interceptors. These are abstracted as a single pump in Figure 7.

Name	Symbol	Capacity
Primary sedimentation tank 1	P_{ps}	3.5
Primary sedimentation tank 2	P_{ps}	3.5
Selector	P_1	1.57
Anaerobic tank	P_2	4.7
Aeration tank	P_3	5.8
Transfer rate	T_1, T_2, T_3	4
Secondary sedimentation tank 1	P_{ss}	
Secondary sedimentation tank 2	P_{ss}	
Secondary sedimentation tank 3	P_{ss}	
Secondary sedimentation tank 4	P_{ss}	

Table 3: Cleaning street 1. This is the one considered in Figure 7, with two Primary sedimentation tanks merged in one. Note that the capacity of the secondary sedimentation tanks is not known to us.

Name	Symbol	Capacity
Primary sedimentation tank	P_{ps}	2.3
Selector	P_1	0.78
Anaerobic tank	P_2	1.15
Aeration tank	P_3	2.34
Transfer rate	T_1, T_2, T_3	4
Secondary sedimentation tank 1	P_{ss}	2.3
Secondary sedimentation tank 2	P_{ss}	2.3
Secondary sedimentation tank 3	P_{ss}	2.3
Secondary sedimentation tank 4	P_{ss}	2.3

Table 4: Cleaning street 2. Not considered in Figure 7.

3.4.1 What to analyse?

We have created two scenarios of interest for this model, based on the information provided by the crew responsible for maintenance of this specific water treatment facility. The first scenario relates to the increased input of sewage, during heavy rain fall. As mentioned earlier, in the Netherlands there is a contract between water treatment facilities and the municipality about the intake capacity of the facility. Hence if the weather is such that the sewage input rate exceeds this capacity the surrounding area may be flooded with sewage. Dry and rainy weather are modelled with pumps I_1 and I_2, respectively. As can be seen the rate of the I_2 is slightly more than the bottleneck of the system, i.e., rate of sand interceptor, T_z, so we may expect that eventually the community buffer, P_c overflows. The overflow from the community buffer,

Name	Symbol	Capacity
Primary sedimentation tank	P_{ps}	2.3
Selector	P_1	0.78
Anaerobic tank	P_2	11.7
Aeration tank	P_3	2.34
Transfer rate	T_1, T_2, T_3	4
Secondary sedimentation tank 1	P_{ss}	2.3
Secondary sedimentation tank 2	P_{ss}	2.3
Secondary sedimentation tank 3	P_{ss}	2.3
Secondary sedimentation tank 4	P_{ss}	2.3

Table 5: Cleaning street 3. Not considered in Figure 7.

Name	Symbol	Capacity
Primary sludge pump 1	T_{ft}	0.3
Primary sludge pump 2	T_{ft}	0.3
Primary sludge pump 3	T_{ft}	0.3
Primary sludge pump 4	T_{ft}	0.35
Fresh sludge thickener	P_{ft}	1.1
Thickened primary sludge pump 1	T_t	0.03
Thickened primary sludge pump 2	T_t	0.03
Thickened primary sludge pump 3	T_t	0.03

Table 6: Fresh sludge treatment

Name	Symbol	Capacity
Surplus sludge pump 1	T_{st}	0.055
Surplus sludge pump 2	T_{st}	0.055
Surplus sludge pump 3	T_{st}	0.035
Surplus sludge pump 4	T_{st}	0.035
Surplus sludge pump 5	T_{st}	0.035
Surplus sludge pump 6	T_{st}	0.035
Surplus sludge thickener	P_{st}	3.8
Scum pump 1	T_t	0.03
Scum pump 2	T_t	0.03
Scum pump 3	T_t	0.03

Table 7: Surplus sludge treatment

is stored in the tank P_o which models the amount of sewage in the street. Depending on the modelling tool which is used for the analysis of this system, one can assume a stochastic variable for the duration of rain, based on available data, and evaluate how much water is spilled in the street.

For the second scenario we propose to analyse a failure of the sand interceptor, T_z. According to the maintenance crew, this is the most vulnerable part of their system, and if there is failure in the system at this point no intake is possible anymore for the entire system. One can analyse for example how quickly the sand interceptor needs to be repaired, in the presence of failure, before sewage spills into the street.

Regarding the first scenario, we have presented an analysis of how long it may continue to rain without having water in the streets in [24]. Using the logic STL [23] we formalise a property stating that the amount of water in the streets is very low until the rain stops, i.e.,

$$\Phi_A = (x_{P_o} < \epsilon)\ \mathcal{U}^{[0,30]}\ (m_{P_n} = 1), \tag{1}$$

where $(x_{P_o} < \epsilon)$ states that the amount of overflow in Place P_o is smaller or equal to some ϵ, which we have chosen for computational purposes close to zero, and $m_{P_n} = 1$ means that the rain has stopped and we are back to normal operations. Formula Φ_A is satisfied if and only if $(x_{P_o} < \epsilon)$ (the safety condition) holds until we reach $m_{P_n} = 1$ (recovery condition), before the given time bound. We have chosen a time bound of 30, which is considered to be large enough for this kind of analysis.

In the second scenario, we parametrise the failure of the sand interceptor to time α. After the occurrence of a failure, a repair crew will repair the pump with a duration distributed according to an exponential distribution, with mean 2 hours. For this case we investigated in [24] a similar formula, stating that the sand interceptor should be repaired before the street is flooded:

$$\Phi_B = (x_{P_o} < 0.01)\ \mathcal{U}^{[\alpha,\alpha+30]}\ (m_{P_r} = 1), \tag{2}$$

where, $m_{P_r} = 1$, means that the sand interceptor pump is repaired. Here, we have chosen the time bound $[\alpha, \alpha + 30]$ for the Until operator, since the pump is supposed to be repaired within 30 hours after its failure. For an extensive set of results on both scenarios, we refer to [24].

4 Friendly Competition - Setup and Outcomes

4.1 Anaesthesia Model

We consider the problem of providing probabilistic guarantees of safety for the automated anaesthesia delivery problem with a human (anaesthesiologist) in the loop [22, 6, 1]. We use the well-studied multi-compartment model for delivery of Propofol (anaesthetic) in paediatrics. The depth of hypnosis can be described by the following linear dynamics

$$x[k+1] = Ax[k] + B(u[k] + \sigma[k]) + w[k] \tag{3}$$

with state $x[\cdot] \in \mathbb{R}^3$, automation input $u[k] \in [0, 7] \subset \mathbb{R}$, anaesthesiologist (human) input $\sigma[k] \in \{0, 30\}$, and input uncertainty $w[k] \sim \mathcal{N}(0, 5)$. See [1] for the matrices A and B determined for a 11-year old child weighing 35 kilograms from the Paedfusor dataset [3].

In [1], we discussed a model for the human's action (i.e., the delivery (or not) of a bolus dose of anaesthetic), as a non-deterministic, discrete-valued, discrete-time stochastic process that depends on the current state of the system, as well as the past actions of the human in a predetermined interval. Closing the loop with human control in (3) using this stochastic map results in a discrete-time stochastic hybrid system formulation. We consider the safety problem posed in [1].

Problem 4.1.1. ***(Stochastic viability problem)*** *Consider a safe set*

$$\mathcal{S} = \{z \in \mathcal{X} : 1 \leq z_1 \leq 6, 0 \leq z_2 \leq 10, 0 \leq z_3 \leq 10\}.$$

Compute the set of initial states from which the probability of guaranteeing $x[k] \in \mathcal{S}$ for a time horizon of 10 time steps is above 0.99. Synthesize an admissible automation controller that achieves this probability.

4.1.1 Stochastic viability with input uncertainty, but no anaesthesiologist

Figure 8 and Table 8 discuss the results for the stochastic viability computation with no anesthesiologist. In this case, the safety problem simplifies to stochastic viability computation of a Gaussian-perturbed linear time-invariant (LTI) system with a convex safe set. In Table 8 we have distinguished the calculation of a lower bound on the probability of verifying the formula from the computation of the abstraction error, since some techniques are statistical in nature and cannot assess the latter.

SReachTools: We use the chance-constrained and Lagrangian (set-based) approaches available in SReachTools for this problem. SReachTools does not use any abstraction, and analyzes the continuous-state problem directly.

In the chance-constrained approach, the convexity of the stochastic viability set is utilized to construct a polytopic innerapproximation via ray shooting [56]. For the easy of computation of the rays, we fixed $x_3[0]$ as it has slow dynamics.

For the Lagrangian approach, we use the notion of disturbance-minimal safe set. Recall that the computation of disturbance-minimal safe set (set of initial states from which a controller exists that is safe *despite* all disturbances) can be obtained using computational geometry [25]. Using the stochasticity of the disturbance, we compute a subset of the disturbance, which is guaranteed to provide an innerapproximation of the stochastic viability set. This approach does not require fixing $x_3[0]$.

StocHy: For this benchmark model, both abstraction techniques found within StocHy can be directly used with no need for tailoring whatsoever. With the first abstraction technique (leveraging an MDP), the LTI is abstracted into a Markov decision process by gridding the state-space depending on the required abstraction error and the underlying Lipschitz constants (h_s) of the continuous transition kernels. (This abstraction technique corresponds to the methods used in FAUST[2].) For the second technique (leveraging an IMDP and native to StocHy), the LTI is abstracted into an Interval Markov decision process. In this case, the error between the original system and the abstraction is embedded within the model structure resulting in tighter error bounds when compared to the MDP method. For IMDP, the abstraction error is defined as the $\epsilon = \hat{p}(q) - \check{p}(q)$, where $\hat{p}(q)$ and $\check{p}(q)$ correspond to the upper and lower probability bounds for each state in the IMDP. In contrast for the MDP method, the N-step abstraction error corresponds to $\epsilon = h_s \delta \mathcal{L}(A)$ where δ is the max diameter of partition and $\mathcal{L}(A)$ is the volume of set.

We perform the analysis using both abstraction methods, and

- fixing $x_3[0] = 5$,
- maintaining all three continuous variables.

When abstracting the model using the MDP method, the control action space is discretised into three values, $u = 0$, $u = 3.5$, $u = 7$. Whereas, due to the little effect the control actions have over the $N = 10$ time horizon, for the IMDP we fix $u = 7$. Also, note that when gridding the state-space to generate the IMDP for the three dimensional model, x_1 and x_2 where coarsely gridded due the slow dynamics of x_3 which was seen to have small effect on the satisfaction of the safety property.

