Verification of probabilistic bounded $\delta$-reachability for cyber-physical systems

Fedor Shmarov  
School of Computing Science, Newcastle University  
Newcastle, NE1 7RU, UK  
Email: f.shmarov@ncl.ac.uk

Paolo Zuliani  
School of Computing Science, Newcastle University  
Newcastle, NE1 7RU, UK  
Email: paolo.zuliani@ncl.ac.uk

Abstract—Verification of cyber-physical systems is a difficult, yet extremely important, problem. Hybrid systems offer a theoretical framework in which to perform formal verification of cyber-physical systems. In this paper we study the problem of bounded $\delta$-reachability in hybrid systems with random initial parameters. We devise a technique for computing reachability probabilities over a finite number of discrete steps for nonlinear hybrid systems featuring a bounded random initial parameter. Our approach is to define an appropriate $\delta$-relaxation of the (undecidable) reachability problem, so that it can be solved by a $\delta$-complete decision procedure. Specifically, we can compute an interval that is guaranteed to contain the probability of, say, a hybrid system behaving in a faulty way. Moreover, we discuss certain types of random variables with unbounded support and show that the bounded $\delta$-reachability problem can still be solved by using an appropriate $\delta$-complete decision procedure. Finally, we propose the development of a validated integration procedure over an arbitrary Borel set in order to cope with hybrid systems with dynamics given by solutions of ordinary differential equations.

I. INTRODUCTION

Cyber-physical systems are characterised by the tight integration of digital computing (the cyber part) with a physical environment or device. Such systems exploit the flexibility of digital computing and communication to enhance or enable new capabilities of physical systems. Hybrid systems are mathematical models that combine continuous dynamics and discrete control, and enjoy widespread use for modelling cyber-physical systems. For example, Stateflow/Simulink\[1] is the de facto standard tool for model-based design of embedded systems; the semantics of Stateflow/Simulink models can be given in terms of hybrid systems [1], [2]. The use of cyber-physical systems permeates our society, including many safety-critical applications where a malfunctioning can result in threats to, or even loss of, human life. For example, modern aircraft are efficiently flown most of the time by a computer program, while anti-lock brakes, stability control, and traction control contribute to safer cars. Again, electronic biomedical devices (e.g., digital infusion pumps that deliver drugs or nutrients to patients) offer superior flexibility and accuracy than traditional devices. It is therefore extremely important to verify formally that such cyber-physical systems are safe.

The state space of a hybrid system consists of a discrete component and of a continuous component. The fundamental reachability problem is to decide whether a hybrid system reaches an unsafe region of its state space (a subset of states indicating incorrect behaviour of the system). Unfortunately, the reachability problem is undecidable even for simple hybrid systems with constant differential dynamics [3]. With further restrictions, e.g., same constant differential dynamics across all the variables — timed automata — the reachability problem becomes PSPACE-complete [4]. However, hybrid systems arising from practical applications feature much richer dynamics, including non-linear functions over the reals, e.g., exponentials and trigonometric functions, for which reachability questions are in general undecidable [5]. This means that the reachability problem must be modified, if we want to solve it algorithmically. It has been recently noted that the reachability problem for general hybrid systems can be relaxed in a sound manner and encoded as a first-order satisfiability modulo theory (SMT) formula [6] which can be correctly solved by a $\delta$-complete decision procedure [7]. Such procedures allow a "tuneable" precision in the answer provided. Basically, SMT solvers and rigorous numerical methods (e.g., interval constraint propagation) are used to verify a conservative approximation of the system behaviour, so that bugs will always be detected. Although the over approximation can be tight (tunable by an arbitrarily small rational parameter, $\delta$), the decision procedure may produce false alarms. It may report a bug when in reality the system is safe. However, such a scenario indicates that the system is safe but fragile, thereby providing valuable information to the system designer. The dReal tool [8] implements $\delta$-complete procedures for nonlinear arithmetic first-order formulae.

