EPiC Series in Computing

Volume 54, 2018, Pages 71–103

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

ARCH-COMP18 Category Report: Stochastic Modelling

Alessandro Abate[1], Henk Blom[2], Nathalie Cauchi[1], Sofie Haesaert[3], Arnd Hartmanns[4], Kendra Lesser[5], Meeko Oishi[6], Vignesh Sivaramakrishnan[6], Sadegh Soudjani[7], Cristian-Ioan Vasile[8], and Abraham P. Vinod[6]

[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@nlr.nl
[3] California Institute of Technology, Pasadena, California, USA haesaert@caltech.edu
[4] University of Twente, Formal Methods and Tools group, Enschede, The Netherlands a.hartmanns@utwente.nl
[5] Verus Research, Atlanta, Georgia, USA kendra.lang@verusresearch.net
[6] University of New Mexico, Department of Electrical and Computer Engineering, New Mexico, USA {oishi,vigsiv}@unm.edu,aby.vinod@gmail.com
[7] School of Computing, Newcastle University, UK, Sadegh.Soudjani@ncl.ac.uk
[8] Massachusetts Institute of Technology, Cambridge, MA, USA cvasile@mit.edu

Abstract

This report presents the results of a friendly competition for formal verification and policy synthesis of stochastic models. The friendly competition took place as part of the workshop Applied Verification for Continuous and Hybrid Systems (ARCH) in 2018. In this first edition, we present five benchmarks with different levels of complexities and stochastic flavours. We make use of six different tools and frameworks (in alphabetical order): Barrier Certificates, FAUST^2, FIRM-GDTL, Modest, SDCPN modelling & MC simulation and SReachTools; and attempt to solve instances of the five different benchmark problems. Through these benchmarks, we capture a snapshot on the current state-of the art tools and frameworks within the stochastic modelling domain. We also present the challenges encountered within this domain and highlight future plans which will push forward the development of more tools and methodologies for performing formal verification and optimal policy synthesis of stochastic processes.

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 2018 friendly competition of the ARCH workshop[1] for the newly established *stochastic modelling* group. This new category in the friendly competition aims to push forward the verification of such models, and has the practical goal to test existing or prototype tools that verify a set of new benchmarks for such models.

Within this category, we have identified five novel benchmarks, comprising different model structures and dealing with diverse applications. All the benchmarks have varying levels of complexity, different type of stochasticity and different problem specifications. Using the current tools and algorithmic frameworks, we perform verification tasks on each of the benchmarks. The tools and frameworks used are (in alphabetical order): Barrier Certificates, FAUST^2, FIRM-GDTL, Modest, SDCPN modelling & MC simulation, SReachTools. We further identify the current challenges with running the benchmarks and set out a plan for this category within upcoming rounds of the ARCH competition.

This report has the following structure. Section 2 presents the benchmark descriptions which include a discussion of the individual models syntax and semantics, and a presentation of the specifications of interest. Next, in Section 3 we present the participating tools or algorithmic frameworks that are used to solve instances of the individual benchmarks. We present the results for each benchmark in Section 4. Finally, we identify key challenges and discuss future plans in Section 5.

2 Benchmarks

We discuss a new set of benchmarks for stochastic models, and attempt a uniform syntactic presentation of them. These are: an anesthesia delivery system; a building automation system; a heated tank; a lawn mower; and a benchmark dealing with a Mars rover. Each benchmark has specific features, which we delineate first. Next, we discuss each benchmark, the different problems to be solved together with specifications of interest, and different paths to their successful verification. Afterwards, the outcomes of the verification of each benchmark are presented in detail.

Specific modelling features We briefly list the key features of each benchmark:

- *Anaesthesia delivery system benchmark*: this benchmark is based on an automated anaesthesia delivery problem, with a human (an anaesthesiologist) in the loop [21, 5]. The model is described in the form of a discrete-time stochastic hybrid system [1] that depends both the current state of the model and a history of the past input actions by the anaesthesiologist. Stochasticity comes both in the form of process noise in the underlying continuous variables and in uncertainty on the applied input action. This benchmark focuses on two verification problems: (i) the stochasticity viability problem and (ii) the stochastic first-time hitting reach-avoid problem.

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

- *Building automation systems benchmark*: this benchmark is based on a library of models describing different components within a smart building system [10]. We focus on two instances of possible benchmarks, both of which describe discrete-time linear systems with varying numbers of continuous state variables. The overall benchmark focuses on the verification and policy synthesis around safety properties.

- *Heated tank benchmark*: this benchmark is based on a known model within the area of reliability and safety [33, 34, 41]. The benchmark is composed of discrete states that transition using Poisson arrivals, depending on the failure rates of the tank and on continuous variables describing the temperature and water level in the tank. The continuous variables evolve in continuous time. The difficulty in this benchmarks includes the very low probability of transitions between states due to the low failure rates.

- *Lawn mower benchmark*: this benchmark focuses on controller synthesis and on verification for a driverless industrial-size lawn mower [20]. While a number of diverse schemes can be used for synthesising a controller, we use one that has a simple switching mechanism for guiding the mower between way-points in the field. We describe discrete-time dynamics of the mower together with the controller architecture. The closed-loop model fits into the framework of discrete-time hybrid systems, which has six continuous states, two discrete modes, and nonlinear vector fields in each mode. Although statistical techniques such as multilevel Monte Carlo method [17] can be used to verify properties of the *continuous-time* dynamics presented in [20], we transform the model into *discrete-time polynomial* dynamics and use barrier certificates [27] for the verification task.

- *Mars rover benchmark*: this benchmark considers the problem of planetary exploration with a two-agent team composed of a Mars rover and a Mars helicopter, tasked to collect a set of samples from the Mars surface. Due to the high cost and risk associated with these missions, it is critical to have strong correctness and performance guarantees on the robot behaviours. This benchmark is composed of two agents described using partially observable models operating in a partially-known deterministic environment. It is tasked with complex mission specifications that could be used for real Mars rover operations [36, 35].

2.1 Anaesthesia Delivery System Benchmark

This section proposes a benchmark problem for stochastic verification techniques. Past benchmarks and work exist for automated anaesthesia delivery systems [21, 5]. We consider the problem of providing probabilistic guarantees of safety for the automated anaesthesia delivery problem with a human (anaesthesiologist) in the loop. We use a discrete-time stochastic hybrid system model to describe the patient's depth of hypnosis, based on the well-studied multi-compartment model for delivery of Propofol (anaesthetic) in paediatrics. Propofol can be delivered via two mechanisms: via a bolus dose, delivered by the human, or via infusion pump, whose rate is determined by the automation. We model the human's action (i.e., the delivery (or not) of a bolus dose of anaesthetic), as a non-deterministic, 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. For the infused Propofol, we presume a piecewise constant input that is bounded. Uncertainty in the system dynamics is captured via additive Gaussian noise. Two

problems of interest are 1) a viability problem, in which we seek to maximise the probability of the state lying within known, safe depths of hypnosis; and 2) a reach-avoid problem, in which we seek to maximise the probability of the state reaching a desirable depth of hypnosis, while staying within safe hypnosis depths.

2.1.1 Model formulation

We denote a discrete-time time interval by $\mathbb{N}_{[a,b]}$ for $a, b \in \mathbb{N}$ and $a \leq b$, which inclusively enumerates all integers in between (and including) a and b. We denote non-random vectors with an overline, random variables with bold case, random vectors with bold case and an overline, $I_n \in \mathbb{R}^n$ as the identity matrix, the Cartesian product of the set $\mathcal{G}$ with itself $k \in \mathbb{N}$ times as $\mathcal{G}^k$, and the cardinality of $\mathcal{G}$ with $|\mathcal{G}|$.

Propofol concentration in the patient with external drug administration The concentration of a drug administered to a patient is often represented by a multi-compartment model [26]. The concentration of Propofol in different compartments of the body are modelled using the three-compartment pharmacokinetic system [32, 30, 21]:

$$\dot{\overline{x}}(t) = \begin{bmatrix} -(k_{10}+k_{12}+k_{13}) & k_{12} & k_{13} \\ k_{21} & -k_{21} & 0 \\ k_{31} & 0 & -k_{31} \end{bmatrix} \overline{x}(t) + \begin{bmatrix} \frac{1}{V_1} \\ 0 \\ 0 \end{bmatrix} u(t) \tag{1}$$

The state $\overline{x}(t) = [x_1(t)\ x_2(t)\ x_3(t)]^\top \in \mathcal{X} = \mathbb{R}^3$ represents the concentration in each of the three compartments. The input $u(t) \in \mathcal{U} \subset \mathbb{R}$ represents the rate of administration of Propofol, and $V_1, k_{ij} \in \mathbb{R}$ $(i, j \in \{1, 2, 3\})$ are patient dependent parameters. Table 1 lists the parameter values determined for a 11-year old child weighing 35 kilograms from the Paedfusor dataset [2].

k_{10}	k_{12}	k_{13}	k_{21}	k_{31}	V_1
0.4436	0.1140	0.0419	0.0550	0.0033	16.044

Table 1: Model Parameters from the Paedfusor dataset.

Automated anaesthesia delivery and anaesthesiologist in the loop We discretise the continuous-time model (1) using a zero-order hold with a time-step of $T_s = 20s$, and presume that the anaesthesiologist input occurs exactly at the sampling instant. The discrete-time approximation of (1) (using parameter values given in Table 1) is

$$\overline{\mathbf{x}}[k+1] = \begin{bmatrix} \mathbf{x}_1[k+1] \\ \mathbf{x}_2[k+1] \\ \mathbf{x}_3[k+1] \end{bmatrix} = \begin{bmatrix} 0.8192 & 0.03412 & 0.01265 \\ 0.01646 & 0.9822 & 0.0001 \\ 0.0009 & 0.00002 & 0.9989 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1[k] \\ \mathbf{x}_2[k] \\ \mathbf{x}_3[k] \end{bmatrix} + \boldsymbol{\omega}[k] + \begin{bmatrix} 0.01883 \\ 0.0002 \\ 0.00001 \end{bmatrix} (v[k] + \boldsymbol{\sigma}[k]) \tag{2}$$

$$= A\overline{\mathbf{x}}[k] + \boldsymbol{\omega}[k] + B(v[k] + \boldsymbol{\sigma}[k]) \tag{3}$$

The automated input is $v[k] \in \mathcal{V}$, and the input from the anaesthesiologist is the random variable $\boldsymbol{\sigma}[k] \in \mathcal{H}$, a finite set that contains all possible dosages the anaesthesiologist can give at the current time instant k. We presume the anaesthesiologist may either choose to not provide a bolus dosage or give a bolus dosage at the rate of 30 mg/min for T_s seconds resulting in a dose of 10 mg being administered during the interval of T_s seconds. The zero-mean Gaussian random vector $\boldsymbol{\omega}[k] \sim \mathcal{N}(0, M)$ with known covariance matrix $M \in \mathbb{R}^{3\times3}$ accounts for the variation in the system model for different patients. State matrices are described by

system matrix $A \in \mathbb{R}^{3\times3}$ and input matrix $B \in \mathbb{R}^{3\times1}$. The bolus dosage ensures that the required concentration of the drug can be achieved and maintained throughout the duration of administration with the help of a trained anaesthesiologist [26]. In this benchmark problem, we choose $\mathcal{V} = [0, 7]$ mg/min, $\mathcal{H} = \{0, 30\}$ mg/min, and $M = 10^{-3}I_3$. The continuous state $\overline{\mathbf{x}}[k] = [\mathbf{x}_1[k]\ \mathbf{x}_2[k]\ \mathbf{x}_3[k]]^\top \in \mathbb{R}^3$ is a random vector due to the stochastic human input $\boldsymbol{\sigma}[k]$ and the noise $\boldsymbol{\omega}[k]$.