Backwards reachability: Due to the small noise signal, gridding might not be necessary. This conclusion is drawn based on the same reasoning and computations as described in Section B.0.1. Therefore, the noise is truncated, where the error, denoted with δ, is chosen equal to $\delta = 0.0001$. This yields the Gaussian realizations w_1, w_2, w_3 to be limited to $w_1 \in [-0.1312, 0.1312], w_2 \in [-0.1312, 0.1312], w_3 \in [-0.1312, 0.1312]$. The new abstract model

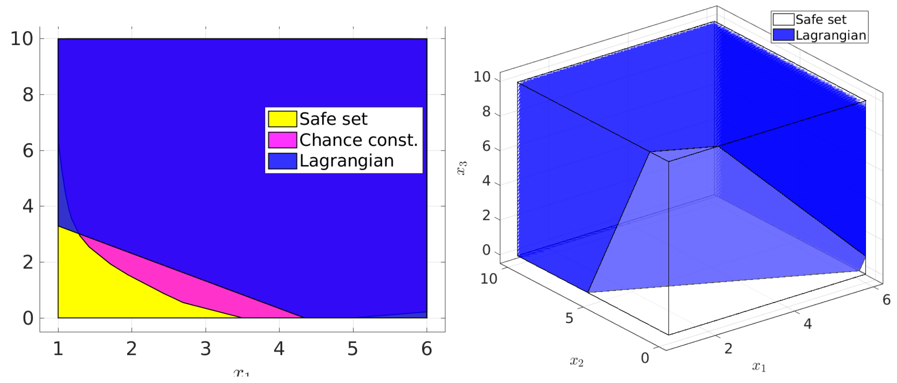


Figure 8: Stochastic viability set (initial states with safety probability of at least $\alpha = 0.99$). The plot on the left fixes $x_3[0] = 5$, and the one on the right shows the Lagrangian (set-based) underapproximation for the entire state space, obtained using SReachTools. Compute times are given in Table 8. Code available at `https://doi.org/10.24433/CO.3325937.v1`.

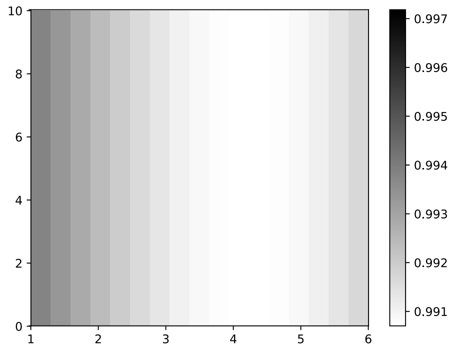


Figure 9: The lower probability of satisfying safety property when $x_3[0] = 5$ is fixed and policy is generated using StocHy via abstractions into IMDP. The associated computational times are given in Table 8.

$\tilde{M}$ with truncated noise, is approximately bisimilar to the original model M. This is denoted as $\tilde{M} \equiv^{\epsilon=0}_{\delta=1.00e-04} M$. Next, a backwards reachability computation is performed using the Multi-Parametric toolbox [33] in Matlab. After every step, the 3-dimensional polyhedron noise is subtracted. The resulting intitial set for which you will stay within the safe set is given in Figure 10. Due to the trunaction of the noise, the probability of staying in the safe set with this initial set equals $(1 - \delta)^{10} = 0.999$.

4.1.2 Stochastic viability with anaesthesiologist, but no uncertainty

Consider inputs $(v[k], \sigma[k]) \in \{0, 7\} \times \{0, 30\}$, respectively, enumerated as a, b, c, and d. When applied as a constant input, for the $(0, 0)$ input (a) the systems behaviour becomes autonomous. Similarly for the $(7, 0)$ input (b), the system is only affected by the control input and not by the anaesthesiologist. In contrast, inputs $(7, 30)$ c and $(0, 30)$ d reflect the (unrealistic) cases where the anaesthesiologist continuously applies a bolus dosage. Neglecting the low noise disturbance $w_k = 0$, we perform a preliminary analysis which yields the forward reach sets of the in Figures

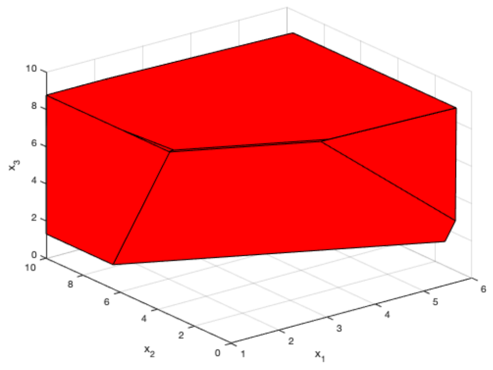


Figure 10: Stochastic viability set (with probability 0.999) computed using $\epsilon - \delta$ abstraction.

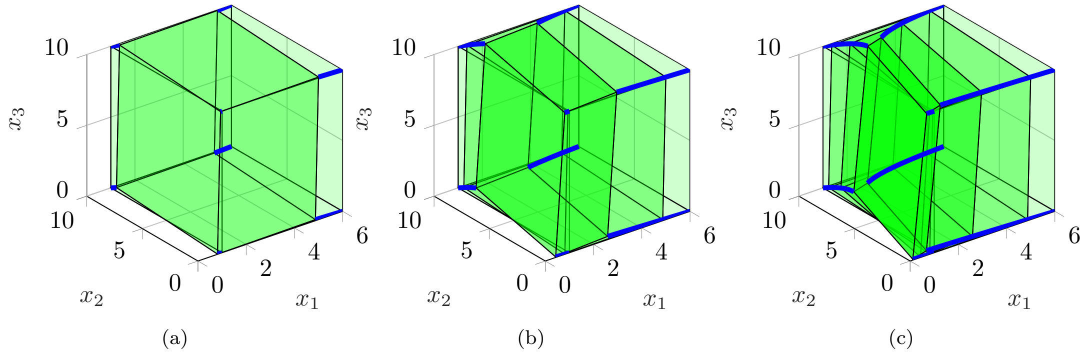


Figure 11: Forward reach sets for input a are given for $k \in \{0, 1\}$ in (a), $k \in \{0, 1, 4\}$ in (b), and for $k \in \{0, 1, 4, 10\}$ in (c).

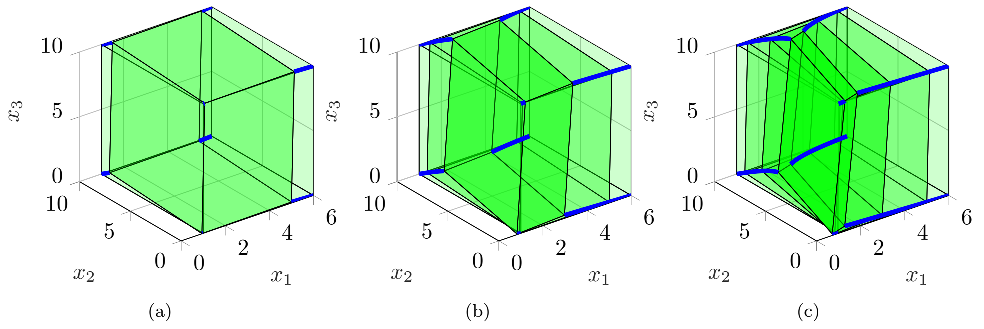


Figure 12: Forward reach sets for input b are given for $k \in \{0, 1\}$ in (a), $k \in \{0, 1, 4\}$ in (b), and for $k \in \{0, 1, 4, 10\}$ in (c).

instance	Problem 4.1.1 with no anaesthesiologist and fixed $x_3[0] = 5$				
tool	method	time [s]	Lower bound on max. safety prob.	abstraction error	Programming lang.
FAUST^2	Uniform	1596.93	1	1	MATLAB
SREACHTOOLS	Chance const.	33.93	$\geq$0.99	-	MATLAB
STOCHY	MDP	422.92	1	1	C++
STOCHY	IMDP	27.86	$\geq$ 0.991	0.008	C++

instance	Problem 4.1.1 with no anaesthesiologist and $0 \leq x_3 \leq 10$				
FAUST^2	Uniform	18147.00	1	1	MATLAB
SREACHTOOLS	Lagrangian	0.42	$\geq$0.99	-	MATLAB
STOCHY	IMDP	85.80	$\geq$0.994	0.02	C++
(ϵ, δ)-abstract.	Truncate noise	0.278	0.999	$1e^{-3}$	MATLAB

Table 8: Results for stochastic viability problem with no anaesthesiologist.

11, ??, ??, and ??. More precisely, each figure depicts the forward reach sets X_k (green boxes) starting from the safe set, that is, $x_0 \in X_0 =: S$ for one of the possible input combinations. The blue line gives trajectories starting from the vertices of the safe set. From Figure 11, we observe that most initial states stay in the safe sets, only states in the neighborhood of $(x_{0,1}, x_{0,2}, x_{0,3}) = (1, 0, 0)$ will leave the safe set at $k = 1$. Also for input b (cf. Figure ??) most states remain safe except for initial states in the neighborhood of $(x_{0,1}, x_{0,2}, x_{0,3}) = (1, 0, 0)$. Thus in this neighborhood we observe that the maximal control input is not sufficient to avoid failure if no bolus is applied. This scenario has a low probability.
For input c (cf. Figure ??), most states remain safe except for states in the neighborhood $(x_{0,1}, x_{0,2}, x_{0,3}) = (6, 10, 10)$ which leave the safe set at $k = 1$.
In contrast, when the bolus is not given together with the control input v, the states remain in the safe set. as illustrated for input d in Figure ??.

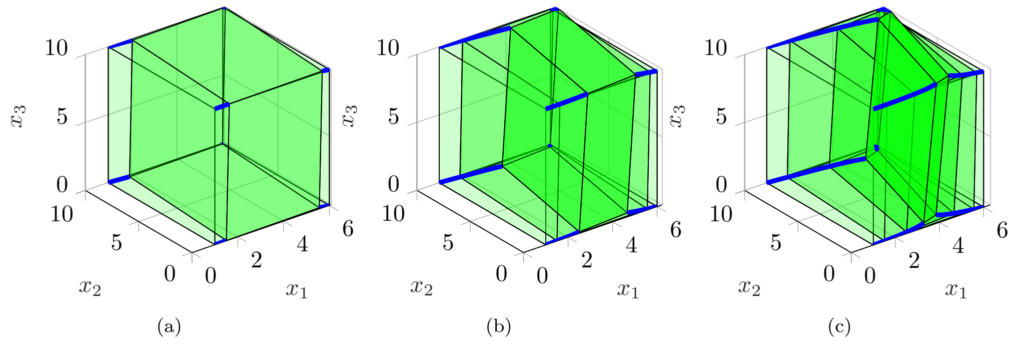


Figure 13: Forward reach sets for input c are given for $k \in \{0, 1\}$ in (a), $k \in \{0, 1, 4\}$ in (b), and for $k \in \{0, 1, 4, 10\}$ in (c).