For many practical applications it is necessary to augment the definition of hybrid system with stochastic behaviour. Stochastic systems arise naturally when modelling phenomena which are intrinsically probabilistic, e.g., soft errors in computing hardware. Also, stochastic systems can arise due to uncertainty in (deterministic) system components, its behaviours, and its environment. The reachability problem for stochastic hybrid systems is no longer a decision problem. Rather, it generalises that by asking what is the probability that the system reaches the unsafe region. Note that for hybrid systems with both stochastic and non-deterministic behaviour, the problem results in general in a range of probabilities, thereby becoming an optimisation problem. In this paper we give a formal definition of the probabilistic bounded $\delta$-reachability problem for hybrid systems with random continuous initial parameters. We then show how $\delta$-complete decision procedures can be employed for solving this problem. One of the constraints imposed by the $\delta$-complete procedure on the formula is that all the variables should be bounded. Nevertheless, we show...
how to solve a special case of the reachability problem that include random variables with unbounded support. Currently, dReal allows solving the reachability problems only for the systems with one random initial parameter and with dynamics given explicitly. However, many hybrid systems have multiple random parameters and dynamics defined by a system of ordinary differential equations (ODEs). This motivated us for implementing a validated procedure for integration.

Summarising, in this paper we make the following contributions:

- we give a SMT formulation of the probabilistic δ-reachability problem for hybrid systems with random initial parameters;
- we show how to use δ-complete procedures to solve probabilistic δ-reachability, also with random variables with unbounded support;
- we give a validated integration procedure that enables solving probabilistic δ-reachability of general hybrid systems with random initial parameters and ODEs dynamics.

At the time of writing we are not aware of any other technique that addresses probabilistic bounded reachability from a rigorous point of view, i.e., guaranteed to be accurate numerically.

II. BOUNDED δ-REACHABILITY IN HYBRID SYSTEMS

We now formally define hybrid systems, which we use for modelling cyber-physical systems. The following definition is standard.

**Definition 1.** A hybrid system consists of the following components:

- $Q = \{q_0, \ldots, q_n\}$ a set of modes (discrete components of the system),
- $X = [u_1, v_1] \times \ldots \times [u_n, v_n] \times [0, T] \subset \mathbb{R}^{n+1}$ a domain of continuous variables,
- $S = Q \times X$ is the hybrid state space of the system,
- $U \subseteq S$ an unsafe region of the state space, and
- predicates (or relations)

- $flow_q(x^0, x^t)$ mapping the continuous state $x^0$ at time $0$ to state $x^t$ at time point $t \in [0, T]$ in mode $q$;
- $init_q(x)$ indicating that $s = (q, x)$ belongs to the set of initial states;
- $jump_{q \rightarrow q'}(x^t, x^0)$ indicating that the system can make a transition from mode $q$, upon reaching the jump condition in continuous state $x^t$ at time point $t \in [0, T]$, to mode $q'$ and setting the continuous state to $x^0$;
- $unsafe_q(x)$ indicating that $s = (q, x) \in U$.

**Remark.** The continuous dynamics of the system is defined in each flow, and it can either be presented as a system of ODEs or explicitly.

In this paper we are interested in deterministic hybrid systems, where in each mode only one jump is allowed.

Informally, the semantics of a hybrid system can be thought as piece-wise continuous: in each mode the system dynamics follows a continuous flow, while a mode change (as result of a jump) might also involve a discontinuous change in the variables $x$. The evolution of a hybrid system is then obtained by iteratively composing the relations flow and jump, starting with init — see Definition 5 below for an example. More details can be found in [3].

In order to overcome the undecidability of reasoning about hybrid systems, Gao et al. recently defined the concept of δ-satisfiability over the reals [6], and presented a corresponding δ-complete decision procedure [7]. The main idea is to decide correctly whether slightly relaxed sentences over the reals are satisfiable or not. The following definitions are from [6].

**Definition 2.** A bounded quantifier is one of the following:

\[ [a, b] x = \exists x : (a \leq x \land x \leq b) \]

\[ \forall [a, b] x = \forall x : (a \leq x \land x \leq b) \]

**Definition 3.** A bounded $\Sigma_1$ sentence is an expression of the form:

\[ \exists I_1 x_1, \ldots, \exists I_n x_n : \psi(x_1, \ldots, x_n) \]

where $I_i = [a_i, b_i]$ are intervals, $\psi(x_1, \ldots, x_n)$ is a Boolean combination of atomic formulas of the form $g(x_1, \ldots, x_n)$ or $0$, where $g$ is a composition of Type 2-computable functions and $\circ \in \{<, \leq, >, \geq, =, \neq\}$.

**Remark.** Any bounded $\Sigma_1$ sentence is equivalent to a $\Sigma_1$ sentence in which all the atoms are of the form $f(x_1, \ldots, x_n) = 0$ (i.e., the only $\circ \in \{=\}$ needed is ‘=’) [7].