Probabilistic model of the anaesthesiologist actions We model the bolus dosage, $\boldsymbol{\sigma}[k]$, as a stochastic process that depends on the current state $\mathbf{x}_1[k]$ and the past values of $\boldsymbol{\sigma}[\cdot]$. We also assume the anaesthesiologist uses a concentration threshold parameter α to decide if a bolus dosage is to be given. Table 2 describes a fictitious anaesthesiologist model, through the probability of applying the bolus dosage under various conditions.

Probability of applying the bolus dosage	Conditions		
0.95	$\mathbf{x}_1[k] > \alpha$	and	0/1 bolus dosage was given in past three minutes
0.90	$\mathbf{x}_1[k] \leq \alpha$	and	0 bolus dosage was given in past three minutes
0.50	$\mathbf{x}_1[k] \leq \alpha$	and	1 bolus dosage was given in past three minutes
0	otherwise		

Table 2: A model of the anaesthesiologist's decision to apply a bolus

2.1.2 Discrete-time stochastic hybrid system model

We write (2) as a discrete-time stochastic hybrid system (DTSHS) [1] by defining the tuple $\mathcal{H}(\mathcal{X}, \mathcal{Q}, \mathcal{V}, T_x, T_q)$, where

1. $\mathcal{X} = \mathbb{R}^3$ is the continuous state space,
2. $\mathcal{Q} = \{0, 1\}^9$ is the discrete state space,
3. $\mathcal{S} = \mathcal{Q} \times X$ is the hybrid state space,
4. $\mathcal{V} = [0, 7]$ mg/min is the admissible controls for the automation,
5. $T_q : \mathcal{Q} \times \mathcal{S} \to [0, 1]$ is a discrete stochastic transition kernel on Q given the current hybrid state $(\overline{q}, \overline{z}) \in \mathcal{S}$ and is defined by (5), and
6. $T_x : \mathcal{B}(\mathcal{X}) \times \mathcal{S} \times \mathcal{V} \to [0, 1]$ is a Borel-measurable stochastic kernel on $\mathcal{X}$ given given the current hybrid state and the current automation action $(\overline{q}, \overline{z}, \overline{v}) \in \mathcal{S} \times \mathcal{V}$ and is defined by (6).

We first define the (normalised) anesthesiologist input augmented with his/her past actions as $\overline{\mathbf{q}}[k] \in \mathcal{Q} = \{0, 1\}^9$, i.e., a binary "counter" that counts the number of bolus dosages given by the user in the past 9 time steps or three minutes ($9T_s = 180s$), since the stochastic process describing the anaesthesiologist actions is not inherently Markov. The dynamics of the discrete

state are described by

$$\overline{\mathbf{q}}[k+1] = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix} \overline{\mathbf{q}}[k] + \begin{bmatrix} \frac{1}{30} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \boldsymbol{\sigma}[k]. \tag{4}$$

The binary counter state $\overline{\mathbf{q}}[k]$ is a random vector since $\boldsymbol{\sigma}[k]$ is a random variable. We incorporate the anaesthesiologist model in Table 2 to define the discrete transition kernel,

$$T_q(\overline{q}', \overline{q}, \overline{z}) = \begin{cases} 0.95 & \overline{q}' = [1\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 \leq 1, \overline{z}_1 > \alpha \\ 0.05 & \overline{q}' = [0\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 \leq 1, \overline{z}_1 > \alpha \\ 0.9 & \overline{q}' = [1\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 = 0, \overline{z}_1 \leq \alpha \\ 0.1 & \overline{q}' = [0\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 = 0, \overline{z}_1 \leq \alpha \\ 0.5 & \overline{q}' = [1\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 = 1, \overline{z}_1 \leq \alpha \\ 0.5 & \overline{q}' = [0\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 = 1, \overline{z}_1 \leq \alpha \\ 1 & \overline{q}' = [0\ \overline{q}_{1:8}]^\top, \|\overline{q}\|_1 = 2 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $\overline{q}_{1:8}$ is the vector $[\overline{q}_1\ \dots\ \overline{q}_8]^\top \in \{0,1\}^8$, and $\overline{z}_1 \in \mathbb{R}$ is the first component of $\overline{z}$. From (4), the first component of $\overline{q}'_1$ is 1 if the current action of the anaesthesiologist $\boldsymbol{\sigma}[k]$ is 30 (apply bolus); otherwise, $\overline{q}'_1$ is 0.

The stochastic continuous state transition kernel, T_x, can be defined using (2) and the probability measure over $(\mathbb{R}^3, \mathcal{B}(\mathbb{R}^3))$. Since $\boldsymbol{\omega}[k] \sim \mathcal{N}(0, M)$, if the current automation input is $\overline{\nu} \in \mathcal{V}$, the current continuous state is $\overline{z} \in \mathcal{X}$, and the discrete state $\overline{q} \in \mathcal{Q}$

$$T_x(\cdot|\overline{q}, \overline{z}, \overline{\nu}) \sim \mathcal{N}(A\overline{z} + B(\overline{\nu} + 30\overline{q}_1), M). \tag{6}$$

2.1.3 Verification problem

Consider a safe set $\mathcal{K} = \{\overline{z} \in \mathcal{X} : 1 \leq \overline{z}_1 \leq 6, 0 \leq \overline{z}_2 \leq 10, 0 \leq \overline{z}_3 \leq 10\} \subset \mathcal{X}$, and a target set $\mathcal{T} = \{\overline{z} \in \mathcal{X} : 4 \leq \overline{z}_1 \leq 6, 8 \leq \overline{z}_2 \leq 10, 8 \leq \overline{z}_3 \leq 10\} \subset \mathcal{X}$.

Problem 2.1.1. ***(Stochastic viability problem)*** *Given an initial state $(q[0], z[0]) \in \mathcal{S}$, compute the maximum probability of guaranteeing $x[k] \in \mathcal{K}$ for $k \in \mathbb{N}_{[0,10]}$, and design an admissible controller that achieves this probability.*

Problem 2.1.2. ***(Stochastic first hitting time reach-avoid problem)*** *Given an initial state $(q[0], z[0]) \in \mathcal{S}$, compute the maximum probability of reaching $\mathcal{T}$ as early as possible while staying in $\mathcal{K}$, and design an admissible controller that achieves this probability.*

2.2 Building Automation System Benchmark

The Building Automation System (BAS) benchmark is a modular and extendable benchmark constructed from a library of stochastic models representing the different components making up a BAS. The library of models is presented in [10] and the reader is referred to the aforecited paper for more details. Using this library of models we setup two instances which aim to address verification and policy synthesis respectively. We denote the first instance as CS1BAS, while the second instance as CS2BAS. For each instance (i) we establish the dynamics of the models, (ii) the specification of interest, and (iii) we describe possible solutions.

2.2.1 CS1BAS: Model

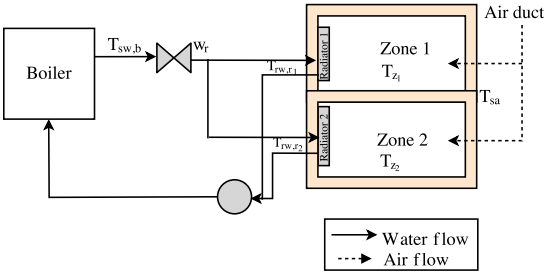


Figure 1: BAS setup for the first case study

We consider two zones, each heated by one radiator and with a common supply air, as portrayed in Figure 1. The model is described as stochastic linear discrete-time model

$$x[k+1] = Ax[k] + Bu[k] + Q + \Sigma W[k] \tag{7}$$

$$y_s[k] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x[k], \tag{8}$$

with a uniform sampling time $\Delta = 15$ minutes. Here, the matrices A, B, Q are properly sized and $\Sigma = diag([(\sqrt{\Delta}\sigma_{z_1})^2\ (\sqrt{\Delta}\sigma_{z_2})^2\ (\sqrt{\Delta}\sigma_{rw,r_1})^2\ (\sqrt{\Delta}\sigma_{rw,r_2})^2])$ encompasses the variances of the process noise for each state. $W = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \end{bmatrix}^T$ are independent Gaussian random variables, which are also independent of the initial condition of the process. The exact matrix values are kept within the ARCH repository.

2.2.2 CS1BAS: Specification

We consider the stochastic safety property: to decide whether traces generated by the models remain within a specified safe set for a given time period. Specifically, this is described using the PCTL property:

$$\Phi := \mathbb{P}_{=p}[\Box^{\leq N=1.5\text{hours}}\ \mathcal{S}] \tag{9}$$

where p is the probability of satisfaction, $\mathcal{S}$ is the safe set is described as an interval around the temperature set-point $T_{SP} = 20^oC\ \pm 0.5^oC$, specifically,

$$\mathcal{S} = \begin{bmatrix} 19.5 & 20.5 \\ 19.5 & 20.5 \\ 19.5 & 20.5 \\ 19.5 & 20.5 \end{bmatrix}.$$

We have defined the acceptable probability of the specification to be true for $p \geq 0.9$.

2.2.3 CS2BAS: Model

In this second case study we focus on the stochastic dynamics of the zone component. The model is a discrete-time model with a sampling time $\Delta = 15$ minutes, and takes the form of

$$x_c[k+1] = A_c x_c[k] + B_c u_c[k] + F_c d_c[k] + Q_c \tag{10}$$

$$y_c[k] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} x_c[k]. \tag{11}$$

Here the continuous state variable x_c is composed of 7 states representing the zone and wall temperatures, the input u_c corresponds of the fan supply rate having a dimension of 1. The disturbance vector d_c has a dimension of 6 which represent external heat sources (weather, occupancy, radiators), while Q_c represents constant additive terms within the model. We model the disturbances as random external effects following Gaussian distributions with a mean μ and variance σ, affecting the room temperature dynamics as $T_{out}[k] \sim \mathcal{N}(\mu = 9, \sigma = 1)$, $T_{hall}[k] \sim \mathcal{N}(\mu = 15, \sigma = 1)$, $CO_{2_i}[k] \sim \mathcal{N}(\mu = 500, \sigma = 100), i \in \{1, 2\}$, $T_{rw,r_i}[k] \sim \mathcal{N}(\mu = 35, \sigma = 5), i \in \{1, 2\}$.