Based on this preliminary analysis, a policy that achieves safety can be composed as

$$v[k] = \begin{cases} 7, & 1 \leq x_1[k] \leq 3 \\ 0, & 3 < x_1[k] \leq 6. \end{cases} \tag{4}$$

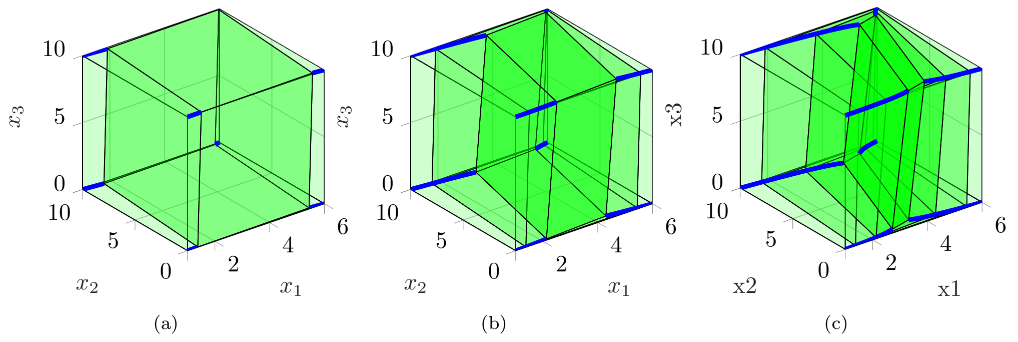


Figure 14: Forward reach sets for input d are given for $k \in \{0, 1\}$ in (a), $k \in \{0, 1, 4\}$ in (b), and for $k \in \{0, 1, 4, 10\}$ in (c).

Anaesthesiologist's actions depends on the past actions and current state The past history of the bolus dosages can be modelled with 46 states and transitions given in Table 9.

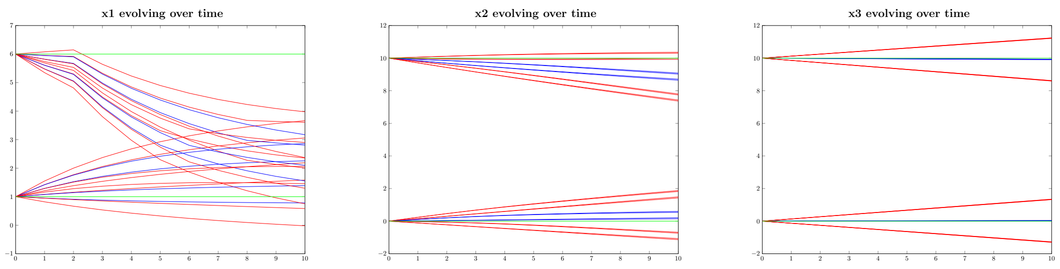
	x	1	2	3	4	5	6	7	8	9
x	↓↺									
1	↓↘									
2	↓↘	↘								
3	↓↘	↘	↘							
4	↓↘	↘	↘	↘						
5	↓↘	↘	↘	↘	↘					
6	↓↘	↘	↘	↘	↘	↘				
7	↓↘	↘	↘	↘	↘	↘	↘			
8	↓↘	↘	↘	↘	↘	↘	↘	↘		
9	$(x, x/1)$	$(2, x)$	$(3, x)$	$(4, x)$	$(5, x)$	$(6, x)$	$(7, x)$	$(8, x)$	$(9, x)$	

Table 9: Bolus counter

4.1.3 Stochastic viability with anaesthesiologist *and* uncertainty - preliminary analysis

The analysis as discussed in the previous subsection can be extended by adding the noise signal $\mathbf{w}[k]$ to the system. Due to the small noise signal, gridding might not be necessary. This conclusion is drawn based on the same reasoning and computations as described in Section B.0.1. Therefore, the noise is truncated, where the error, denoted with δ, is chosen equal to $\delta = 0.0001$. This yields the Gaussian realizations w_1, w_2, w_3 to be limited to $w_1 \in [-0.1312, 0.1312], w_2 \in [-0.1312, 0.1312], w_3 \in [-0.1312, 0.1312]$. The new abstract model $\tilde{M}$ with truncated noise, is approximately bisimilar to the original model M. This is denoted as $\tilde{M} \equiv^{\epsilon=0}_{\delta=1.00e-04} M$.

Next, we consider the worst-case scenarios. Since the input has the biggest impact on the first state x_1, we focus on this state. If state x_1 is high $3 < x_1 \leq 6$, the worst case scenario is the anaesthesiologist giving two boluses after each other (at $k = 1, 2$). This could imply that we leave the safe set by exceeding the upper-bound on x_1. The best thing we can do with our automated input is to supply no additional input. On the other hand, if state x_1 is high $1 \leq x_1 \leq 3$, the worst case scenario is the anaesthesiologist doing nothing at all. Even tough this has a very low probability, it could happen. This could imply that we leave the safe set by exceeding the lower-bound on x_1. The best thing we can do with our automated input is to supply as much as possible. This yields the same control strategy as given in (4). The results are given in Figure 15. The green lines denote the upper- and lowerbound of the safe set, the red lines are the extrema of the noise and the blue lines the nominal trajectory. The trajectories are shown for all extreme initial conditions inside the safe set. Besides that, the anaesthesiologist has been implemented as described by the worst case scenarios before. The states are not always safe and due to the coupling between the states, it is not possible to conclude anything about the initial set from which this input will guarantee that the original system will always be in the safe set (with a probability of $(1-\delta)^{10} = 0.999$). The analysis shows that it is necessary to have more advanced tools to solve this problem.



(a) (b) (c)

Figure 15: Evolution of the state over time, x_1 in (a), x_2 in (b), x_3 in (c)

4.2 Building Automation System

The Building Automation System (BAS) benchmark is split into three different versions built upon the library of stochastic models presented in [11, 1]. For each model instance (i) we establish the dynamics of the models, (ii) the specification of interest, and (iii) we describe the results of the friendly competition.

4.2.1 CS1BAS

Model This model is a two zone thermal model consisting of a single discrete location and four continuous variables which evolve according to a stochastic difference equation which takes the form of

$$\mathbf{M}_s : \begin{cases} x[k+1] &= Ax[k] + Bu[k] + Q + \Sigma W[k] \\ y_s[k] &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x[k], \end{cases} \tag{5}$$

with a uniform sampling time $\Delta = 15$ minutes. Here,

$$A = \begin{pmatrix} 0.6682 & 0 & 0.02632 & 0 \\ 0 & 0.6830 & 0 & 0.02096 \\ 1.0005 & 0 & -0.000499 & 0 \\ 0 & 0.8004 & 0 & 0.1996 \end{pmatrix}, B = \begin{pmatrix} 0.1320 \\ 0.1402 \\ 0 \\ 0 \end{pmatrix}, Q = \begin{pmatrix} 3.3378 \\ 2.9272 \\ 13.0207 \\ 10.4166 \end{pmatrix}, \Sigma = \begin{pmatrix} 0.0774 & 0 & 0 & 0 \\ 0 & 0.0774 & 0 & 0 \\ 0 & 0 & 0.3872 & 0 \\ 0 & 0 & 0 & 0.3098 \end{pmatrix},$$

and $W = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \end{bmatrix}^T$ are independent Gaussian random variables, which are also independent of the initial condition of the process (cf. Section 2.2.1 in [1]).

Specification The thermal model is performing within the comfort range if 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.

Problem 4.2.1. *Define* $\mathcal{S}_1 = [19.5\ 20.5] \times [19.5\ 20.5]$. *Characterize the set of initial states and an admissible controller such that the probability of the corresponding traces* $(y_s[\cdot])$ *generated by* $\boldsymbol{M}_s$ *remain within* $\mathcal{S}$ *for 1.5 hours (i.e. 6 time steps), with a minimum likelihood of* 0.8.

4.2.2 CS2BAS

Model The second case studies a model similar to Eq. (5) in Section 4.2.1, but as a higher dimensional problem for more model fidelity. In this case, 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 trace $y_s[\cdot]$ in this case is defined as the first component of the state. The model details can be found in [1].

Specification We would like to synthesise a policy ensuring that the temperature within zone 1 does not deviate from the set point by more then 0.5^oC over a time horizon equal to 1.5 hours.

Problem 4.2.2. *Define* $\mathcal{S}_2 = [19.5\ 20.5]$. *Characterize the set of initial states and an admissible controller such that the probability of the corresponding traces* $(y_s[\cdot])$ *generated by* $\boldsymbol{M}_s$ *remain within* $\mathcal{S}$ *for 1.5 hours (i.e. 6 time steps), with a minimum likelihood of* 0.8.

Model manipulations Using a singular value decomposition, the rank deficient $B_w \in \mathbb{R}^{7\times 6}$ matrix is replaced by an equivalent matrix $B_w \in \mathbb{R}^{7\times 4}$. More specifically, $B_w = U\Sigma V'$ can be replaced by the equivalent $\tilde{B}_w = U\Sigma$.

4.2.3 Results

Problems 4.2.1 and 4.2.2 are stochastic viability problems with Gaussian-perturbed linear system and convex safe set.

SReachTools Since SReachTools analyzes safety under full state information, we use the safe sets $\mathcal{S}_1 \times \mathbb{R}^2$ and $\mathcal{S}_2 \times \mathbb{R}^6$ respectively. See Section 4.1.1 for details of the approaches used in SReachTools. For Problem 4.2.2, SReachTools declares all initial states whose first component lie in $\mathcal{S}_2$ can be driven to achieve a safety probability of 0.8.

StocHy synthesises the optimal policy by performing abstractions into an IMDP. For this method, we discretise the action space once again resulting in a discrete-time switched system. Note that StocHy could also have solved this problem using the MDP method, however due to the length of time needed to generate the abstractions for small abstraction errors this was left out of the comparison for this benchmark.

FAUST^2 performs policy synthesis by performing abstractions into MDP. Both the continuous and action space are discretised depending on the input maximum required level of abstraction, representative points are selected and an MDP is built. The abstract model error increases linearly with the time horizon, hence for longer time horizons one needs to discretise more finely in order to obtain models with smaller abstraction errors.