Essentially, Type 2-computable functions can be approximated arbitrarily well by finite computations of a special kind of Turing machines (Type 2 machines); most of the ‘useful’ functions over the reals (e.g., continuous functions) are Type 2-computable [9].

The notion of δ-weakening [6] of a bounded sentence is central to δ-satisfiability.

**Definition 4.** Let $\delta \in \mathbb{Q}^+ \cup \{0\}$ be a constant and $\phi$ a bounded $\Sigma_1$-sentence in the standard form

\[ \phi = \exists I_1 x_1, \ldots, \exists I_n x_n : \bigwedge_{i=1}^{m} \bigwedge_{j=1}^{k_i} \bigvee f_{ij}(x_1, \ldots, x_n) = 0 \]

where $f_{ij}(x_1, \ldots, x_n) = 0$ are atomic formulas. The δ-weakening of $\phi$ is the formula:

\[ \phi^\delta = \exists I_1 x_1, \ldots, \exists I_n x_n : \bigwedge_{i=1}^{m} \bigwedge_{j=1}^{k_i} \bigvee |f_{ij}(x_1, \ldots, x_n)| \leq \delta \]

Note that $\phi$ implies $\phi^\delta$, while the converse is obviously not true. The bounded δ-satisfiability problem asks for the following: given a sentence of the form [1] and $\delta \in \mathbb{Q}^+$, correctly decide whether

- **unsat:** $\phi$ is false,
- **δ-true:** $\phi^\delta$ is true.

If the two cases overlap either decision can be returned: such a scenario reveals that the formula is fragile — a small
perturbation (i.e., a small \( \delta \)) can change the formula’s truth value. The dReal tool \([3]\) implements an algorithm for solving the \( \delta \)-satisfiability problem. Basically, the algorithm combines a DPLL procedure \([10]\) (for handling the Boolean parts of the formula) with interval constraint propagation \([11]\) (for handling the real arithmetic atoms). More details on the implementation can be found in \([8]\).

A qualitative property of hybrid systems that can be checked is bounded \( \delta \)-reachability. It asks whether the system reaches the unsafe region after \( k \in \mathbb{N} \) discrete transitions.

**Definition 5.** \([12]\) Bounded \( k \) step \( \delta \)-reachability in hybrid systems can be encoded as a bounded \( \Sigma_1 \)-sentence

\[
\exists x_{0,0}^0, \exists x_{0,0}^t, \ldots, \exists x_{0,q_m}^0, \exists x_{0,q_m}^t, \ldots, \exists x_{k,q_m}^0, \exists x_{k,q_m}^t : \\
( \bigvee_{q \in Q} (\text{init}_q(x_{0,0}^0) \land \text{flow}_q(x_{0,0}^0, x_{0,q_m}^0))) \\
\land (\bigwedge_{i=0}^{k-1} (\bigvee_{q,q' \in Q} (\text{jump}_{q \rightarrow q'}(x_{i,q}^t, x_{i+1,q'}^0)))) \\
\land (\text{flow}_{q'}(x_{i+1,q'}^0, x_{i+1,q'}^t)) \land (\bigvee_{q \in Q} \text{unsafe}_q(x_{k,q}^0)))
\]

where \( x_{i,q}^0 \) and \( x_{i,q}^t \) represent the continuous state in the mode \( q \) at the depth \( i \), and \( q' \) is a successor mode.

Intuitively, the formula above can be understood as follows: the first conjunction is asking for a set of continuous variables which satisfy the initial condition in one of the modes and the flow in that mode; the second conjunction is looking for a set of vectors which satisfy any \( k \) discrete jumps and flows in each successor mode defined by the jumps; the third conjunction is verifying whether the state of the system (the mode and the set of continuous variables in the mode after \( k \) jumps) belongs to the unsafe region. Note that the previous definition asks for reachability in exactly \( k \) steps. One can build a disjunction of formula \([3]\) for all values from 1 to \( k \), thereby obtaining reachability within \( k \) steps.

The \( \delta \)-reachability problem can be solved using the described \( \delta \)-complete decision procedure, which will correctly return one of the following answers:

- **unsat**: the system never reaches the bad region \( U \),
- **\( \delta \)-true**: the \( \delta \)-perturbation of \([3]\) is true, and a witness, \( i.e., \) an assignment for all the variables, is returned.

**III. BOUNDED \( \delta \)-REACHABILITY IN PROBABILISTIC HYBRID SYSTEMS**

We now introduce \( \delta \)-reachability in a probabilistic context.