2.2.4 CS2BAS: 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 (i.e $N = 6$). This can be translated into following PCTL specification:

$$\Phi := \mathbb{P}_{=p}[\Box^{\leq N=6} \, |T_{z_1} - T_{SP}| \leq 0.5]$$

over which p is to be maximised for the optimal policy. Here, $T_{SP} = 20^oC$.

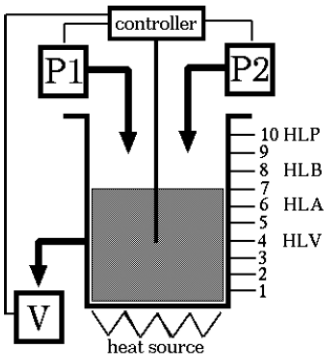


Figure 2: Illustration of the heated tank benchmark (version 5)

2.3 Heated Tank Benchmark

The Heated Tank benchmark is a well-known example problem from the dynamic reliability literature [33, 34, 41]. The system consists of a tank containing liquid whose level is influenced by two pumps and one valve managed by a controller (see Fig. 2). 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.

2.3.1 Continuous-time hybrid-state process model

The general model adopted for this heated tank system is that of piecewise deterministic Markov processes (PDMP). The discrete-valued process θ_t consists of components for the two inflow pumps P_1 and P_2, the outflow valve V, the controller C and the tank T, i.e.

$$\theta_t = \langle \theta_{P_1,t}, \theta_{P_2,t}, \theta_{V,t}, \theta_{C,t}, \theta_{T,t} \rangle.$$

The Euclidean-valued process x_t has two $\mathbb{R}$-valued components, one for liquid height $x_{H,t}$ and another for liquid temperature $x_{T,t}$. These Euclidean-valued components evolve according to the following switching differential equations:

$$\dot{x}_{H,t} = (\chi_{P_1,t} + \chi_{P_2,t} - \chi_{V,t}) \cdot q \qquad \text{with initial condition } x_{H,0} = H_{init} \text{ and}$$
$$\dot{x}_{T,t} = ((\chi_{P_1,t} + \chi_{P_2,t}) \cdot (T_{in} - x_{T,t}) \cdot q + E_{in}) / x_{H,t} \qquad \text{with initial condition } x_{T,0} = 15\,{}^2\!/_3\,^\circ\text{C}$$

where $\chi_{U,t}$ indicates if unit $U \in \{P_1, P_2, V\}$ is working or not (i.e. $\chi_{U,t} = 1$ if unit U is working and 0 otherwise), q is the flow parameter, $T_{in} = 15\,^\circ$C is the inflow temperature, and $E_{in} = 1 \frac{^\circ\text{Cm}}{h}$ is the heat source parameter. The differential equation for $x_{T,t}$ is from [41].

The (rare) event probabilities to be estimated on a finite time interval $[0, t_{end}]$ are:

- the dryout probability $\mathbb{P}(\exists\, t \in [0, t_{end}] \colon x_{H,t} \leq H_{dryout})$,
- the overflow probability $\mathbb{P}(\exists\, t \in [0, t_{end}] \colon x_{H,t} \geq H_{overflow})$, and
- the overheating probability $\mathbb{P}(\exists\, t \in [0, t_{end}] \colon x_{T,t} \geq 100\,^\circ\text{C})$.

Versions	q	$H_{overflow}$	H_{high}	H_{init}	H_{low}	H_{dryout}
1-3	$0.6\,\frac{\mathrm{m}}{\mathrm{h}}$	3 m	1 m	0 m	−1 m	−3 m
4	$0.6\,\frac{\mathrm{m}}{\mathrm{h}}$	5 m	1 m	0 m	−1 m	−5 m
5	$1.5\,\frac{\mathrm{m}}{\mathrm{h}}$	10 m	8 m	7 m	6 m	4 m

Table 3: Parameter values for different versions of the heated tank benchmark

The evolution of the two $\mathbb{R}$-valued state components $x_{H,t}$ and $x_{T,t}$ depends on the evolution of the indicator processes $\chi_{P_1,t}$, $\chi_{P_2,t}$ and $\chi_{V,t}$. Codetta-Raiteri [11] identified five different versions for the evolution of these indicator processes and the parameters. The values used across the five versions for the flow and level parameters are listed in Tab. 3. We describe all other differences between the five versions in the following subsections.

2.3.2 Version 1: Baseline

Each of the three units P_1, P_2 and V has the following set of four discrete modes:

$$\Theta_U = \{On, Off, StuckOn, StuckOff\}.$$

The initial condition is

$$\theta_{P_1,0} = On \wedge \theta_{P_2,0} = Off \wedge \theta_{V,0} = On.$$

The switching from *On* to *Off* and vice-verse is managed by the controller.

In addition to this switching, there are four exponentially distributed failure switching possibilities: from *On* to *StuckOn*, from *On* to *StuckOff*, from *Off* to *StuckOff*, and from *Off* to *StuckOn*. Each of these failure transitions happens at rate $\hat{\lambda}_{P_1} = 1/438\,\mathrm{h}$ for P_1, at rate $\hat{\lambda}_{P_2} = 1/350\,\mathrm{h}$ for P_2, and at rate $\hat{\lambda}_V = 1/640\,\mathrm{h}$ for V.

The controller mode process $\theta_{C,t}$ switches between the following three configurations:

$$\Theta_C = \{Normal, Increase, Decrease\}.$$

The initial condition of the controller is $\theta_{C,0} = Normal$. If $x_{H,t} \leq H_{Low}$ then $\theta_{C,t}$ switches from *Normal* or *Decrease* to *Increase*. If $x_{H,t} \geq H_{High}$ then $\theta_{C,t}$ switches from *Normal* or *Increase* to *Decrease*. The low and high liquid level parameter values are $H_{Low} = 6\,\mathrm{m}$ and $H_{High} = 8\,\mathrm{m}$.

The control law of the controller is a function of $\theta_{C,t}$. If $\theta_{C,t} = Normal$ then the controller does not try to influence the pumps and valve. If $\theta_{C,t} = Increase$ then the controller aims to increase the height of the liquid in the tank, by switching both pumps on, and by switching the valve off. If $\theta_{C,t} = Decrease$ then the controller aims to decrease the height of the liquid in the tank, by switching both pumps off, and by switching the valve on. The switching by the controller has no effect on unit U if it is in failure mode *StuckOn* or *StuckOff*.

2.3.3 Version 2: Mode-dependent Failure Rates

Compared to version 1, the failure rates now depend on the current mode of the respective component, and the two different kinds of failures occur with different rates. The following modifications are made to the rates: For P_1, the rate to move from *Off* to *StuckOn* is $20 \cdot \hat{\lambda}_{P_1}$ and the rate to move from *Off* to *StuckOff* is $200 \cdot \hat{\lambda}_{P_1}$. For P_2, the rate to move from *On* to *StuckOn* is $20 \cdot \hat{\lambda}_{P_2}$ and the rate to move from *On* to *StuckOff* is $200 \cdot \hat{\lambda}_{P_2}$. For V, the rate to move from *Off* to *StuckOn* is $20 \cdot \hat{\lambda}_V$ and the rate to move from *Off* to *StuckOff* is $200 \cdot \hat{\lambda}_V$. All other rates are multiplied by a factor of 2.

Method	References	v1	v2	v3	v4	v5
Conditional MC	[34]	Yes	Yes	Yes	Yes	Yes
PDMP-based MC	[48]	Yes	–	–	–	Yes
Fluid stochastic Petri net & MC	[12, 13]	Yes	Yes	Yes	Yes	Yes
Stoch. activity network & MC	[11]	Yes	Yes	Yes	Yes	Yes
Dynamic Bayes network	[14]	Yes	Yes	Yes	Yes	–
Advanced RESTART simulation	[42]	–	–	–	Yes	–

Table 4: Methods previously applied to the heated tank benchmark

2.3.4 Version 3: Controller Failure on Switching

Relative to version 1, each time the controller's law is to trigger a switching of one or more of the units P_1, P_2 and V, it is assumed that this controller action will be realized with a probability of 90 % only. This means that there is a 10 % probability that a required switch is not implemented for P_1, P_2 and V.

2.3.5 Version 4: Controller-Triggered Repairs

Relative to version 1, it is assumed that the control laws under *Increase* and *Decrease* include triggering a repair of any unit (P_1, P_2 or V) if it cannot be switched. The duration of such a repair is assumed to be exponentially distributed, with a mean of 5 hours. Upon repair of unit U, the controller implements the desired switching for this unit.

2.3.6 Version 5: Temperature-Dependent Failure Rates

Relative to version 1, the static failure rates $\hat{\lambda}_{P_1}$, $\hat{\lambda}_{P_2}$ and $\hat{\lambda}_V$ are replaced by temperature-dependent failure rates $\hat{\lambda}_{P_1} \cdot \alpha(x_{T,t})$, $\hat{\lambda}_{P_2} \cdot \alpha(x_{T,t})$ and $\hat{\lambda}_V \cdot \alpha(x_{T,t})$, respectively, where

$$\alpha x_{T,t} = \left(b_1 \cdot \mathrm{e}^{b_c \cdot (x_{T,t} - 20)} + b_2 \cdot \mathrm{e}^{-b_d \cdot (x_{T,t} - 20)} \right) / \left(b_1 + b_2 \right) \tag{12}$$

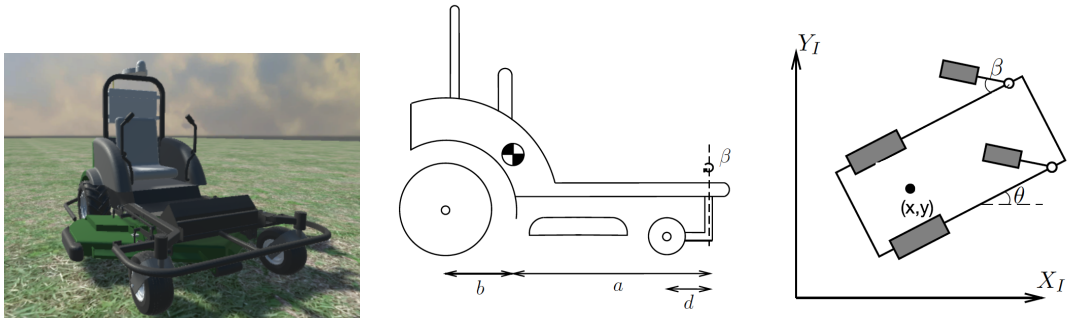
with parameter values $b_1 = 3.0295$, $b_2 = 0.7578$, $b_c = 0.05756$ and $b_d = 0.2301$. Note that, in order to resolve a typo in [11], for the above equation we followed [41].