ϵ, δ-abstraction computes an abstract model that can be verified and for which a controller can be synthesised. This abstract model can either be a finite state MDP, a deterministic LTI

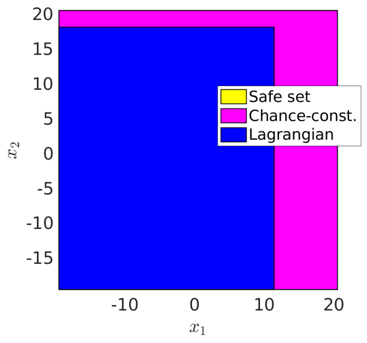


Figure 16: Stochastic viability set for Problem 4.2.1 for $x_3[0] = x_4[0] = 20$, computed using SReachTools. Code available in https://doi.org/10.24433/CO.8093142.v1.

instance	CS1BAS (Problem 4.2.1)				
tool	method	time [s]	Lower bound on max. safety prob.	abstraction error	Programming lang.
FAUST^2	Uniform	2940.00	≥ 0.80	1	Matlab
SReachTools	Chance const.	20.59	≥ 0.99	-	Matlab
SReachTools	Lagrangian	0.86	≥ 0.8	-	Matlab
StocHy	IMDP	0.43	≥ 0.99	0	c++
instance	CS2BAS (Problem 4.2.2)				
(ϵ,δ)-abstract	Truncate noise	0.51	>0.9994	6e-3	Python
SReachTools	Chance const.	8.35	≥ 0.99	-	Matlab
SReachTools	Lagrangian	18.96	≥ 0.8	-	Matlab

Table 10: Results for stochastic viability analysis for building automation systems. SReachTools does not use any abstraction, and analyzes the continuous-state problem directly.

model, or an LTI model with bounded disturbance. For the Building Automation System, only the 7 dimensional model was solved via this approach. For this model it was computed that truncating the noise of the LTI model would be more effective than computing an MDP model. System dynamics are loaded in Python based on the .mat file generated by Matlab in the other benchmarks. The Gaussian disturbance on this model is very low. An estimated amount of 17×10^9 would be needed to achieve an MDP that has similar performance and which does not have deterministic dynamics. In Table 10, the computation time is given together with the probability that the system starting from a stationary state leaves the safe set.

The obtained results for each of the specifications in the BAS benchmark are presented in Table 10. We have distinguished the calculation of a lower bound on the probability of verifying the formula from the computation of the abstraction error, since some techniques are statistical in nature and cannot assess the latter. Figure 16 shows the stochastic viability set for Problem 4.2.1.

4.3 Heated Tank

4.3.1 Description

The Heated Tank benchmark stems from safety literature; there it is a well-known example of a Piecewise Deterministic Markov Process (PDMP) [16]. This made the Heated Tank benchmark a logical candidate for early inclusion in the set of ARCH stochastic models [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 (p_{dryout}), Overflow probability ($p_{overflow}$), and Overheating probability ($p_{overheating}$). The heated tank benchmark has five versions:

- Version 1: Pumps and Valve have constant failure rates;
- Version 2: Pumps and Valve have mode dependent failure rates;
- Version 3: In version 1, Controller may fail to implement its switching decision;
- Version 4: In version 1, Pumps and Valve are repaired;
- Version 5: In version 1, failure rates depend on the liquid temperature.

For ARCH2018 the focus has been on the estimation of the dryout probability for version 4 [1] using the methods Modest Toolset and SDCPN & Monte Carlo simulation. The very reason for this focus was that it is a rare event estimation type of problem.

4.3.2 ARCH2019 objectives

For ARCH2019 one objective is to apply two novel methods to version 4 of the heated tank benchmark: HYPEG and SDCPN & IPS. A complementary objective is to make the heated tank benchmark more challenging by extending version 4 with other complications, e.g. from versions 2, 3 or 5. In order to specify such extensions in an unambiguous way it is needed to move from a textual type of heated tank benchmark description to a formal model description.

Because the heated tank benchmark falls in the class of PDMP, it would have been an option to use the PDMP or GSHP formalisms for its model description. Although these formalisms work well at an abstract model level, it is very demanding to specify all PDMP or GSHP model elements for a system involving multiple interacting systems, even for a relatively simple example as the heated tank benchmark. Most demanding is to specify the hybrid state transition probability measure element of a PDMP/GSHP. Therefore the second objective for ARCH2019 is to collect the formal models from Modest Toolset, SDCPN and HPnG of version 4 of the heated tank benchmark, and evaluate them on their ability in supporting future heated tank benchmark extensions.

4.3.3 ARCH2019 results

A. Rare event estimation results

For the heated tank version 4, Table 11 lists the parameter values that have been used in the estimation of the dry-out probability.

Parameter	Symbol	values in version 4
Failure rate of P1	λ_{P1}	$2.2831 \times 10^{-3}\ h^{-1}$
Failure rate of P2	λ_{P2}	$2.8571 \times 10^{-3}\ h^{-1}$
Failure rate of V	λ_V	$1.5625 \times 10^{-3}\ h^{-1}$
Repair rate	μ	$0.2\ h^{-1}$
Liquid flow	q	$0.6\ m/h$
Normalized input energy	E_{in}	$1\ ^oCm/h$
Luiqid inflow temperature	T_{in}	$15\ ^oC$
Initial liquid temperature	T_{init}	$15^2/_3\ ^oC$
Liquid overflow level	$H_{Overflow}$	$5\ m$
Liquid too high level	H_{High}	$1\ m$
Liquid initial level	H_{init}	$0\ m$
Liquid too low level	H_{Low}	-1 m
Liquid dryout level	H_{Dryout}	-5 m
Initial time	t_0	$0\ h$
End time	t_{end}	$500\ h$

Table 11: Parameters and their values in version 4

Table 12 gives the p_{dryout} estimation results obtained by the different methods; i.e. SAN&MC, Restart, Modest Toolset, SDCPN & MC, HYPEG and SDCPN & IPS respectively. Of these methods, Restart, Modest Toolset and IPS have in common of using splitting techniques in MC simulation of rare events. The importance functions that are used by Restart and Modest Toolset counts the number of component failures that are relevant for dry-out. The importance function that is used by IPS is the distance between liquid height $x_{H,t}$ and H_{Dryout}.

Method	SAN &MC [15]	Restart [51]	MODEST TOOLSET [1]	SDCPN &MC [1]	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	$\pm 1.4 \times 10^{-4}$	$\pm 3.4 \times 10^{-4}$	$\pm 0.26 \times 10^{-4}$	$\pm 1.4 \times 10^{-4}$	$\pm 1.0 \times 10^{-4}$	$\pm 0.24 \times 10^{-4}$

Table 12: p_{dryout} estimation results: two from safety literature (SAN & MC and Restart), two from methods applied in ARC2018 (Modest Toolset and SDCPN & MC), and two from the new methods HYPEG and SDCPN & IPS.

B. Modest Toolset model of heated tank benchmark version 4

For the readers that are not familar with Modest Toolset, we refer to [30]. We used the hierarchical description formalisms shown below for version 4.

```
const real $t_{end}$;

const real $H_{Overflow}$;
const real $H_{High}$;
const real $H_{Low}$;
const real $H_{Dryout}$;

const real $\lambda_{P1}$;
const real $\lambda_{P2}$;
const real $\lambda_{V}$;
const real $\mu$;
const real $C_{fail} = 0$;

action init;

bool on_p1 = true, on_p2 = false, on_v = true;
int(0..2) fail_p1, fail_p2, fail_v;

var $x_{H,t} = 0$;
der($x_{H,t}$) = (on_p1 && fail_p1 == 0 || fail_p1 == 1 ? 0.6 : 0) + (on_p2 && fail_p2 == 0 ||
fail_p2 == 1 ? 0.6 : 0) + (on_v && fail_v == 0 || fail_v == 1 ? -0.6 : 0);

bool overflow, dryout;
int(0..2) grace;

property $p_{dryout}$ = Pmax(<>[T<= $t_{end}$] dryout);
property IDryOut = (fail_p1 == 2 ? 1 : 0) + (fail_p2 == 2 ? 1 : 0) + (fail_v == 1 ? 1 : 0);

process Pump1()
{
    clock c; real x;
    invariant(false) init {= x = Exp($\lambda_{P1}$) =};
    do {
        when(c >= x) invariant(c <= x) palt {
        :1: {= fail_p1 = 1 =}
        :1: {= fail_p1 = 2 =}
        };
        when(grace ! = 0) invariant(grace == 0) {= c = 0, x = Exp($\mu$) =};
        alt {
        :: when(c >= x) invariant(c <= x) {= fail_p1 = 0, c = 0, x = Exp($\lambda_{P1}$) =}
        :: when(grace == 2) invariant(grace ! = 2) break
        }
    }
}

process Pump2()
{
    clock c; real x;
    invariant(false) init {= x = Exp($\lambda_{P2}$) =};
    do {
        when(c >= x) invariant(c <= x) palt {
        :1: {= fail_p2 = 1 =}
        :1: {= fail_p2 = 2 =}
        };
        when(grace ! = 0) invariant(grace == 0) {= c = 0, x = Exp($\mu$) =};
        alt {
        :: when(c >= x) invariant(c <= x) {= fail_p2 = 0, c = 0, x = Exp($\lambda_{P2}$) =}
        :: when(grace == 2) invariant(grace != 2) break
        }
    }
}

process Valve()
{
    clock c; real x;
    invariant(false) init {= x = Exp($\lambda_V$) =};
    do {
        when(c >= x) invariant(c <= x) palt {
        :1: {= fail_v = 1 =}
        :1: {= fail_v = 2 =}
        };
        when(grace ! = 0) invariant(grace == 0) = c = 0, x = Exp($\mu$) =;
        alt {
        :: when(c >= x) invariant(c <= x) = fail_v = 0, c = 0, x = Exp($\lambda_V$) =
        :: when(grace == 2) invariant(grace != 2) break
        }
    }
}

process Controller()
{
    process Normal()
    {
        alt {
        :: invariant(level <= $H_{High}$) when(level >= $H_{High}$) palt {
            :1 − $C_{fail}$: {= on_p1 = false, on_p2 = false, on_v = true, grace = 1 =}
            :$C_{fail}$: {= grace = 1 =}
            }; High()
        :: invariant(level >= $H_{Low}$) when(level <= $H_{Low}$) palt {
            : 1 − $C_{fail}$ : {= on_p1 = true, on_p2 = true, on_v = false, grace = 1 =}
            :  $C_{fail}$ : {= grace = 1 =}
            }; Low()
        }
    }

    process High()
    {
        invariant(level >= $H_{Low}$) when(level <= $H_{Low}$) palt {
        : 1 − $C_{fail}$ : {= on_p1 = true, on_p2 = true, on_v = false =}
        : $C_{fail}$ : {==}
        }; Low()
    }
    process Low()
    {
        invariant(level <= $H_{High}$) when(level >= $H_{High}$) palt {
        : $1 - C_{fail}$ : {= on_p1 = false, on_p2 = false, on_v = true =}
        : $C_{fail}$ : {==}
        }; High()
    }
    Normal()
    }
process Observer()
{
    par {
    :: invariant(level <= $H_{Overflow}$) when(level >= $H_{Overflow}$) {= overflow = true, grace = 2
=}
    :: invariant(level >= $H_{Dryout}$) when(level <= $H_{Dryout}$) {= dryout = true, grace = 2 =}
    }
}
par {
:: Pump1()
:: Pump2()
:: Valve()
:: Controller()
:: Observer()
}
```

C. HPnG model of heated tank benchmark version 4

Since HYPEG is a simulator for hybrid Petri nets with general transitions (HPnGs) [27], the heated tank benchmark has been modeled as an HPnG. A graphical representation of the HPnG for version 4 of the heated tank benchmark is presented in Figures 18 to 21, divided into four parts, which compose one large HPnG (connected via places with identical names in different figures).