**Definition 6.** A hybrid system with random initial parameters consists of the following components:

- \( Q = \{q_0, \ldots, q_m\} \) a set of modes (discrete components of the system),
- \( X = [u_1, v_1] \times \ldots \times [u_n, v_n] \times [0, T] \subset \mathbb{R}^{n+1} \) a domain of continuous variables,
- \( R = [a_1, b_1] \times \ldots \times [a_l, b_l] \subset \mathbb{R}^l \) a domain of random parameters,
- \( S = Q \times R \times X \) is the state space of the system,
- \( U \subseteq S \) an unsafe region of the state space,

predicates (relations)

- \( \text{flow}_q(r, x^0, x^t) \) mapping the continuous state \( x^t \) at time 0 to state \( x^t \) at time point \( t \in [0, T] \) in mode \( q \),
- \( \text{init}_q(r, x) \) indicating that \( (q, r, x) \) belongs to the set of initial states,
- \( \text{jump}_{q \rightarrow q'}(r, x^t, x^0) \) indicating that the system can make a transition from mode \( q \), upon reaching the jump condition in continuous state \( x^t \) at time point \( t \in [0, T] \), to mode \( q' \) and setting the continuous state to \( x^0 \),
- \( \text{unsafe}_q(r, x) \) indicating that \( (q, r, x) \in U \),

and we require that for all \( q \in Q \) the sets defined by \( \text{flow}_q, \text{init}_q, \text{jump}_{q \rightarrow q'}, \) and \( \text{unsafe}_q \) are Borel.

**Remark.** The newly introduced random parameters are assigned in the initial mode and remain unchanged throughout the system’s evolution: note that flow and jump do not specify any change on \( r \). Also, the Borel assumption for the sets defined by the predicates is a theoretical requirement for well-definedness of probabilities, and in practice it is easily satisfied. (We recall that Borel sets are obtained through countable union and complement of open sets.)

Then bounded \( k \) step reachability problem for the defined system will be modified.

**Definition 7.** The bounded \( k \) step \( \delta \)-reachability property for hybrid systems with random initial parameters is the bounded \( \Sigma_1 \) sentence:

\[
\exists r, \exists x_{0,0}^0, \exists x_{0,0}^t, \ldots, \exists x_{0,q_m}^0, \exists x_{0,q_m}^t, \ldots, \exists x_{k,q_m}^0, \exists x_{k,q_m}^t : \\
( \bigvee_{q \in Q} (\text{init}_q(r, x_{0,0}^0) \land \text{flow}_q(r, x_{0,0}^0, x_{0,q_m}^0))) \\
\land (\bigwedge_{i=0}^{k-1} (\bigvee_{q,q' \in Q} (\text{jump}_{q \rightarrow q'}(r, x_{i,q}^t, x_{i+1,q'}^0)))) \\
\land (\text{flow}_{q'}(r, x_{i+1,q'}^0, x_{i+1,q'}^t)) \land (\bigvee_{q \in Q} \text{unsafe}_q(r, x_{k,q}^0)))
\]

If we associate a probability measure to the random parameters, then we can assess quantitative system properties (such as the probability that the system reaches the unsafe region). In particular, we consider the following problem: what is the probability that a hybrid system with random initial parameters reaches the unsafe region in \( k \) steps?

To calculate the probability of reaching the unsafe region, we first need to find the set which contains all the values of the random variables satisfying the reachability property of a
hybrid system. Such a set is constructed as follows:

\[
B = \{ (x_0, q) : x_i^0, x_i^1, \ldots, x_i^N, x_i^0_q, \ldots, x_i^N_q, x_i^0_{q_m}, \ldots, x_i^N_{q_m} : \}
\]

\[
( \bigwedge_{q \in Q} (\text{init}_q(x_0_i, q) \land \text{flow}_q(x_i^0_q, x_i^1_q)))
\]

\[
\land \left( \bigwedge_{i=0}^{k-1} \left( \bigvee_{q,q'} \left( \text{jump}_{q} \rightarrow q' \left( x_i^0_q, x_i^1_q, x_i^{i+1}_q \right) \right) \right) \right)
\]

\[
\land \left( \bigwedge_{q \in Q} \left( \text{un safe}_q(x_i^0_q) \right) \right) \right). }
\]

(4)

**Proposition 1.** The set \( B \) defined by (4) is Borel.

**Proof:** Immediate from the fact that (Definition 6) the sets defined by \( \text{flow}_q, \text{init}_q, \text{jump}_q, \) and \( \text{un safe}_q \) are Borel, and conjunction and disjunctions correspond to set intersection and union, respectively.