2.3.7 Results from the Literature

Because the heated tank benchmark has been well-addressed in the literature, for each of the five versions top event probabilities have been assessed. We give an overview of which modelling and analysis methods have successfully been applied to which of the versions in Tab. 4, where "MC" refers to Monte Carlo simulation.

2.4 Lawn Mower Benchmark

Here we describe dynamics of a lawn mower adapted from [20]. A conceptual model of the driverless lawn mower is shown in Fig. 3(a). The lawn mower is steered by controlling the velocity separately on the two driving wheels. A sketch of the vehicle is shown in Fig. 3(b) and its chassis and caster wheels are shown in Fig. 3(c).



(a) (b) (c)

Figure 3: Lawn mower schematics [20]

2.4.1 Dynamical model

The dynamic response of the lawn mower can be described by a 6-dimensional nonlinear stochastic difference equation

$$x[k+1] = f(x(k), v(k), w(k)), \quad k = 0, 1, 2, \ldots \tag{13}$$

where $x(\cdot) \in \mathbb{R}^6$ is the state, $v(\cdot) = [V_r(\cdot), V_l(\cdot)]^T \in \mathbb{R}^2_{\geq 0}$ is the input, and $w(\cdot) \in \mathbb{R}^2$ is the noise. The vector field $f : \mathbb{R}^6 \times \mathbb{R}^2_{\geq 0} \times \mathbb{R}^2 \to \mathbb{R}^6$ is defined by the following elements

$$\begin{aligned}
f_1(x,v,w) &= x_1 + \frac{T_s}{I_{zz}}\left(A_{fy}\sin\left(x_6 + \frac{\pi}{2}\right)a + 2C_\gamma \tan\left(\frac{2(x_2 - bx_1)}{(V_r + V_l)}\right)b\right), \\
f_2(x,v,w) &= x_2 + \frac{T_s}{M}\left(A_{fy}\sin\left(x_6 + \frac{\pi}{2}\right) - 2C_\gamma \tan\left(\frac{2(x_2 - bx_1)}{(V_r + V_l)}\right)\right) - \frac{T_s(V_r + V_l)}{2}x_1, \\
f_3(x,v,w) &= x_3 + \frac{T_s(V_r + V_l)}{2}\cos(x_5), \\
f_4(x,v,w) &= x_4 + \frac{T_s(V_r + V_l)}{2}\sin(x_5), \\
f_5(x,v,w) &= x_5 + T_s\left(\frac{V_r - V_l}{2D} + x_1\right) + w_1, \\
f_6(x,v,w) &= x_6 - \frac{T_s(V_r + V_l)}{2d}\cos(x_6) - \frac{T_s(V_r - V_l)(d + \sqrt{D^2 + (a+b)^2}\sin(x_6))}{2Dd} + w_2,
\end{aligned} \tag{14}$$

Mass, M	Axle half width, D	Dimensions, (a, b)	Caster wheel offset, d	A_{fy}	I_{zz}	C_γ
$700kg$	$1.2m$	$0.35m$, $0.15m$	$0.1m$	199	$394Kg\text{-}m^2$	22000

Table 5: Lawn mower parameters

where x_1, x_2, x_3, x_4, x_5, and x_6 are respectively yaw, yaw rate, x, y, angle of vehicle with respect to x-axis (θ), and relative orientation of caster wheel (β), respectively. The w_1 and w_2 are noises in orientation and angle of caster wheel generated due to uneven ground surface. V_r and V_l are the control inputs representing velocity of right and left driving wheels, respectively. Parameter T_s is the sampling time. The remaining parameters are listed in Table 5.

2.4.2 Controller mechanism

The main objective for the lawn mower is to cover a given area as soon as possible. This can be seen as a path planning problem and is usually achieved by specifying a set of way-points, and the task of the controller would be to move from an initial position in the vicinity of the current way-point to the vicinity of the next way-point. We provide simple On-Off type controller for the lawn mover to reach the desired position (x_d, y_d). Consider base velocity V_b and boost velocity V_d. The control inputs that is velocities of right and left wheel are given as

$$\begin{aligned} V_r &= V_b + V_d \text{sgn}(\sin(\theta_d - x_5)) \\ V_l &= V_b - V_d \text{sgn}(\sin(\theta_d - x_5)), \end{aligned} \tag{15}$$

where $\theta_d = \text{atan2}((y_d - x_4), (x_d - x_3))$ and atan2 is the four-quadrant inverse tangent.

2.4.3 Specification of interest

The controller of the previous subsection does not take into account avoiding obstacles and preventing collision with boundaries of the area. Here we define a specification that computes the probability of avoiding obstacles over finite-time traces of the system.

The state space of the system is $X = \mathbb{R}^6$. Consider the desired location $(x_d, y_d) = (-10, -5)$ with selection of $V_b = 2$ and $V_d = 1.5$ We also consider the region $X_0 = [-0.5, 0.5]^2 \times [-1, 1]^2 \times [-\pi, \pi]^2$ for the initial state of the system and $X_1 = [-0.5, 0.5]^2 \times [-4, 4] \times [2.5, 4] \times [-\pi, \pi]^2$ for the obstacles or one of the boundaries of the area. The set of atomic propositions is given by $\Pi = \{p_0, p_1, p_2\}$ with labeling function $L(x_i) = p_i$ for all $x_i \in X_i$, $i \in \{0, 1, 2\}$, with $X_2 := X \backslash (X_0 \cup X_1)$. The objective is to compute a lower bound $\alpha > 0$ on the probability that the solution process of length $N = 200$ satisfies the safe LTL$_f$ formula $p_0 \wedge \Box^{\leq 200} \neg p_1$:

$$\Phi := \mathbb{P}_{\geq \alpha}[p_0 \wedge \Box^{\leq 200} \neg p_1].$$

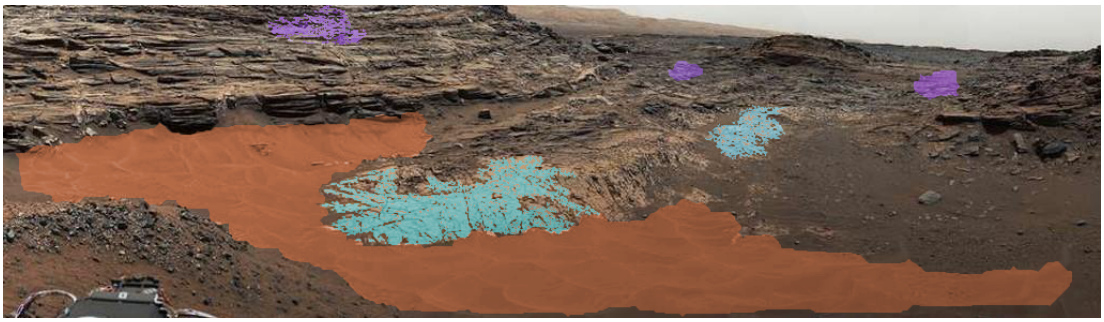


Figure 4: Labelled map of a campaign where blue and purple mark regions containing science targets for the rover and orange area shows the region in sand.

2.5 Mars Rover Benchmark

We introduce the problem of planetary exploration with a two-agent team composed of a Mars rover and a Mars helicopter, tasked to collect a set of samples from the Mars surface. Due to the high cost and risk associated with these missions, it is critical to have strong correctness and performance guarantees on the robots' behaviours. To aid in this objective, we give a mission specification framework and formally encode complex mission specifications used for real Mars rover operation. The logic of choice is syntactically co-safe Linear Temporal Logic (scLTL) defined over predicates in belief space [28, 43]. This approach allows us to capture temporal and uncertainty constraints ranging from science campaign (e.g., getting science results from specific regions) to operational constraints such as maintaining certain accuracy over the vehicles' state and the environment, inter-agent distance, and energy level.

For a first and more accessible case study, we consider a simplified version of the full specification set. Additionally, the rover is assumed to have stochastic motion and perception, and is tasked with collecting samples from a partially-known deterministic environment. The helicopter provides support by exploring the environment on-demand. Exploration is necessary if insufficient knowledge is available to synthesise a control policy with sufficient success probability.

2.5.1 Scientific Mars Mission

The set of requirements below specify a potential science campaign. We say that actions taken by the rover or copter have an associated risk (r) of a certain (hidden) event, the requirements impose bounds on the risks associated to specific events that the Mars rover copter mission can take.

1. *Rover must get a type A sample from a type A target and it must get one type B sample of a science target of type B.* Areas that potentially include either type A or B are shown in purple and blue, respectively, in Fig. 4. They represent areas for potentially high-science-value rock/soil samples that need to be collected.

2. *Rover must maintain the variance of its state below* Σ_{max}. This specification is to maintain accuracy along the mission, and reduce the risk of unexpected events.

3. *Rover routes must avoid hazards.* The rover can only take moderate risks when planning its route along hazard areas such as large rocks, deep sands, steep slopes, which are given as labelled regions on the map as part of the science campaign.

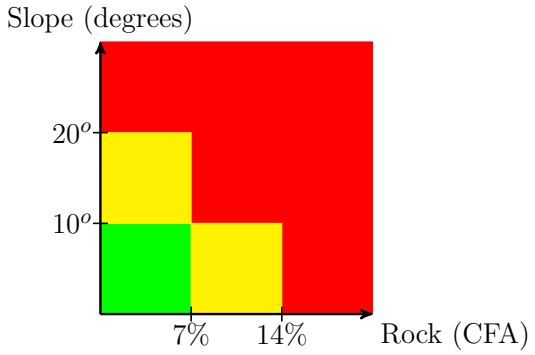


Figure 5: Speed Constraints for Slope vs Rock CFA (Cumulative Fractional Area). High speed (green), Low speed (yellow) and Stop (red).

4. *Rover must follow speed constraints.* Rover's maximum speed is a function of the cumulative fractional area of rocks, and the terrain slope, see Fig. 5.

5. *Mars helicopter must avoid landing near hazards.* These hazards include, inter alia, large rocks, deep sands, and steep slopes. More precisely, the helicopter cannot land somewhere, where it has a risk probability greater than r_5 of landing within x_5 meters of a hazard zone.

6. *Helicopter must maintain a minimum distance d_{min} and a maximum distance d_{max} from the rover.* The purpose of this specification is to protect the primary asset of the mission: the rover, which carries all vital science instruments. The helicopter should not pose any risk for the rover in case of helicopter failure, but should also maintain communication range.

2.5.2 Introduction: A formal Belief space planning problem

In this section, we define the planning problem under uncertainty and in partially-observable environments. We start by reviewing Partially-Observable Markov Decision Process (POMDP) and associated synthesis problem. Then, we present the belief-based linear temporal logic (LTL) properties to be used for mission planning.