An HPnG consists of places, transitions and arcs, whose graphical representation is presented in Figure 17. Discrete places hold a natural number of tokens and continuous places contain an amount of fluid. Continuous transitions carry fluid between continous places via continuous arcs, whereas we separate between static continuous transitions with a constant fluid rate (r) and dynamic continuous transition, whose fluid rate (also denoted by r) might depend on the current marking (tokens or fluid level) of places in the HPnG. Tokens are transported via immediate, deterministic or general transitions via discrete arcs. Immediate transitions fire as soon as enabled, whereas deterministic transitions fire after a pre-defined period of time. The firing time of general transitions follow a continuous probability distribution. The enabling status of transitions can be controlled by test arcs or inhibitor arcs, whereas for test arcs, the marking of the connected place has be greater or equal to the arc's weight (w) to enable the connected transition, and for inhibitor arcs it has to be lower than the weight.

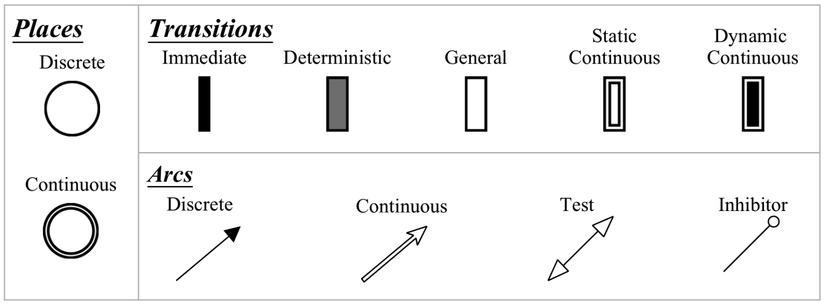


Figure 17: Graphical representation of HPnG components [37].

Figure 18 shows how the state of the valve is modeled via four different discrete places `V_On`, `V_Off`, `V_SOn`, `V_SOff`. The token (small black circle) can be transported to another discrete place, thus switching the valve state via general transitions, e.g. `V_On_SOn`, or immediate transitions, e.g. `V_On_Off`. The firing times of the general transitions for the valve follow exponential distributions with rate λ_V. The immediate transitions can only fire when there is a token in the places `Increase` or `Decrease`, due to the connecting test arcs. The state of the valve is further stored in the places `Valve_On` and `Valve_Off`, which is required for the filling of the tank (see Figure 20). When the valve is in stuck mode, the repair is modeled by the general transitions `V_SOff_Repair` resp. `V_SOn_Repair`, which are enabled when the place `Valve_grace` holds a token and follow an exponential distribution with repair rate μ.

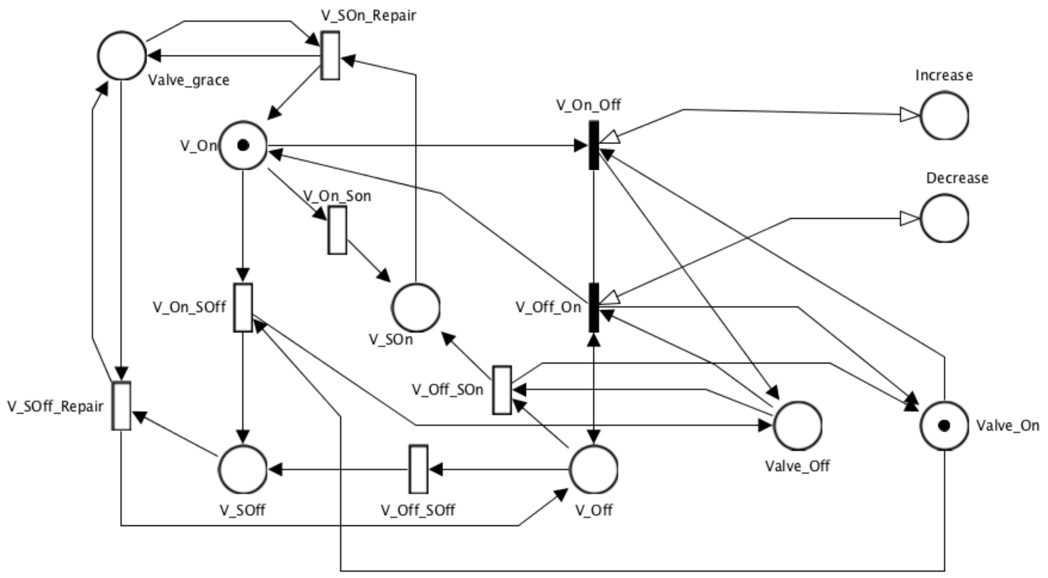


Figure 18: HPnG of the heated tank version 4: Valve.

The states of the pumps are modeled in a similar way, as shown in Figure 19, however the connections to the places `Increase` and `Decrease` are switched. The general transitions for the pumps follow exponential distributions with rate λ_{P_1} respectively λ_{P_2}. The number of activated pumps is counted via the places `No_Pump`, `One_Pump` and `Two_Pumps`.

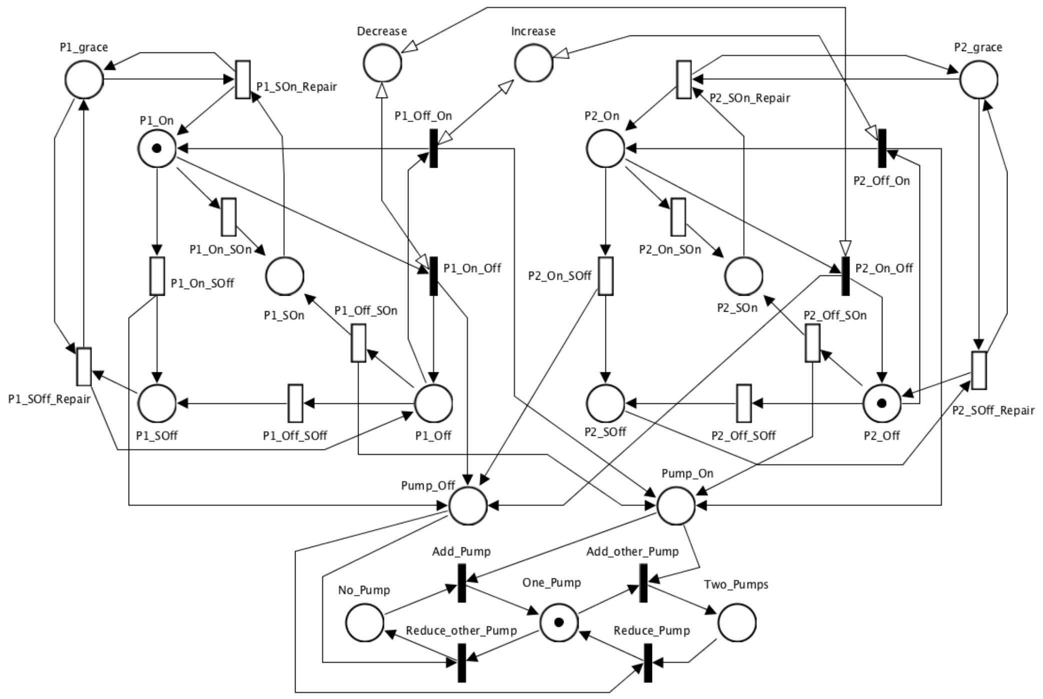


Figure 19: HPnG of the heated tank version 4: Pumps.

Figure 20 shows how the heated tank and its controller are modeled. The fluid level of the heated tank is modeled by the continuous place x_H , which has two static continuous input transitions H_in1 and H_in2, with a rate of q resp. $2 \cdot q$, and one static continuous output transition H_out with rate q. The transitions get enabled via test arcs or inhibitor arcs, depending on the number of tokens in the places One_Pump and Two_Pumps and Valve_On.

The controller is initially in the Normal state, but moves to Increase or Decrease, when the transition Normal_Decrease resp. Normal_Increase fires. Since Normal_Decrease is controlled by a test arc, it gets enabled when the fluid level of x_H reaches the weight of the test arc, i.e. H_{High}. On the other hand, Normal_Increase is controlled by an inhibitor arc and thus gets enabled when x_H $< H_{Low}$. In a similar way, the token can further move between Increase and Decrease via the transitions Decrease_Increase and Increase_Decrease. The firing of Normal_Increase resp. Normal_Decrease further adds a token into the places Valve_grace, P1_grace and P2_grace, activating the repair mode for the components. Each of these places can lose its token via the connected immediate transitions for the case of an outflow or dryout, i.e. when the fluid level of x_H reaches H_{Dryout} or $H_{Overflow}$.

The temperature of the tank is modeled by another continuous place x_T, as shown in Figure 21. It is connected to different dynamic continuous input and output transitions, which are enabled depending on the state of the pumps and the current fluid level of x_H (via test and inhibitor arcs). E.g., for one activated pump and a positive fluid level, transitions T_in1 and T_out1 are enabled, such that the input rate for x_T equals $(T_{in} \cdot q + E_{in})/(\texttt{x\_H} - H_{Dryout})$ and the output rate is $(q \cdot \texttt{x\_T})/(\texttt{x\_H} - H_{Dryout})$.