The proposition entails that the probability that the system reaches the unsafe region is well-defined. Such a probability is computed by integrating the probability measure of the random variables over the set (4). In particular, we need to compute the following integral

\[
\int_B dP(r)
\]

(5)

where \( B \) is the set (4) and \( dP(r) \) is a probability measure over the random parameters \( r \). In case that the random parameters are independent then \( P(r) = \Pi_{i=1} P_i(r), \) where \( P_i \) is the probability measure of parameter \( r_i. \) For most practical applications, the infinitesimal probability measure \( dP(r) \) is simply given by a (probability) density function.

**IV. SYSTEMS WITH ONE RANDOM PARAMETER**

In the scope of this paper we consider hybrid systems with one random initial parameter, and we propose an algorithm solving the probabilistic \( \delta \)-reachability problem for such hybrid systems. In a hybrid system with random parameters, the random variable \( r \) is defined over a bounded interval. However, we will demonstrate that we can correctly evaluate probabilistic reachability problems also for certain types of unbounded random variables.

**Proposition 2.** The probabilistic reachability problem for hybrid systems with a single random parameter can be correctly solved by a \( \delta \)-complete decision procedure if:

- the domain of the random parameter is bounded from one side,
- the domain of the random parameter is unbounded and its probability density function is symmetric.

**Proof:** Let’s consider the first case when the domain \( \Omega \) of the continuous random variable is bounded from one side. The problem when set \( B \subseteq \Omega \) is bounded is trivial and probabilistic reachability problem can be solved correctly by the \( \delta \)-complete procedure.

If \( B \) is unbounded then its complement \( \Omega / B \) is bounded (because \( \Omega \) is bounded from one side and \( B \subseteq \Omega \). Set \( \Omega / B \) can be constructed by inverting the relation \( \text{un safe}_q(r, x) \).

Taking into account that

\[
\int_B dP(r) + \int_{\Omega / B} dP(r) = \int_{\Omega} dP(r) = 1
\]

(6)

probabilistic reachability problem (5) is equivalent to

\[
\exists B_1 \subseteq \Omega / B : 1 - \int_{B_1} dP(r) < C
\]

(7)

Therefore, solving two problems (5) and (7) at the same time by the \( \delta \)-complete decision procedure guarantees that one of them will terminate with the correct result. Knowing one, we can derive another from the equation (6).

Let’s consider the second case when \( \Omega \) is unbounded and the probability density function is symmetric. We can find two disjoint bounded from one side sets \( \Omega^-, \Omega^+ \subseteq \Omega \) such that \( \Omega^- \cup \Omega^+ = \Omega \) and \( \int_{\Omega^-} dP(r) = \int_{\Omega^+} dP(r) = \frac{1}{2} \).

Let \( B \subseteq \Omega \). We can find two disjoint sets \( B^-, B^+ \subseteq B \) such that \( B^- \cup B^+ = B, B^- \subseteq \Omega^- \) and \( B^+ \subseteq \Omega^+. \) Then problem (5) is equivalent to:

\[
\exists B^-_1 \subseteq B^-, \exists B^+_1 \subseteq B^+ : \int_{B^-_1} dP(r) + \int_{B^+_1} dP(r) \geq C
\]

(8)

where for each of the subsets \( B^- \) and \( B^+ \):

\[
\int_{B^-} dP(r) + \int_{\Omega^- / B^+} dP(r) = 0.5
\]

\[
\int_{B^+} dP(r) + \int_{\Omega^+ / B^-} dP(r) = 0.5
\]

(9)

If either of subsets \( B^- \) and \( B^+ \) is unbounded then the technique described above should be applied.

In general, the set \( B \) obtained in (4) will be of the form:

\[
B = \bigcup_{i=1}^w B_i
\]

for some \( w \in \mathbb{N} \), where \( B_i = [r_i, \tau_i] \) and \( \forall i \neq j \in \{1, \ldots, w\} : B_i \cap B_j = \emptyset \). However, if the functions representing the dynamics of the system are invertible with respect to the random parameter, then set \( B \) is formed by a single interval. This allows us to write the computation of integral (5) as a bounded \( \Sigma_1 \) sentence

\[
\exists r \in [a, b] : \int_a^b f_R(x) dx \geq C
\]

where \( f_R \) is the probability density function of the random parameter and \( C \in [0, 1] \) is a constant.