POMDP. Let us denote the system state, action, and observation at the k-th time-step by $x[k], u[k], z[k]$, respectively. Let

$$x[k+1] = g\left(x[k], u[k], w[k]\right) \tag{16}$$
$$z[k] = h\left(x[k], v[k]\right) \tag{17}$$

denote the system dynamics and measurement models, where $w[k] \sim p_w\left(\cdot|x[k]\right)$ and $v[k] \sim p_v\left(\cdot|x[k]\right)$ denote the state-dependent process and observation noise. Due to the observation noise, the best one can infer about the system state is a probability distribution over all possible states, referred to as *belief* $b[k] = p(x[k]|\mathbf{i}[k])$ with history $\mathbf{i}[k] := \{(u, z)_{0:k}\}$. The belief space (i.e., the set of all beliefs) is denoted by $\mathbb{B}$. Belief is typically evolved using a recursive filter denoted by τ as

$$b[k+1] = \tau(b[k], u[k], z[k+1]). \tag{18}$$

As is common, one can generally assume that the belief space is a Polish space[2] and that the recursive filter is a Borel measurable mapping.

Belief-space LTL. Let $\mathbf{b} = b[0]\, b[1]\, b[2] \ldots \in \mathbb{B}^\omega$ be a belief sequence, and AP be finite set of atomic propositions. Each element $f \in AP$ is a predicate over the belief space $\mathbb{B}$, and we say that a belief b satisfies f if $f(b) = \top$, where $\top$ is the truth Boolean value. Moreover, the predicate functions f must be Borel measurable mappings. Using the set of belief-based atomic proposition AP, we defined mission specifications using scLTL, which has as syntax

$$\phi := \top \mid \ f \ \mid \ \neg f \ \mid \phi_1 \wedge \phi_2 \mid \phi_1 \,\mathcal{U}\, \phi_2 \mid \ \bigcirc \phi$$

For convenience, we use the additional operators: $\phi_1 \vee \phi_2 \equiv \neg(\neg\phi_1 \wedge \neg\phi_2)$, $\Diamond\phi \equiv \top \,\mathcal{U}\, \phi$, and $\Box\phi \equiv \neg\Diamond\neg\phi$, where $\equiv$ denotes semantic equivalence.

Control synthesis problem. The control synthesis problem for the POMDP with belief-based LTL specifications can be formulated as follows:

Problem 2.5.1. *Let ϕ be a given LTL formula and let the robot evolve according to dynamics (16), with observation dynamics (17), and using a Bayesian filter defined by (18). Find a policy μ^* such that*

$$\mu^* = \arg\max_{\mu \in \mathbb{M}} Pr[\mathbf{b} \models \phi] \quad \textit{subject to (16), (17), (18).} \tag{19}$$

2.5.3 POMDPs modeling rover, helicopter, and environment

In this section, we define the models for the Mars rover robot and helicopter robot, and the Martian environment, where the two robots must perform science campaigns such as the one described in Sec. 2.5.1.

The state of this combined system model is denoted as $x = (x^r, x^c, m)$ and is comprised of the rover state x^r, the helicopter state x^c, and the unknown environment state m. Each of these states is not exactly known. Due to the noise in the robots' motion and perception, the best one can infer about the overall system state is its belief, i.e., the probability distribution over all possible states, denoted by $b[k] = p(x[k] \,|\, \mathbf{i}[k])$, where the history is given composed of the past actions u and the past observations z, that is, $\mathbf{i}[k] := \{(u, z)_{0:k}\}$.

Based on joint belief over rover, helicopter, and the environment, b we can compute three separate marginalised beliefs for each of the models as

$$\begin{aligned} b^r &= p(x^r|\mathbf{i}[k]) \quad &&= \textit{rover position distribution} \\ b^c &= p(x^r|\mathbf{i}[k]) \quad &&= \textit{copter position distribution} \\ b^m &= p(m|\mathbf{i}[k]) \quad &&= \textit{environment belief} \end{aligned} \tag{20}$$

Rover model. The kinematics of the rover can be described using the special Euclidean group of planar motions, i.e. $SE(2)$. The rover's state $x^r = (p^r, v^r)$ is a combination of its pose parameterised as $p^r \in \mathbb{R}^2 \times [-\pi, \pi)$ and its velocity $v^r \in \mathbb{R}^3$. The dynamics of the rover can be represented by a unicycle model

$$\begin{bmatrix} x[k+1] \\ y[k+1] \\ \varphi[k+1] \end{bmatrix} = \begin{bmatrix} x[k] + v\cos(\varphi[k]\delta k \\ y[k] + v\sin(\varphi[k]\delta k \\ \varphi[k] + \omega\delta k \end{bmatrix} + w, \tag{21}$$

where v (velocity) and ω (angular velocity) are the inputs and $w \sim \mathcal{N}(0, Q)$ is Gaussian process noise.

[2] With Euclidean spaces as a typical example, a Polish space is a separable and completely metrizable topological space.

Helicopter model. Similarly to the rover, the helicopter's kinematic are given via $SE(3)$ and its state $x^c = (p^c, v^c)$ includes the pose parameterised as $p^c \in \mathbb{R}^3 \times [-\pi, \pi)^2 \times [0, \pi]$, and velocity $v^c \in \mathbb{R}^6$.

Simplified rover and copter belief models For simplicity and practical implementation, we will assume that the belief about the state of the rover's and helicopter's pose and velocity are maintained using extended Kalman filters [29]. Thus, the beliefs are normal distributions parameterised by mean and covariance, i.e., $b^r \equiv (\hat{x}^r, \Sigma^r)$ and $b^c \equiv (\hat{x}^c, \Sigma^c)$, respectively.

Environment and sensor model. The Martian environment contains N regions of type $A, B, ...$ More specifically, every region is labeled by a single type indicating whether it could potentially include either samples of that type, or hazards in case of hazard typed regions. Thus the state of the environment comprises of $0, 1$ values for each of the regions, 0 indicates the absence of the type whereas 1 signifies its presence. Therefore the set of states of the environment is given as $\mathcal{R} := \{0, 1\}^N$ with elements $m \in \mathcal{R}$. Over the time of a Mars mission the unknown environment state remains constant.

Let the $i-$th region be represented by a bounded connected set $R^i \subset \mathbb{R}^2$, and assume that R^i and R^j are disjoint sets iff $i \neq j$.

Furthermore, we use the notation R_A to indicate the collection of sets of type A and R_B the collection of sets of type B.

At every time instant the rover, or helicopter, can decide to make an observation of one of the regions. For the i-th region, we can obtain measurements $z^i \in \{0, 1\}$ of the semantic labels over the environment by running classifiers on the images collected from the camera on the rover. The false rate of these measurement is as follows:

$$p(z^i \neq x^i_{true}) = \frac{1}{2d_{max}} \min\{d^i, d_{max}\} \tag{22}$$

where d^i is the relative distance to the $i-$th region and where $x^i_{true} \in \{0, 1\}$ is the true, but unknown, state of that regions. Where d_{max} is the largest distance from which you still get informative images.

2.5.4 Mission Specification in LTL

In this section, we define the LTL predicates and properties corresponding to the mission specifications described in Sec. 2.5.1.

The overall specification ϕ for the campaign executed over the span of T time units:

$$\phi := \Box^{\leq T}(f_{uncert} \wedge \phi_{safety} \wedge \phi_{operational}) \wedge \phi_{science} \tag{23}$$

which expresses the mission where the science objective $\phi_{science}$ needs to be satisfied and we need to maintain a desired level of accuracy, safety, and operational constraints until the end of the mission as expressed by f_{uncert}, ϕ_{safety}, and $\phi_{operational}$, respectively. The operator $\Box^{\leq T}$ denotes the bounded time invariance.

Science: The science objective in this campaign is to collect one sample from each of the high-valued science areas A and B. The rover can obtain a sample from a region once it has entered the region and if the sample type is present in the region. Thus we can specify its behavior next. Consider the following predicate to indicate the position of the rover with respect region i and risk ϵ given as

$$f_{i,\epsilon}(b^r) := \begin{cases} 1 & \text{if } \mathbb{P}(p^r \in R_i \times [-\pi, \pi]) \geq \epsilon \\ 0 & \text{else.} \end{cases}$$

Furthermore, we denote a predicate associated to the observation of the i-th region as $(z^i = 1)$. Thus we finally have the mission specification as

$$\phi_{science} := \Diamond^{\leq T} \bigvee_{i \in A} \big((z^i = 1) \wedge f_{i,1-\epsilon}(b^r)\big) \wedge \Diamond^{\leq T} \bigvee_{i \in B} \big((z^i = 1) \wedge f_{i,1-\epsilon}(b^r)\big). \tag{24}$$

Uncertainty: The uncertainty objective is to maintain the desired level of accuracy on the state estimates.

$$\begin{aligned} f_{uncert} &:= tr(\Sigma^r) < \Sigma^r_{max} \text{ with } b^r \equiv (\hat{x}^r, \Sigma^r), \\ f_{uncert} &:= tr(\Sigma^c) < \Sigma^c_{max} \text{ with } b^c \equiv (\hat{x}^c, \Sigma^c). \end{aligned} \tag{25}$$

Safety: Space applications fall into the category of safety-critical applications. Thus, maintaining the safety of systems during operation is a paramount objective of the mission. We express the safety specification as:

$$\phi_{safety} := \phi_{keepout} \wedge f_{near}. \tag{26}$$

The keepout part of the specification is given as follows. The distance from risky regions in the map such as non-traversable obstacles, sandy areas with a risk of getting stuck, cliffs, and high-slope areas, is captured by property $\phi_{keepout}$. We define keep-out constraints for the rover:

$$\phi_{keepout} := \bigwedge_{i \in O} (b^m_i \geq 0.05) \rightarrow \neg f_{(i,0.05)}(b^r) \tag{27}$$

where b^m_i is marginalised belief in the i-th region.

3 Participating Tools & Frameworks

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

Barrier Certificates The paper [27] provides a systematic approach of using barrier certificates for algorithmic verification of stochastic systems against a wide class of temporal properties. It provides a method for computing lower bounds on the probability that a discrete-time stochastic system satisfies a given *safe LTL specification* over a *finite* time horizon. This is achieved by first decomposing the specification into a sequence of simpler verification tasks based on the structure of the automaton associated with the negation of the specification. Then it uses barrier certificates for computing probability bounds for simpler verification tasks, which are further combined to get a lower bound on the probability of satisfying the original specification. Computation of barrier certificates can be performed for polynomial dynamics using sum-of-squares optimizations.