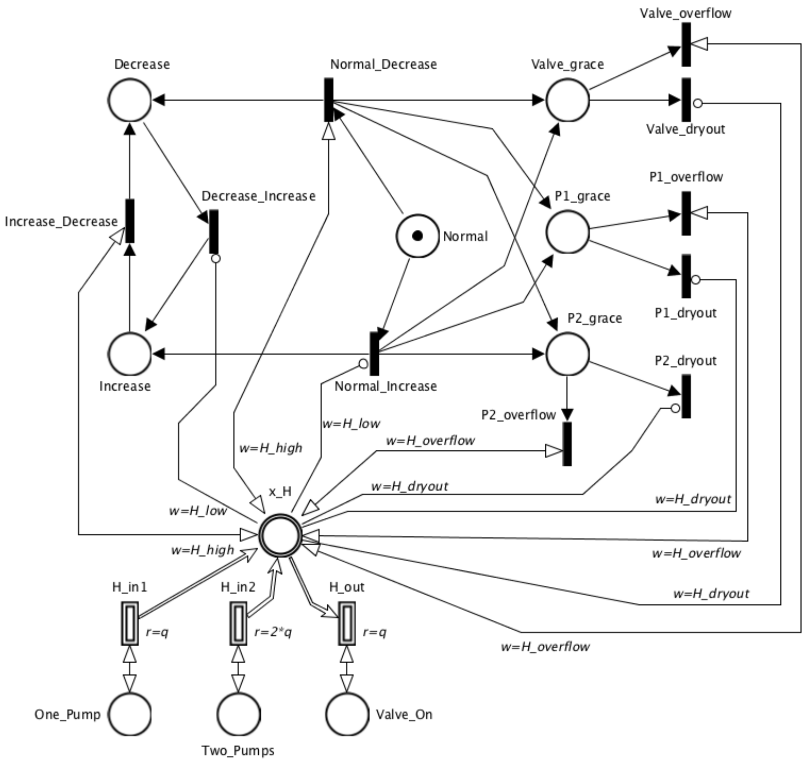


Figure 20: HPnG of the heated tank version 4: Tank and Controller.

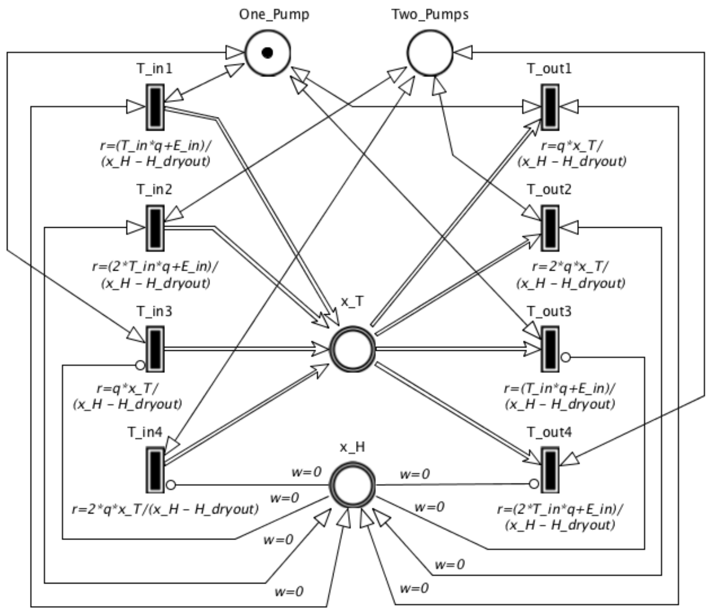

One_Pump
Two_Pumps
T_in1
T_out1
r=(T_in*q+E_in)/(x_H − H_dryout)
r=q*x_T/(x_H − H_dryout)
T_in2
T_out2
r=(2*T_in*q+E_in)/(x_H − H_dryout)
r=2*q*x_T/(x_H − H_dryout)
T_in3
x_T
T_out3
r=q*x_T/(x_H − H_dryout)
r=(T_in*q+E_in)/(x_H − H_dryout)
T_in4
x_H
T_out4
r=2*q*x_T/(x_H − H_dryout)
r=(2*T_in*q+E_in)/(x_H − H_dryout)
w=0
w=0
w=0
w=0
w=0
w=0
w=0
w=0

Figure 21: HPnG of the heated tank version 4: Temperature.

D. SDCPN model of heated tank benchmark version 4

The graphical part of the SDCPN model is given in Figure 22; it has one local Petri Net (LPN) for each of the systems Pump 1(P1), Pump 2(P2), Valve(V), Tank(T), and Controller(C). Figure 22 shows the places ◯, the transitions □ and the arcs (→ or –●) within and between each of these five LPN's. Figure 22 also shows which places have a token at initial moment t_0. Each LPN is such that it always has exactly one token.

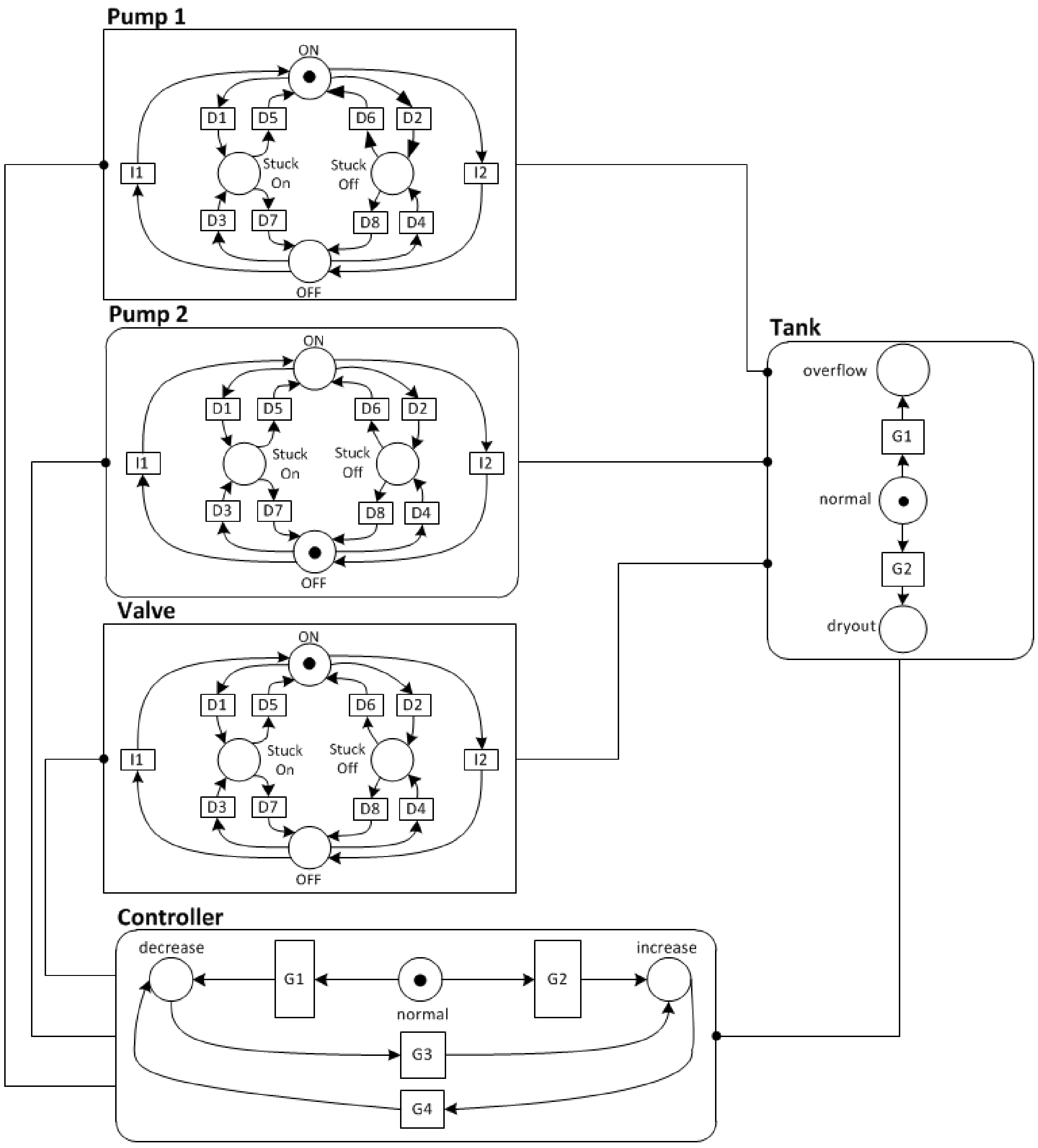


Figure 22: Graphical part of the SDCPN model for the heated tank benchmark, version 4.

For readers that are not familiar with SDCPN, we provide a short explanation of the graphical elements in this SDCPN model. Arcs ending with an arrow are normal arcs; arcs ending with a ball are enabling arcs. In contrast with a normal arc, an enabling arc can only go from a place to a transition, and this transition does not consume any token from this place. Each enabling arc that starts from the boundary of a local Petri net (LPN) and ends at the boundary of another local Petri net (LPN), say from LPN A to LPN B, represents multiple enabling arcs, i.e. one enabling arc for each combination of a place in LPN A and a transition in LPN B. This allows that all transitions in LPN B have full access to the hybrid state information available in LPN A. There are three types of transitions: Immediate transitions (I), Delay transitions (D) and Guard transitions (G). A transition is said to be pre-enabled if each place from which it receives an arc has a token. If an immediate transition is pre-enabled it fires a token to the places where outgoing arcs lead to. If a delay transition is pre-enabled then its firing happens stochastically at a given firing rate. If a guard transition is pre-enabled, then it fires as soon as its guard condition is satisfied.

If in LPN "Tank" there is a token in place Normal, i.e $\theta_{T,t} = Normal$, then to this token a 2-D Euclidean valued process $\{x_{H,t}, x_{T,t}\}$ is connected which is the solution of the following differential equations:

$$dx_{H,t} = q \cdot (\chi_{P1,t} + \chi_{P2,t} - \chi_{V,t}) \cdot dt \qquad \text{with } x_{H,0} = H_{init}$$
$$dx_{T,t} = [q \cdot (\chi_{P1,t} + \chi_{P2,t})(T_{in} - x_{T,t}) + E_{in}]/(x_{H,t} - H_{Dryout}) \cdot dt \qquad \text{with } x_{T,0} = T_{init}$$

where for $U \in \{P1, P2, V\}$: $\chi_{U,t} = 1$ if LPN U has a token in place On or in place $StuckOn$, else $\chi_{U,t} = 0$.

In LPN "Tank" the guard conditions of G1 and G2 are $x_{H,t} \geq H_{Overflow}$ and $x_{H,t} \leq H_{Dryout}$ respectively.

In LPN "Controller" the guard conditions of G1, G2, G3 and G4 are $x_{H,t} \geq H_{High}$, $x_{H,t} \leq H_{Low}$, $x_{H,t} \leq H_{Low}$ and $x_{H,t} \geq H_{High}$ respectively.