The probabilistic bounded reachability problem in hybrid systems with one random initial parameter and with invertible dynamics with respect to the random variable can be solved using a validated ODE solver that allows encoding integrals as IVPs (initial value problem). We aim for this formalisation as that is the way in which dReal handle ODE dynamics.
V. INTEGRATION PROBLEM VIA DIFFERENTIATION

In the previous section we argued that solving probabilistic reachability questions amounts to solving bounded $\Sigma_1$ sentences of the following type:

$$\exists x \in [a, b] : \int_a^x f(t) \, dt \geq C \quad (10)$$

for some constant $C \in [0, 1]$.

**Proposition 3.** Let $f: [a, b] \rightarrow \mathbb{R}$ be a Lipschitz-continuous function. Then there exists one and only one continuous and differentiable function $F$ on $[a, b]$ such that $F(a) = 0$ and $\forall x \in [a, b] : \int_a^x f(t) \, dt = F(x)$

**Proof:** Let us formulate an initial value problem:

$$(F'(x) = f(x)) \land (F(a) = 0) \quad (11)$$

where $f(x)$ is Lipschitz-continuous in $F(x)$ and continuous on $[a, b]$. By the Picard-Lindelöf theorem and the assumption $F(a) = 0$, the formulated IVP has a unique solution on $[a, b]$ derived as following:

$$\forall x \in [a, b] : \int_a^x f(t) \, dt = F(x) - F(a) = F(x)$$

**Proposition 3** allows assessing the value of the integral by the value of the function which is obtained from the initial value problem (11). Therefore, the integration problem (10) is equivalent to the sentence

$$\exists x \in [a, b] : (F'(x) = f(x)) \land (F(x) \geq C) \land (F(a) = 0)$$

which can be directly solved by dReal as a \(\delta\)-satisfiability problem.

VI. PROBABILISTIC BOUNCING BALL

As an example of a hybrid system described above we consider the following scenario (Figure 1): the ball is launched from the initial point with initial speed \(v_0\) and angle to the horizon \(\alpha\). When the ball reaches the ground it reflects according to the laws of reflection.

![Fig. 1. Bouncing ball scenario](image)

The ball trajectory is defined by the system of equations:

$$S_x(t) = S_{x_0} + v_0 t \cos \alpha \quad (12)$$

$$S_y(t) = S_{y_0} + v_0 t \sin \alpha - \frac{gt^2}{2} \quad (13)$$

where $S_x$ and $S_y$ are projections (on the axes $x$ and $y$, respectively) of the distance traveled by the ball starting at the point $(S_{x_0}; S_{y_0})$, \(v_0\) is the initial speed, \(\alpha\) is the angle to the horizon, \(g\) is standard gravity and \(t\) is time. In this example we assume that the initial point of the ball’s center is $(S_{x_0} = 0; S_{y_0} = 0)$

A hybrid system characterizing the behavior of the ball will consists of a single mode with the dynamics defined in (12) and (13) and the only jump to the same mode, during which the time \(t\) is reset to 0, the value of \(S_x(t)\) before the jump is assigned to the initial value of \(S_x(t)\) in the mode after the jump, and the speed of the ball is reduced by the coefficient 0.9.

In order to comply with the description of the hybrid system with random continuous initial parameters, let \(\alpha\) be uniformly distributed over the interval $[0, 0.5]$ with the probability density function $f_\alpha(\alpha) = 2$. Let \(v_0 = 20\) and \(t \in [0, 3]\).

Let us consider a bounded probabilistic reachability problem: is the probability that the system reaches in 4 steps the box $D = \{S_x \in [90, 91], S_y \in [1, 2]\} > 0.062$?