FAUST^2 The tool *Formal Abstractions of Uncountable-STate STochastic processes* (FAUST^2) [39] 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 [31] or STORM [16]. Alternatively, it can handle internally the computation of 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 is available at `https://sourceforge.net/projects/faust2/`

FIRM-GDTL The feedback information roadmap tool for Gaussian distribution temporal logics supports the path planning of partially observable robots. The python-based toolbox can handle several kinds of robots The tool is available at `https://github.com/wasserfeder/gdtl-firm/tree/dev-mars-fsa`

Modest Toolset The Modest Toolset [25] supports the modelling and analysis of hybrid, real-time, distributed and stochastic systems. A modular framework centred around the stochastic hybrid automata formalism [24] and supporting the JANI specification [8], 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 PA/MDP (probabilistic automata/Markov decision processes). The toolset can can be obtained from `http://www.modestchecker.net/`. For the experiments on the Heated Tank benchmark, we used the Modest Toolset's simulator "modes" [7] 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) [6].

SDCPN modelling & MC simulation Stochastically and Dynamically Coloured Petri Nets (SDCPN) [18, 19] have been developed in support of the compositional modelling of any Generalized Stochastic Hybrid Automaton [9]. 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].

SReachTools A MATLAB toolbox to tackle various problems in stochastic reachability. It aims to support the following problems: Stochastic reach-avoid problem [40, 1] (guaranteeing safety for stochastic systems), Lagrangian methods-based underapproximation [22], Fourier transforms-based underapproximation [46, 47], and forward stochastic reachability [45] (characterizing the stochasticity of the state at a future time of interest). The tool is available at https://unm-hscl.github.io/SReachTools/.

4 Results

We present the results obtained for each benchmark using the tools highlighted in Sec.3. Given the different modelling structures, the dissimilar stochastic elements and that most of the tools and frameworks are tailored to a specific task, not all tools or frameworks could be applied to each benchmark. Consequently, in this section we present the results of the specific tools and frameworks when a solution can be computed.

4.1 Anaesthesia benchmark

4.1.1 Verification and controller synthesis using SReachTools

We solve Problem 2.1.1 by synthesising an open-loop controller as well as a polytopic underapproximation of the true stochastic reach-avoid set. Figure 6 shows the obtained 2-dimensional underapproximative polytope ($\mathbf{x}_3[0] = 5$) of the stochastic reach-avoid set for a probability threshold of 0.99 using 10 direction vectors. This computation took $\sim$ 77 seconds on an Intel Xeon CPU with 3.4GHz clock rate and 32 GB RAM. Figure 7 shows the Monte-Carlo simulation-based validation of the open-loop controller from one of the vertices.

The SReachTools code to solve Problem 2.1.1 is available on [44].

4.1.2 Verification using FAUST^2

We perform probabilistic reachability analysis for the specification,

$$\Phi := \mathbb{P}_{=p}[\Box^{\leq N}\, \mathcal{S}]$$

with

$$\mathcal{S} = \begin{bmatrix} 1 & 6 \\ 0 & 10 \\ 0 & 10 \end{bmatrix}.$$

We perform this task using FAUST^2, first over a 1 step time horizon and then for 5 times steps. In both instances, we make use of an open-loop control policy with the input discretised between [07]. Fig. 4.1.2 shows the obtained partition together with the optimal safety probability. In this case, the abstraction only required 76832 states for an overall abstraction error of 0.1613

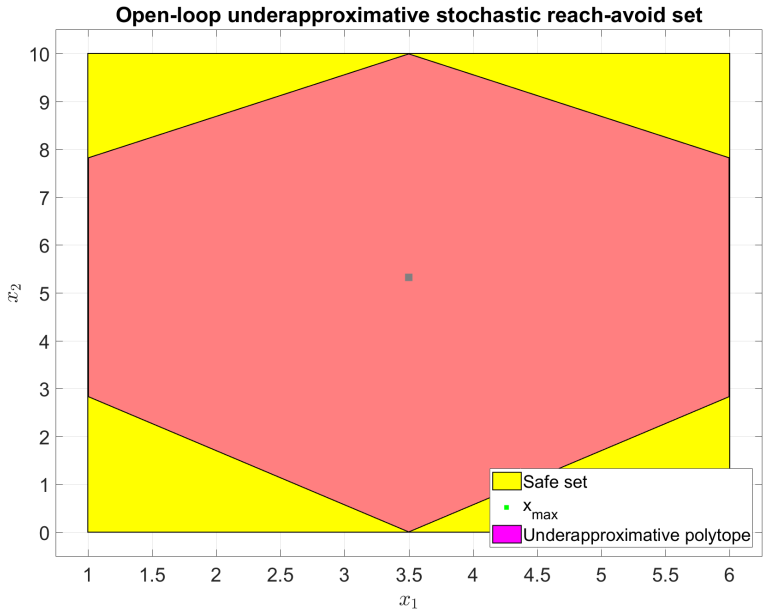


Figure 6: Polytope containing a subset of the initial states from which an open-loop controller exists that solves Problem 2.1.1.

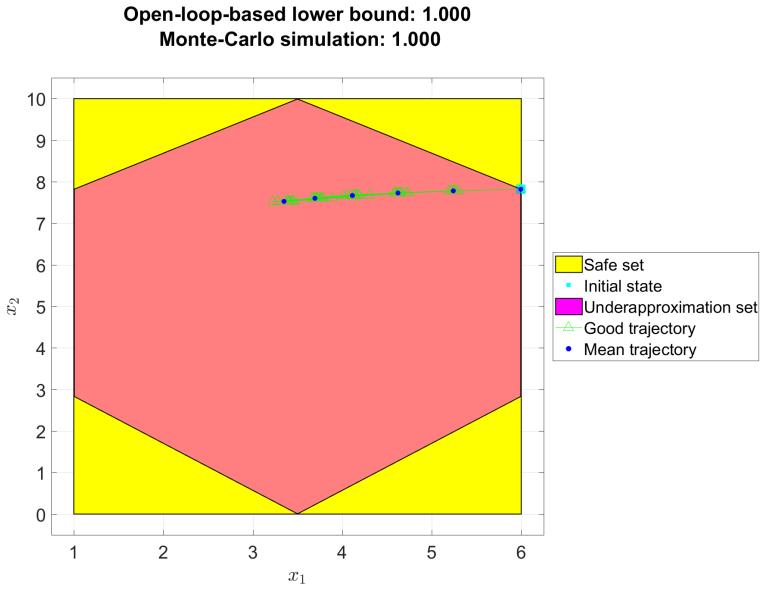
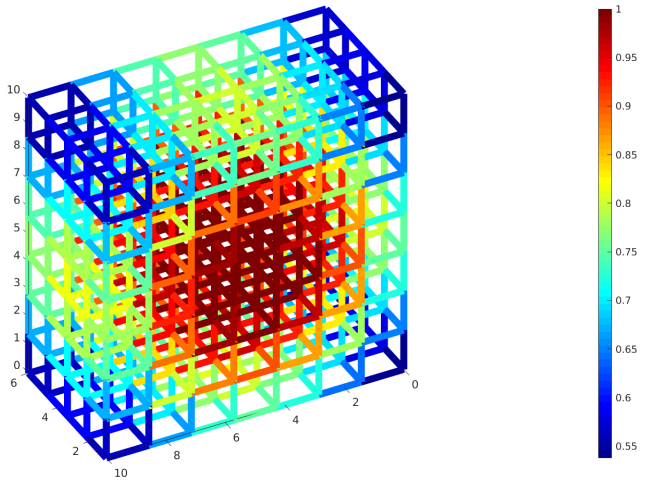
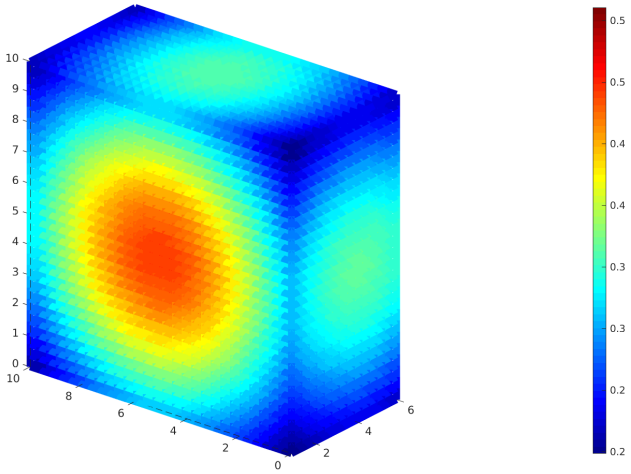


Figure 7: Validating the open-loop controller at a particular vertex of the polytope using Monte-Carlo simulation



(a) N = 1



(b) N = 5

Figure 8: Partition of the safe set along with the optimal safety probability after N time steps

and took 46.6 seconds to be generated. We obtain a maximal probability of 1. Fig. 4.1.2 shows the obtained partition together with the optimal safety probability. In this case, the abstraction required 104162436 states for an overall abstraction error of 0.2281 and took 818.0 seconds to compute. We obtain a maximal probability of 0.5607. In both instances, we can note that the structure of the partitions correspond to the results obtained using SReachTools. They also highlight however the current challenge in performing such abstractions. Namely, that the number of states generated using the abstraction process is a function of the required abstraction error and the time horizon of property of interest. It is also a function of the variance associated with the process noise of the underlying dynamics of the stochastic model. In this case, the model has a diagonal covariance matrix with a value of 5.

Next, we generate the optimal policy for the given abstractions. The input has been discretised into two points: [3.5 7]. We result with an optimal policy in the form of a look up table which describes the optimal action to take for each grid point and time horizon. We show the corresponding plot for a time horizon $N = 1$ in Fig. 4.1.2.

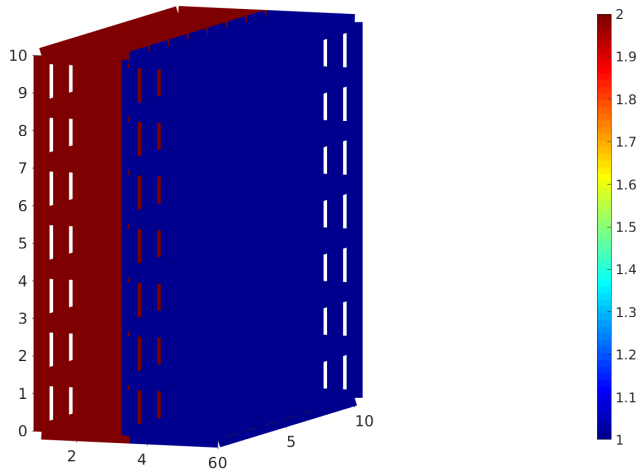


Figure 9: Input action lookup table at $N = 1$

4.2 Building Automation System benchmark

4.2.1 CS1BAS: Results

The results obtained using the individual tools are presented hereunder.