In LPN's Pump 1, Pump 2 and Valve the following applies to the transitions:

- Di, i $\in \{1, 2, 3, 4\}$ fires at rate λ_U, $U \in \{P1, P2, V\}$.
- Di, i $\in \{5, 6, 7, 8\}$ fires at rate $\frac{1}{2}\mu \neq 0$, iff the token of LPN Controller is not in place $Normal$.
- In LPN P_j, j=1,2, I1 fires if P_j's place Off has a token as well as LPN C's place $Increase$,
- In LPN V, I1 fires if V's place Off has a token as well as LPN C's place $Decrease$,
- In LPN P_j, j=1,2, I2 fires if P_j's place On has a token as well as LPN C's place $Decrease$,
- In LPN V, I2 fires if V's place On has a token as well as LPN C's place $Increase$.

The Generalised Stochastic Hybrid Process (GSHP) defined by this SDCPN model is $\{x_t, \theta_t\}$ with $x_t = [x_{H,t}, x_{T,t}]^T$ and $\theta_t = [\theta_{U,t}; U = P1, P2, V, T, C]^T$, where the value of process $\theta_{U,t}$ equals the name of the place where the token in LPN U is at moment t.

E. Relevant extensions of heated tank benchmark

From a rare event estimation challenge perspective the following five extensions of heated tank version 4 are of relevance:

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

II. Non-zero probability of not communicating a decision made (e.g. in version 3)

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

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

V. Brownian motion in a Euclidean state component, e.g. in the heat source E_{in}.

For the modelling of these extensions it also is quite helpful if the syntax supports compositional modelling. In Table 13, it is indicated which extensions are supported by PDMP, by GSHP and by which of the three syntax models that have been used for the Heated tank benchmark, i.e. Modest Toolset, SDCPN and HPnG.

Model syntax	I	II	III	IV	V	Compositional modelling
PDMP	Yes	Yes	Yes	Yes	No	No
GSHP	Yes	Yes	Yes	Yes	Yes	No
Modest Toolset	Yes	Yes	No	Yes	No	Yes
HPnG	Yes	Yes	No	Yes	No	Yes
SDCPN	Yes	Yes	Yes	Yes	Yes	Yes

Table 13: Capabilities of the methods to support relevant extensions.

4.4 Tandem Network

4.4.1 Model

Let us first give solution to the time bounded reachability problem over CTMCs in general and then we will discuss *tandem network* as a concrete example.

A continuous-time Markov chain (CTMC) $\mathcal{M} = (S_{\mathcal{M}}, R)$ consists of a finite set $S_{\mathcal{M}} = \{1, 2, \cdots, |S_{\mathcal{M}}|\}$ of states, a rate matrix $S_{\mathcal{M}} \times S_{\mathcal{M}} \to \mathbb{R}_{\geq 0}$. Intuitively, $R(s, s') > 0$ indicates that a transition from s to s' is possible and that the timing of the transition is exponentially distributed with rate $R(s, s')$. Q is the *infinitesimal generator* matrix of $\mathcal{M}$ defined as $Q = R - diag_s(\sum_{s' \in S_{\mathcal{M}}} R(s, s'))$. Note that $\sum_{s'} Q(s, s') = 0$ for any $s \in S_{\mathcal{M}}$. Let $\mathcal{M} = (S \uplus \{\mathbf{good}, \mathbf{bad}\}, R)$ be a CTMC, with $|S| = m$, and absorbing states **good** and **bad**. For computing the reachability probability vector $Z(t)$ we need to solve for the following differential equation

$$\frac{d}{dt}Z(t) = QZ(t), \quad Z(0) = \mathbf{1}(\mathbf{good}), \tag{6}$$

where $Z(t) \in \mathbb{R}^m$ is a column vector with elements $Z_i(t) = \mathrm{Prob}^{\mathcal{M}}(\mathbf{1}(s_i), t)$, denoting the probability of reaching the **good** state after time t for the i^{th} state and $\mathbf{1}(i)$ denotes a $m \times 1$ vector which contains zeros everywhere and 1 on its i^{th} position. In order to solve Equation (6) the most common approach is to use uniformization. By using uniformization, one can compute time bounded reachability for an arbitrary time bound with an arbitrary precision. However, it is well-known that for large time bounds, uniformization is not fast enough.

A Lyapunov based approach is proposed in [40] to compute the solution of (6) approximately with formal error bounds. The approach generalizes the binary interpretation of bisimulation relations to a quantitative setting. It allows states to be members of different partition sets with some membership degree and utilizes Lyapunov theory to compute formal error bounds. From

Time bound [h]	run-time [s]	reach. prob
333.34	0.019	0.153
666.67	0.011	0.284
1000	0.012	0.394
1333.34	0.012	0.487
1666.67	0.015	0.566

Table 14: Results for running LyapMMC over *tandem network* using $\varepsilon(T)$=0.01

Formal error bound ($\varepsilon(T)$) [h]	run-time [s]	Percentage of order reduction
0.5	0.0056	25.2%
0.4	0.008	16.5%
0.3	0.009	9.56%
0.2	0.01	3.4%
0.1	0.018	0%

Table 15: Results for running LyapMMC over *tandem network* with different $\varepsilon(T)$

the computation pespective, instead of (6), a set of differential equations of order r ($r << m$) will be solved,

$$\frac{d}{dt}\bar{Z}(t) = \bar{Q}\bar{Z}(t), \quad \bar{Z}(0) = \bar{Z}_0, \tag{7}$$

where $\bar{Z}(t) \in \mathbb{R}^{r\times 1}$ and $\bar{Q}$, $\bar{Z}_0$ should be computed such that

$$\left|Z(t) - P\bar{Z}(t)\right| \leq \varepsilon(t), \tag{8}$$

where $P \in \mathbb{R}^{n\times r}$ is a projection matrix and ε denotes the time varying upper bound over order reduction error. The details on the computation of $\bar{Q}, \bar{Z}_0$ and P and the characterization of $\varepsilon(t)$ can be found in [40].

We consider **cap** = 10 which results in a CTMC with 231 states. We can choose values $\mu_1 = 1$, $\mu_2 = 2$, $\kappa = 2$, $\lambda = 4$, $a = 1$, $b = 0$ and time bounds $T = 60000 \times r$ for r being integers from 1 to 5. Once the tandem network is modelled as a CTMC, the generator matrix Q is computed such that *block state* (chosen as the **good** (absorbing) state) is made absorbing. All we need to do is to compute $\bar{Q}, \bar{Z}_0$, P for $\varepsilon(t)$ and solve (7) up-to time T.

4.4.2 Results

We present the results in two tables. Table 14 demonstrates results of running LyapMMC over the described tandem network for five different time bounds (T), while $\varepsilon(T)$ is set to 0.01. Results for LyapMMC are given in Table 14. Surprisingly, it can be noticed that 5 times increasing the time horizon not only does not result in a linear increase in computation time but also keeps the run-time almost the same. Next, in Table 15, we report the effect of changing the formal error bound ε on the run-time and the percentage of reduction in the order of differential equation, while time bound is set to $T = 1666.7h$. It can be noticed that by reducing the precision, execution time decreases as the order of differential equations in (7) decreases. Interestingly, even for relatively large formal error bounds, the computed error bound is very small. It can also be observed that one can significantly reduce the run-time by choosing larger error bounds.

5 Discussion

A highlight of this year's edition has been the evident and ever growing interest in tools that can verify and perform synthesis of stochastic models. We have seen the introduction of three new software tools and of one early synthesis framework, which together can tackle different problems and benchmarks. Building on the experience gathered from the outcomes of the friendly competition and from the work on the benchmarks (old and new), some key points for future work have been identified as follows:

- one of the key issues is the absence of an accepted formal modelling description for the benchmarks, which would (among other things) facilitate their comparison and the running of experiments on them. It is understood that there is a plethora of stochastic models characterisations, each of which are tailored to specific tasks and problems (e.g., rare-event setup, or a control setup), however there is a need for a common modelling language. Related to this is also the need for a common format for sharing models used in the benchmarks (e.g. .MAT files);
- many tools have different ways to describe outcomes, which are tailored towards the underlying of algorithms used within specific tools. A common standard for the output of tools is required (e.g., max satisfiability for a specific initial condition);
- it was noted that in most of the current benchmarks the effect of noise is small, hence we should design models where noise has a larger effect on the dynamics;
- a realistic safety analysis problem of industrial relevance is typically is much larger than what can be currently handled in an ARCH benchmark;
- there is a need for more diverse specifications, beyond safety or reach-avoid;
- finally, we should work towards having all tools working on benchmarks run over the same machine.

6 Conclusion

This report has presented the results of a first friendly competition for the formal verification of stochastic models, as part of the ARCH'19 Workshop. The reports of other categories can be found in the proceedings and on the ARCH website: cps-vo.org/group/ARCH.

7 Acknowledgments

This work is partially supported by the Alan Turing Institute, UK and Malta's ENDEAVOUR Scholarships Scheme. We thank Dr. Mariken Everdij (NLR) for syntax verification of the SDCPN model specification for the heated tank version 4.

References

[1] 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. *EPiC Series in Computing*, 54:71 – 103, 2018.

[2] 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.

[3] A Absalom and G Kenny. 'paedfusor' pharmacokinetic data set. *British Journal of Anaesthesia*, 95(1):110–110, 2005.

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

[5] H.A.P. Blom, H. Ma, and G.J. Bakker. Interacting Particle System-based Estimation of Reach Probability for a Generalized Stochastic Hybrid System. In *IFAC Conference on Analysis and Design of Hybrid Systems*, pages 79–84, Oxford, UK, 2018.

[6] Olivier Bouissou, Eric Goubault, Sylvie Putot, Aleksandar Chakarov, and Sriram Sankaranarayanan. Uncertainty propagation using probabilistic affine forms and concentration of measure inequalities. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 225–243. Springer, 2016.

[7] Carlos E. Budde, Pedro R. D'Argenio, and Arnd Hartmanns. Better automated importance splitting for transient rare events. In *Third International Symposium on Dependable Software Engineering. Theories, Tools, and Applications (SETTA)*, volume 10606 of *Lecture Notes in Computer Science*, pages 42–58. Springer, 2017.

[8] Carlos E. Budde, Pedro R. D'Argenio, Arnd Hartmanns, and Sean Sedwards. A statistical model checker for nondeterminism and rare events. In *24th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 10806 of *Lecture Notes in Computer Science*, pages 340–358. Springer, 2018.