The Borel set of continuous parameters is formed as:

$$B = \{\alpha : \exists t_{0, q_0}, t_{1, q_0}, t_{2, q_0}, t_{3, q_0} \in [0, T] :$$

$$\begin{align*}
(S_{x_0} = v_0 t_{0, q_0} \cos \alpha) & \land (S_{y_0} = v_0 t_{0, q_0} \sin \alpha - \frac{gt^2}{2} q_0) \land (S_{y_0} = 0) \land (t_{0, q_0} > 0) \\
& \land (S_{x_1} = S_{x_0} + 0.9 v_0 t_{1, q_0} \cos \alpha) \\
& \land (S_{y_1} = 0.9 v_0 t_{1, q_0} \sin \alpha - \frac{gt^2}{2} q_0) \land (S_{y_1} = 0) \land (t_{1, q_0} > 0) \\
& \land (S_{x_2} = S_{x_1} + 0.9 v_0 t_{2, q_0} \cos \alpha) \\
& \land (S_{y_2} = 0.9^2 v_0 t_{2, q_0} \sin \alpha - \frac{gt^2 q_0}{2} ) \land (S_{y_2} = 0) \land (t_{2, q_0} > 0) \\
& \land (S_{x_3} = S_{x_2} + 0.9^3 v_0 t_{3, q_0} \cos \alpha) \\
& \land (S_{y_3} = 0.9^3 v_0 t_{3, q_0} \sin \alpha - \frac{gt^2 q_0}{2} ) \land (S_{y_3} = 0) \land (t_{3, q_0} > 0)
\end{align*}$$

with the probability of \(S_x \in [90, 91]\) and \(S_y \in [1, 2]\). Functions $S_x(\alpha, t)$ and $S_y(\alpha, t)$ are invertible with respect to $\alpha$ on the domain $[0, 0.5] \times [0, 3]$. Then by applying **Proposition 1**, the formulated reachability problem can be encoded as an SMT formula and solved using dReal:

\[\exists \alpha \in B : [\alpha_{min}, \alpha_{max}] : (F_\alpha^A(\alpha)) \land (F_\alpha(\alpha_{min}) = 0) \land (F_\alpha(\alpha) \geq 0.062)\]

The actual dReal output on the formula above was as follows:

SAT with the following box:

- **time_0**: [0.0312762400337171, 0.0312762400337171]
- **Fa_t**: [0.0625524053632542, 0.06255247702735234]
- **Fa_0**: 0
- **a_0**: [0.4292134593911809, 0.4292134922320248, 1.528734859618606, 1.528734859618606]
- **t_10**: [1.37586135729942, 1.375861377493665, 1.375861377493665, 1.375861377493665]
- **t_20**: [1.042516535440058, 1.042516555982107, 1.042516555982107, 1.042516555982107]
- **t_30**: [1.1813824229846001, 1.1813824229846001, 1.1813824229846001, 1.1813824229846001]
- **t_40**: [1.63244180862484, 1.63244191572182]
The probability of reaching the unsafe region by the system thus falls into the interval 
\([0.062524053632542, 0.0625247702735234]\), that is, the interval associated to the final value of \(F_a\) (corresponding to \(F_{a,t}\) in the printout above). The obtained result was validated using a simple Monte Carlo method in MATLAB. The time variable was discretised over the interval \([0, 3]\) and the system was simulated using 10,000 uniformly random samples from the interval \([0, 0.5]\). The number of samples hitting the unsafe region was 620, thereby giving a probabilistic estimate of \(0.620 = 0.062\). In comparison, our method gives an interval of size \(0.72 \times 10^{-7}\) which is guaranteed to contain the actual probability.

### VII. Probabilistic Reachability

We now present a more general procedure for calculating the probability of reaching the unsafe region in \(k\) steps. The procedure works for general hybrid systems whose dynamics is given implicitly, e.g., as a solution of ODEs. In particular, we do not require the dynamics to be invertible with respect to the random parameter. The main idea is to compute the probability by integrating an indicator function over the probability measure of the random variable as:

\[
\int_{\Omega} I_U(r)dP(r)
\]

where \(P(r)\) is a probability measure of the random variable, \(\Omega\) is the range of the random variable, and \(I_U\) is the indicator function defined as:

\[
I_U(r) = \begin{cases} 
1, & \text{system with parameter } r \text{ reaches } U \text{ in } k \text{-steps} \\
0, & \text{otherwise}
\end{cases}
\]

The procedure for solving probabilistic reachability thus consists of a validated integration procedure and a \(\delta\)-complete decision procedure used for verifying the indicator function (and thus building the set \(B\)).

**Notation.** For an interval \([r] = [\tau, \pi] \subset \mathbb{R}\) we denote the size of the interval by \(width([r]) = \pi - \tau\) and by \(mid([r]) = \frac{\tau + \pi}{2}\) the central point of the interval.