Verification using FAUST^2 Here, we compute the optimal safety probability for the safe set described in within the PCTL property described using (9) by applying FAUST^2. Here, FAUST^2 abstracts the safe set into uniform partitions over which the safety probabilities are given. The obtained results are depicted in Fig. 10. The abstraction and computation of the value functions took 21.08 seconds to be generated. The abstraction consists of 1444 representative points for the continuous variables, while it consists of 25 representative points of the input.

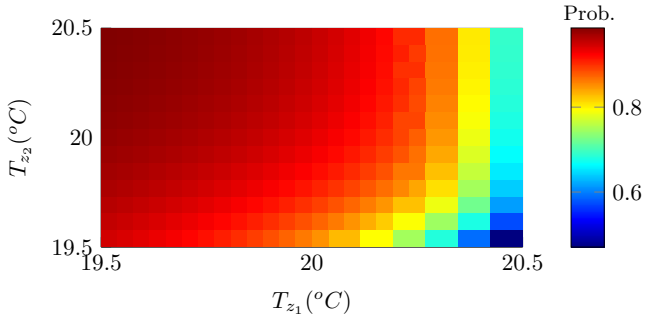


Figure 10: Partition of the safe set for model M_s, along with optimal safety probability for each partition set

4.2.2 CS2BAS: Results

For the model described using (11) and the given specification, we aim to synthesise a policy that maximises the safety probability p. This synthesis goal can be computationally hard due to the number of continuous variables making up (11).

Verification using FAUST^2 In order to make use of the FAUST^2, we perform policy synthesis via abstractions and (ε, δ) simulation relations [23]. The (ε, δ) pair providing the optimal trade off is obtained using an abstract model with 1 continuous variable and corresponds to (0.28592, 10^{-2}). Next, we use FAUST^2 to perform a grid-based computation of the safety probability on the abstract model and obtain a model of size 5943640 with an overall accuracy of 0.15. Over this approximation we synthesise the optimal policy for the abstract model which results in a safety probability of $p' = 1$. We refine the obtained policy [23] such that it can be applied to the original model (11). The overall process results in Φ being satisfied with a safety probability of $p = p'\eta - N\delta = 0.7900$, where η is the abstraction error introduced by FAUST^2. The total computational time taken to obtain the optimal policy was 185.65 seconds.

4.3 Heated Tank benchmark

Versions 1 through 4 of the Heated Tank benchmark have been evaluated using both the Modest and the SDCPN based MC simulation approach. On this basis we identify similarities and differences regarding:

- understanding of the Heated tank benchmark versions
- formal modelling of the Heated tank versions
- estimated probabilities for dryout and overflow

4.3.1 Understanding of the Heated Tank benchmark versions

The understanding of the five versions of the Heated tank benchmark created by far the largest challenge. There appeared to be various types of ambiguities on model details. There were three sources of ambiguities:

1. ambiguity in the English language description of the problem;
2. differences in formal parts of the problem description;
3. differences in parameter values adopted

4.3.2 Formal modelling of the Heated Tank versions

The formal modelling of the Heated tank benchmark in Modest worked straightforward for versions 1 through 4. However the straightforward modelling of the temperature dependent failure rates of the Pump and Valve is not yet supported by Modest. This limitation regarding the modelling of version 5 has been reported before for Dynamic Bayesian Networks (DBN) [14].

The formal modelling of the Heated tank benchmark using SDCPN worked straightforwardly for all five versions. This finding is in line with the formal modelling capability of Fluid Stochastic Petri Net (FSPN) [13, 12], as well as the FSPN extension through Stochastic Activity Network (SAN) [11].

As is shown by [42], the Heated tank example is such simple that it can also be captured in a Piecewise Deterministic Markov Process (PDMP) model without the need to use of a compositional model specification approach of FSPN, SAN, SDCPN or Modest. However, the advantage of the latter is that larger stochastic hybrid models that consist of many interacting systems can also be dealt with.

4.3.3 Estimated probabilities for dryout and overflow

The estimation results obtained for dryout probabilities, Overflow probabilities, and Overheating probabilities appear to agree well with those obtained by others in literature. From a rare event probability estimation perspective, most interesting are the estimates obtained for the dryout probability for version 4 of the Heated tank benchmark. In the Table below the results obtained by the various methods are presented. Of these methods, Restart [42] and Modest have shown the advantage of using splitting techniques in MC simulation of rare event probabilities. An overview of the results produced by different tools is shown in Tab. 6.

The results for the Modest Toolset were obtained by using the modes simulator [7] and rare event simulation using fixed effort importance splitting (64 child runs per main run) and an ad-hoc importance function counting the number of components that fail in a relevant way. They were obtained on an Intel Core i7-4790 system running 64-bit Ubuntu Linux 18.04. To compute the reported result with 95% confidence and a relative-width confidence interval of 5% half-width, 2517 fixed effort runs were used, taking 25 seconds of simulation runtime.

Method	SAN&MC [11]	DBN [14]	MC [42]	Restart[42]	Modest	SDCPN & MC
Estimated $\mathbb{P}_{dryout}$	5.1×10^{-4}	5.2×10^{-4}	4.9×10^{-4}	5.4×10^{-4}	5.09×10^{-4}	5.3×10^{-4}
Number of MC runs	100,000	-	-	-	-	100,000
Standard deviation	-	-	1.7×10^{-3}	1.7×10^{-4}	-	-
95% confidence	$\pm 1.4 \times 10^{-4}$	-	-	-	$\pm 5\,\%$	$\pm 1.4 \times 10^{-4}$

Table 6: Results for the Heated Tank benchmark, version 4

4.4 Lawn Mower benchmark

4.4.1 Verification using Barrier Certificates

We use Barrier Certificates for computing bounds on reachability probability. More specifically, the approach has been recently developed for computing lower bound on probability of satisfying a *safe LTL specification*. Thus, it can be used for finding upper bounds on reachability properties. Computationally, the approach is developed for discrete-time stochastic systems with polynomial dynamics and uses sum-of-squares optimization techniques for finding such bounds.

Transformation to polynomial dynamics. The dynamics of the lawn mower is not polynomial. We employ a nonlinear transformation of the states to get polynomial vector fields with polynomial constraints. We do coordinate transform as

$$[y_1, y_2, y_3, y_4, y_5, y_6] = [x_1, \tan\Big(\frac{2(x_2 - bx_1)}{(V_r + V_l)}\Big), x_3, x_4, x_5, x_6].$$

The closed-loop dynamics become

$$\begin{aligned}
y_1[k+1] &= y_1[k] + T_s\left(\frac{1}{I_{zz}}(A_{fy}z_4[k]a + 2C_\gamma y_2[k]b)\right),\\
y_2[k+1] &= y_2[k] + T_s\left(\frac{2(1+y_2[k]^2)}{V_r[k]+V_l[k]}\Big(\frac{1}{M}(A_{fy}z_4[k] - 2C_\gamma y_2[k])\right.\\
&\qquad\left. - \frac{V_r[k]+V_l[k]}{2}y_1[k] - \frac{b}{I_{zz}}(A_{fy}z_4[k]a + 2C_\gamma y_2[k]b)\Big)\right),\\
y_3[k+1] &= y_3[k] + T_s\left(\frac{V_r[k]+V_l[k]}{2}z_2[k]\right),\\
y_4[k+1] &= y_4[k] + T_s\left(\frac{V_r[k]+V_l[k]}{2}z_1[k]\right),\\
y_5[k+1] &= y_5[k] + T_s\left(\frac{V_r[k]-V_l[k]}{2D} + y_1[k]\right) + w_1[k],\\
y_6[k+1] &= y_6[k] + T_s\left(-\frac{V_r[k]+V_l[k]}{2d}z_4[k] - \frac{(V_r[k]-V_l[k](d+\sqrt{D^2+(a+b)^2}z_3[k]}{2Dd}\right) + w_2[k],
\end{aligned}$$

where $z_1 = \sin(y_5)$, $z_2 = \cos(y_5)$, $z_3 = \sin(y_6)$, $z_4 = \cos(y_6)$. The controller (15) in the new coordinates can be simplified to two modes of operation depending on location in state space as

$$\begin{cases} V_l = V_b - V_d, V_r = V_b + V_d, & \text{if } z_1(x_d - y_3) - z_2(y_d - y_4) < 0\\ V_l = V_b + V_d, V_r = V_b - V_d, & \text{if } z_1(x_d - y_3) - z_2(y_d - y_4) > 0. \end{cases}$$

The lower bound $\alpha = 0.8852$ is obtained using YALMIP and SeDuMi for initial states starting from X_0 with the help of barrier certificate of order 2.

4.4.2 Verification using FAUST^2

The dynamics of lawn mover fits in the class of partially degenerate stochastic systems. Verification of this class of systems using abstraction techniques is studied in [38, 37]. The available theoretical results are developed for the general setting without particular assumption on the dynamics except Lipschitz continuity of functions and sets. But the computations are performed for the restricted case of "affine" deterministic dynamics and "polytopic" safe/target sets. This restriction is due to the deterministic reachability analysis required in the first phase of the approach.

The Lawn Mover case study is challenging due to the fact that the deterministic vector filed is nonlinear, thus we would have to perform deterministic reachability over nonlinear dynamics. This in principle is possible and has to be done approximately using numerical methods from deterministic reachability. However, the error computation requires Lipschitz constants of the boundaries of these sets. The computation can be easily done for polytopes, but since we don't know the shape of reachable sets, again the computation should be done numerically.

4.5 Mars Rover benchmark

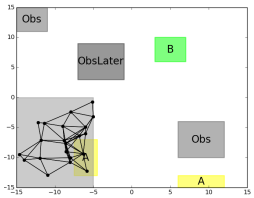
In this section, we present results for a simplified version of the general problem presented in Sec. 2.5. The full problem still presents significant tractability issues, and scalable methods are under active development. Partial solutions to the general problem presented in Sec. 2.5 can also be found in [36, 35].

We tackle the science, safety and uncertainty parts of the mission specification, and assume that: (a) the rover has stochastic motion and perception, and access to a partially known deterministic environment map, and (b) the helicopter has deterministic motion and perception that can suffer from terrain classification errors.

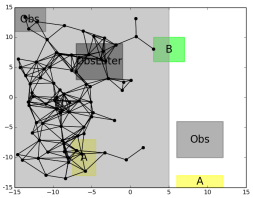
The approach has two components: (a) an off-line planner that returns a motion plan for the rover to satisfy the science mission or generate a request for exploring the environment by the helicopter, and (b) an on-line execution algorithm that is coupled with a monitor to detect mismatches between the maintained environment map and observations.

The off-line method is similar to the work in [43, 15] and starts by converting the LTL specification to a Finite State Automaton (FSA). Then a belief space sampling-based planner called Feedback Information RoadMap (FIRM) [3] is used to compute a finite abstraction of the rover dynamics. The transition probabilities of the product MDP between the FIRM MPD and specification FSA are computed using Monte Carlo simulation as detailed in [43]. The product MDP captures motion, map state evolution, and specification satisfaction concurrently. The maximally satisfying control policy is computed as the solution of a stochastic reachability problem on the product MDP projected onto the system MDP, robot, and environment. Execution failures are handled by re-planning and mapping unexplored regions.