[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] Manuela L Bujorianu and John Lygeros. Toward a general theory of stochastic hybrid systems. In *Stochastic hybrid systems*, pages 3–30. Springer, 2006.

[11] 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.

[12] 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.

[13] 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.

[14] Frédéric Cérou, Pierre Del Moral, François Le Gland, and Pascal Lezaud. Genetic genealogical models in rare event analysis. *ALEA, Latin American Journal of Probability and Mathematical Statistics*, 1:181–203, 2006.

[15] 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.

[16] M.H.A. Davis. *Markov models and optimization*, volume 49. Chapman and Hall/CRC, 1993.

[17] 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.

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

[19] 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, *Electronic Proceedings in Theoretical Computer Science*, volume 20, pages 1–15. 2010.

[20] 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*, pages 227–252. I-Tech Education and Publishing, 2010.

[21] M.H.C. Everdij, Margriet B. Klompstra, H.A.P. Blom, and Bart Klein Obbink. Compositional Specification of a Multi-agent System by Stochastically and Dynamically Coloured Petri Nets. In H.A.P. Blom and J. Lygeros, editors, *Stochastic Hybrid Systems: Theory and safety critical applications*, pages 325–350. Springer, 2006.

[22] Victor Gan, Guy A Dumont, and Ian Mitchell. Benchmark problem: A pk/pd model and safety constraints for anesthesia delivery. In *ARCH@ CPSWeek*, pages 1–8, 2014.

[23] 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*, pages 1–10. IEEE CS Press, 2013.

[24] Hamed Ghasemieh, Anne Remke, and Boudewijn R. Haverkort. Survivability analysis of a sewage treatment facility using hybrid Petri nets. In *Performance Evaluation*, pages 1–21. Elsevier, 2015.

[25] 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.

[26] J. Greifeneder and G. Frey. Probabilistic hybrid automata with variable step width applied to the analysis of networked automation systems. In *Proc. 3rd IFAC Workshop on Discrete Event System Design (DESDes'06)*, pages 283–288. IFAC, September 2006.

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

[28] 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.

[29] Sofie Haesaert, Sadegh Esmaeil Zadeh 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.

[30] E. M. Hahn, A. Hartmanns, H. Hermanns, and J.P. Katoen. A compositional modelling and analysis framework for stochastic hybrid systems. *Formal Methods in System Design*, 43(2):191–232, 2013.

[31] Ernst Moritz Hahn, Arnd Hartmanns, Holger Hermanns, and Joost-Pieter Katoen. A compositional modelling and analysis framework for stochastic hybrid systems. *Formal Methods in System Design*, 43(2):191–232, 2013.

[32] 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.

[33] M. Herceg, M. Kvasnica, C.N. Jones, and M. Morari. Multi-Parametric Toolbox 3.0. In *Proc. of the European Control Conference*, pages 502–510, Zürich, Switzerland, July 17–19 2013. http://control.ee.ethz.ch/~mpt.

[34] H. Hermanns. *Multi terminal binary decision diagrams to represent and analyse continuous time Markov chains*, pages 188–207. 1999.

[35] Holger Hermanns, Joost-Pieter Katoen, Joachim Meyer-Kayser, and Markus Siegle. A tool for model-checking markov chains. *International Journal on Software Tools for Technology Transfer*, 4(2):153–172, Feb 2003.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

[40] Mahmoud Salamati, Sadegh Soudjani, and Rupak Majumdar. Approximate time bounded reachability for ctmcs and ctmdps: A lyapunov approach. In *QEST*, volume 11024 of *Lecture Notes in Computer Science*, pages 389–406. Springer, 2018.

[41] 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*, 2019. (accepted).

[42] Cristian-Ioan Vasile Rohan Thakker Ali-akbar Agha-mohammadi Aaron D. Ames S.Haesaert, P. Nilsson 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.

[43] 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.

[44] 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.

[45] 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.

[46] Sadegh Soudjani. *Formal Abstractions for Automated Verification and Synthesis of Stochastic Systems.* PhD thesis, Delft Center for Systems and Control, Technische Universiteit Delft, Delft, NL, November 2014.

[47] 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.

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

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

[50] Tino Teige, Andreas Eggers, and Martin Fränzle. Constraint-based analysis of concurrent probabilistic hybrid systems: An application to networked automation systems. *Nonlinear Analysis: Hybrid Systems*, 5(2):343–366, May 2011.

[51] 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.

[52] Abraham P. Vinod, Joseph Gleason, and Meeko Oishi. SReachTools: A MATLAB stochastic reachability toolbox, April, 2019. Available online: https://unm-hscl.github.io/SReachTools/.

[53] 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.

[54] 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.

[55] Abraham P. Vinod and Meeko M. K. Oishi. Stochastic reachability of a target tube: Theory and computation. 2018. Available online: https://arxiv.org/abs/1810.05217.

[56] Abraham P. Vinod and Meeko M. K. Oishi. Scalable underapproximative verification of stochastic LTI systems using convexity and compactness. In *Proceedings of the 21st International Conference on Hybrid Systems: Computation and Control (Part of CPS Week)*, pages 1–10. ACM, April, 2018.

A Specification of Used Machines

A.1 $\mathrm{M}_{\mathbf{FAUST^2}}$

- Processor: Intel Core i7-8550U CPU @ 1.80GHz $\times$ 8
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 8337

A.2 $\mathrm{M}_{\mathbf{hypeg}}$

- Processor: Intel Core i5-5257U CPU @ 2.70GHz (2 cores)
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 4403

A.3 $\mathrm{M}_{\mathbf{Modest}}$

- Processor: Intel Core i7-4790 @ 3.6-4.0 GHz (4 cores $\times$ 2 threads)
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 9994

A.4 $\mathrm{M}_{\mathbf{sdcpn}}$

- Processor: Intel Core i5-2400 @ 3.10GHz
- Memory: 8.00 GB (7.88 GB usable)
- Average CPU Mark on www.cpubenchmark.net: 5949

A.5 $\mathrm{M}_{\mathbf{SReachTools}}$

- Processor: Intel Xeon CPU @ 3.4GHz
- Memory: 32.00 GB
- Average CPU Mark on www.cpubenchmark.net:

A.6 $\mathrm{M}_{\mathbf{StocHy}}$

- Processor: Intel Core i7-8550U CPU @ 1.80GHz $\times$ 8
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 8337

A.7 $\mathbf{M}_{\epsilon,\delta}$ (1)

- Processor: Intel Core i5-8259U CPU @ 2.30GHz (4 cores)
- Memory: 16.00 GB
- Average CPU Mark on `www.cpubenchmark.net`: 11003

A.8 $\mathbf{M}_{\epsilon,\delta}$ (2)

- Processor: Intel Core i5-7360U CPU @ 2.30GHz (2 cores)
- Memory: 8.00 GB
- Average CPU Mark on `www.cpubenchmark.net`: 5823

B BAS Explanation of the results

B.0.1 Comments on CS2BASS – ϵ, δ-abstraction

As a first step, we compute the hyper-cube to which the standard Gaussian noise belongs with probability equal to 0.99. For w in this hypercube, the largest disturbance is $\|B_w w\|_\infty \leq .07$. If the cells of our gridding-based approach are of the same magnitude or smaller than $\|B_w w\|_\infty$ this implies that more than 17×10^9 cells are needed. This is actually the amount of cells from which the transition probabilities take values in $[0, 0.99)$.
Truncated noise model. We now compute an abstract model $\tilde{M}$ with truncated noise, that is the Gaussian realizations $(w_1(t), w_2(t), w_3(t), w_4(t))$ are limited to $w_1 \in [-4.215, 4.215]$, $w_2 \in [-4.215, 4.215]$, $w_3 \in [-4.215, 4.215]$, $w_4 \in [-4.215, 4.215]$. This model $\tilde{M}$ is approximately bisimilar model to the original model M, this is denoted as $\tilde{M} \simeq^{\epsilon=0}_{\delta=1.00e-04} M$. The δ value follows from the truncation, which is obtained as follows:

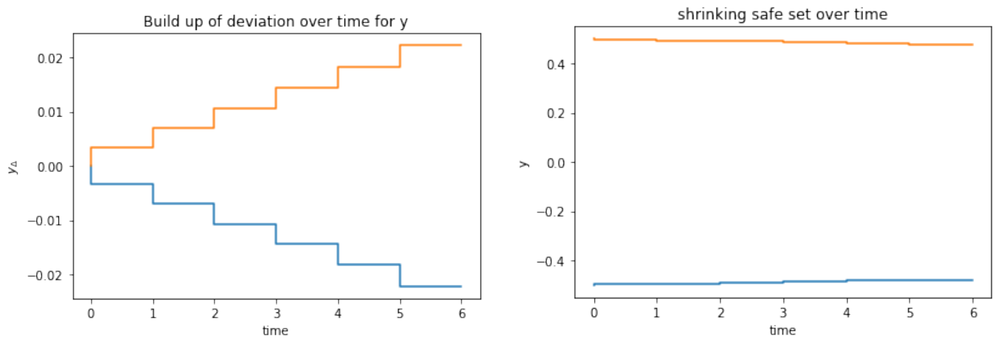
```
In [3]:  delta = 0.0001
         conf = 1-delta
         conf_i = conf ** (1/4)
         w_list_= [list(scipy.stats.norm.interval(conf_i, loc=0, scale=1)) ] *4
         # domain of truncated noise
```

Since the specification is a simple bounded horizon safety specification, and since the model is stable, we have opted to compute forward reach set using hyper-cubes. This is efficient and simple to implement using a one-step reachability mapping.

```
In [4]: def forward_box(AB, xbox, wbox):
            dist = np.concatenate([[np.array(list_items).dot(AB.T)]
                                   for list_items in itertools.product
                                   (*xbox,*wbox)])
            return [row  for row in np.array((np.amin(dist, axis=0),
               np.amax(dist, axis=0))).T]
        AB = np.concatenate([sys_s.a,sys_s.Bw], axis=1)
```

More precisely, this function propagates the effect of the disturbance by encapsulating them in hyper-cubes. On the output, this gives a deviation over time as depicted in Figure 23a.

Figure 23a implies that for a model without noise, the safe set will be time varying. Figure 23b implies that when starting at a stationary temperature (at 0), the stationary policy will stay in the safe set for the abstract model with probability 1. Similarly, for the original model the satisfaction probability is $(1-\delta)^6 = 0.99940015$. By using toolboxes such as MPT (Matlab) or any other verification toolbox for non-stochastic systems similar results can be achieved over the truncated noise model.



(a) (b)

Figure 23: Figure of (a) deviation over time and (b) the effect of the disturbance on the allowed set for deterministic dynamics.