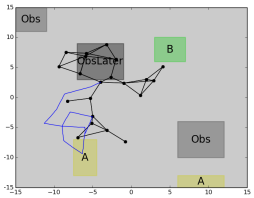
In Fig. 11, we show the results of our planner for a campaign at the Kimberly region in Gale Crater on Mars, visited by Curiosity around Sol 607. The task is to collect one sample from each region of type A (yellow) and type B (green) while avoiding all the obstacles (red) shown in Fig. 11. The environment is partially known, and initially, map labels are available only in the region surrounding the rover's initial state. The rover can request the helicopter to explore new areas to extend the planning region. Furthermore, if there is a mismatch between the map and what the rover encounters during execution, the execution is aborted, and a new plan is computed. Figure 11 shows results of the simulation along with a visualization in the Mars environment. The rover starts by collecting a sample from region A (yellow). Next, it



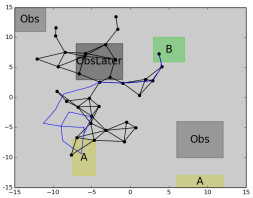
(a) Rover starts by trying to plan in the initial region explored by the helicopter (shaded in gray) but fails to find a solution. Hence, the rover asks the helicopter to map new regions.



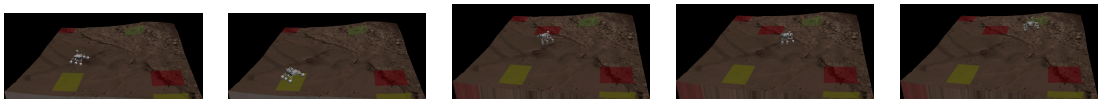
(b) Rover gets the map from the copter and successfully finds a policy that satisfies the specification.



(c) Rover starts executing (blue line) the policy but detects an additional obstacle `ObsLater` that was not labelled in the map obtained from the helicopter. Rover tries to replan but fails to find a solution and extends the FIRM abstraction.



(d) Rover uses the updated map to find a solution from it's current belief and state in the FSA. Finally, it executes the solution and reaches the goal state.



(e) Visualization of the Mars environment and trajectory executed by the rover at Kimberly region in Gale Crater.

Figure 11: Illustration of the incremental planning, re-planning and re-mapping procedure.

moves towards region B (green) but detects a new obstacle (red) on the way. Thus, it re-plans and re-maps the environment until it finds a policy from its current belief and FSA state. After finding the solution, the rover executes it and completes the campaign by picking up a sample from region B.

The planning phase took 19.4 seconds to compute the policy on a system running Ubuntu 14.04 with Intel Core i7-6820HQ CPU @ 2.7 GHz. The product MDP graph had 393 vertices with 2752 edges, and 15 Monte Carlo trials per transition were performed to estimate the transition probabilities. The used algorithms can be found at https://github.com/wasserfeder/gdtl-firm/tree/dev-mars-fsa.

5 Challenges and Future plans

First, we list a set of challenges encountered in this first round of friendly competition within the stochastic modelling category:

- translating and encoding soft specifications, expressed in natural language, into formal specifications, which are tailored to given models
- developing integrated modelling and verification tools, or a software providing an interface (mathematically, a mapping) between different software tools
- addressing a number of mathematical challenges in modelling and in performing rare event simulation of safety-critical Cyber Physical Human Systems (CPHS)

Second, we conclude this section by highlighting our future plans for the future rounds of the competition in this category:

- developing a unified description language for stochastic (hybrid) models
- further developing theory supporting compositional modelling; further, defining specifications for large-scale modular systems
- including benchmarks, tools, or algorithmic advances from areas not currently represented: e.g. work on rare-event estimation, or on agent-based simulations
- extending the barrier certificate technique for synthesising control policies – different optimization techniques are needed for continuous and discrete input spaces

6 Conclusion and Outlook

This report presents the results on a first friendly competition for the formal verification of stochastic models as part of the ARCH'18 workshop. The reports of other categories can be found in the proceedings and on the ARCH website: cps-vo.org/group/ARCH.

7 Acknowledgments

This work is partially supported by the Alan Turing Institute, UK and Malta's ENDEAVOUR Scholarships Scheme.

The authors would like to thank Pushpak Jagtap and Dr Carl Gamble for the discussions on the details of lawn mower case study.

The anesthesia delivery system benchmark is based upon work supported by the National Science Foundation, under Grant Numbers CMMI-1254990 (CAREER, Oishi), CNS-1329878, and IIS-1528047. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

The Mars rover benchmark is based upon research carried out at the Jet Propulsion Lab and Caltech under a contract with the NASA and funded through the President's and Director's Fund Program.

A Specification of Used Machines

A.1 $M_{\text{Barrier Certificates}}$

- Processor: Intel(R) Core(TM) i5-4300U CPU @ 1.90GHz 2.50 GHz
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 3739

A.2 M_{FAUST^2}

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

A.3 $M_{\text{FIRM-GDTL}}$

- Processor: Intel Core i7-6820HQ CPU @ 2.7GHz
- Memory: 8.00 GB
- Average CPU Mark on www.cpubenchmark.net: 8784

A.4 M_{Modest}

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

A.5 $M_{\text{SDCPN \& MC simulations}}$

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

A.6 $M_{\text{SReachTools}}$

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

References

[1] 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.

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

[3] Ali-akbar Agha-mohammadi, Suman Chakravorty, and Nancy Amato. FIRM: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements. *International Journal of Robotics Research (IJRR)*, 33(2):268–304, 2014.

[4] Henk AP Blom, Jaroslav Krystul, GJ Bakker, Margriet B Klompstra, and Bart Klein Obbink. Free flight collision risk estimation by sequential mc simulation. *Stochastic Hybrid Systems*, pages 249–281, 2007.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

[10] 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.

[11] Daniele 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.

[12] Daniele Codetta-Raiteri and Andrea Bobbio. Evaluation of a benchmark on dynamic reliability problems via fluid stochastic Petri nets. In *7th International Workshop on Performability Modeling of Computer and Communication Systems (PMCCS)*, pages 52–55, 2005.

[13] Daniele Codetta-Raiteri and Andrea Bobbio. Solving dynamic reliability problems by means of ordinary and fluid stochastic Petri nets. In *European Safety and Reliability Conference (ESREL)*, pages 381–389, 2005.

[14] Daniele Codetta-Raiteri and Luigi Portinale. Approaching dynamic reliability with predictive and diagnostic purposes by exploiting dynamic Bayesian networks. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 228(5):488–503, 2014.

[15] Eric Cristofalo, Kevin Leahy, Cristian Ioan Vasile, Eduardo Montijano, Mac Schwager, and Calin Belta. Localization of a Ground Robot by Aerial Robots for GPS-deprived Control with Temporal Logic Constraints. In *International Symposium on Experimental Robotics (ISER)*, pages 525–537, Tokyo, Japan, October 2016. doi:10.1007/978-3-319-50115-4_46.

[16] 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.

[17] Sadegh Esmaeil Zadeh Soudjani, Rupak Majumdar, and Tigran Nagapetyan. Multilevel monte carlo method for statistical model checking of hybrid systems. In Nathalie Bertrand and Luca Bortolussi, editors, *Quantitative Evaluation of Systems*, pages 351–367, Cham, 2017. Springer International Publishing.

[18] Mariken HC Everdij and Henk AP Blom. Hybrid petri nets with diffusion that have into-mappings with generalised stochastic hybrid processes. In *Stochastic Hybrid Systems*, pages 31–63. Springer, 2006.

[19] Mariken HC Everdij and Henk AP Blom. Hybrid state petri nets which have the analysis power of stochastic hybrid systems and the formal verification power of automata. In *Petri Nets Applications*. InTech, 2010.

[20] Frederik F Foldager, Peter Gorm Larsen, and Ole Green. Development of a driverless lawn mower using co-simulation. In *International Conference on Software Engineering and Formal Methods*, pages 330–344. Springer, 2017.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] SA Hill. Pharmacokinetics of drug infusions. *Continuing education in anesthesia, critical care & Pain*, 4(3):76–80, 2004.

[27] Pushpak Jagtap, Sadegh Soudjani, and Majid Zamani. Temporal logic verification of stochastic systems using barrier certificates. *ArXiv e-prints*, June 2018.

[28] Austing Jones, Mac Schwager, and Calin Belta. Distribution temporal logic: Combining correctness with quality of estimation. In *IEEE 52nd Conference on Decision and Control (CDC)*, pages 4719–4724, 2013.

[29] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[30] Shahab Kaynama. *Scalable techniques for the computation of viable and reachable sets: Safety guarantees for high-dimensional linear time-invariant systems*. PhD thesis, University of British Columbia, 2012.

[31] 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.

[32] John N Maidens, Shahab Kaynama, Ian M Mitchell, Meeko MK Oishi, and Guy A Dumont. Lagrangian methods for approximating the viability kernel in high-dimensional systems. *Automatica*, 49(7):2017–2029, 2013.

[33] Marzio Marseguerra, Massimo Nutini, and Enrico Zio. Approximate physical modelling in dynamic PSA using artificial neural networks. *Reliability Engineering & System Safety*, 45(1):47–56, 1994.

[34] Marzio Marseguerra and Enrico Zio. Monte Carlo approach to PSA for dynamic process systems. *Reliability Engineering & System Safety*, 52(3):227–241, 1996.

[35] Cristian-Ioan Vasile Rohan Thakker Ali-akbar Agha-mohammadi Aaron D. Ames P. Nilsson, S.Haesaert and Richard M. Murray. Toward specification-guided active mars exploration for cooperative robot teams. In *Conference on Robotics: Science and Systems*, 2018.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

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

[41] B. Tombuyses and T. Aldemir. Continuous cell-to-cell mmapping and dynamic PSA. In *International Conference on Nuclear Engineering*, volume 3, pages 431–438, 1996.

[42] Pietro Turati, Nicola Pedroni, and Enrico 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.

[43] Cristian-Ioan Vasile, Kevin Leahy, Eric Cristofalo, Austin Jones, Mac Schwager, and Calin Belta. Control in Belief Space with Temporal Logic Specifications. In *IEEE Conference on Decision and Control (CDC)*, pages 7419–7424, Las Vegas, NV, USA, December 2016.

[44] Abraham P. Vinod, Joseph Gleason, and Meeko Oishi. Underapproximative verification and controller synthesis for Automated Anesthesia Delivery system using SReachTools.

[45] 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.

[46] 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.

[47] 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.

[48] Huilong Zhang, François Dufour, Yves Dutuit, and K. Gonzalez. Piecewise deterministic Markov processes and dynamic reliability. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 222(4):545–551, 2008.