#### A. Validated Integration Procedure

In the implementation of a validated integration procedure we employ the (1/3) Simpson rule:

\[
K = \int_{a}^{b} f(x) \, dx = \frac{width([I])}{6} \left( f(a) + 4f(mid([I])) + f(b) - \frac{width([I])^5}{2880} f^{(4)}(\xi) \right)
\]

where \([I] = [a, b], \xi \in [I]\) and \(f^{(4)}\) is the fourth derivative of \(f\). Applying interval arithmetics an interval extension of function \(f\) and its fourth derivative can be obtained.

**Definition 8.** An interval extension of a function \(f : X \to Y\) is an operator \([\cdot]\) such that:

\[
\forall x \in [r] \subseteq X : f(x) \in [f]([r]) \subseteq Y
\]

The interval version of Simpson’s rule can be obtained simply by replacing in (14) occurrences of \(f\) and \(f^{(4)}\) with their interval extension \([f]\), and by replacing \(\xi\) with the entire interval \([a, b]\).

\[
K \in [K]([a, b]) = \frac{width([I])}{6} \left( [f](a) + 4[f](mid([I])) + [f](b) - \frac{width([I])^5}{2880} [f^{(4)}([I])] \right)
\]

Furthermore, by the definition of integral:

\[
K \in \sum_{i=1}^{n} [K]([x_i])
\]

where \(n\) is a number of disjoint intervals \([x_i]\), that partition \([a, b]\). Interval extensions can be readily computed using interval arithmetics libraries such as FILIB++ [14].

In order to guarantee \(\delta\)-completeness of the integration it is sufficient to partition \([a, b]\) into \(\delta\) disjoint intervals \([x_i]\), such that for each \([x_i]\) we have \(width([I]([x_i])) < \frac{\delta width([x_i])}{b-a}\). Then the exact value of the integral will be within an interval of width smaller than \(\delta\). Pseudo-code for the procedure computing the integral up to an arbitrary \(\delta \in \mathbb{Q}^+\) is given in Algorithm 1.

**Algorithm 1: Validated Integration Procedure**

**input:** \(f(x), [a, b], \delta;\)  
**output:** \([I]\);  
**B.push([a, b]);\)  
**while size(B) > 0 do**  
\([x] = B.pop();\)  
\([S][([x])] = \frac{width([x])}{6} \left( [f]([x]) + 4[f](mid([x])) + \frac{width([x])^5}{2880} [f^{(4)}][([x])] \right);\)  
**if width([S][([x])]) > \delta \frac{width([x])}{b-a} then**  
**B.push([x, mid([x])]);\)**  
**B.push([mid([x]), [x]]);\)**  
**end**  
**else**  
\([I] = [I] + [S][([x])];\)**  
**end**  
**end**  
**return \([I]\);**

#### B. Borel set

Now we need to correctly identify the Borel set, or equivalently, verifying the indicator function above. Let \(\phi\) be a formula of the form:

\[
\phi([r]) = \exists r \in [r], \exists x_{0,0}, \exists x_{0, q_0}, \exists x_{0, q_0}, \ldots, \exists x_{0, q_m}, \\
\exists x_{k, q_0}, \exists x_{k, q_1}, \ldots, \exists x_{k, q_1}, \exists x_{k, q_m}, \exists x_{k, q_m}; \forall k, q_m : \\
(\bigvee_{q \in Q} (init_q(r, x_{0, q}) \land \text{flow}_q(r, x_{0, q}, x_{1, q})) \land \\
(\bigwedge_{i=0}^{k-1} (\bigvee_{q, q' \in Q} \text{jump}_{q \to q'}(r, x_{i+1, q}, x_{i+1, q'}))) \land \\
(\bigwedge_{q \in Q} (\text{flow}_{q'}(r, x_{i+1, q'}, x_{i+1, q''}) \land \\
(\bigvee_{q \in Q} \text{unsafe}_q(r, x_{i, q}))))
\]

(15)
If the formula is true then \( r \) contains a value such that the system initial random parameter reaches the unsafe region.

Taking a complement of the unsafe region \( U^C = S/U \) (where \( S \) is the state space of the system) and defining a predicate unsafe\(^C\)\((r,x^t)\) evaluating to true iff \( s = (q,r,x^t) \in U^C \) we want to ensure that the system never reaches the unsafe region within the \( k \)-th step with an initial random parameter from \( r \). In order to conclude that it is sufficient to evaluate the formula:

\[
\phi^C([r]) = \exists r \in [r], \exists x^0_{0,q_0}, \exists x^1_{0,q_0}, \ldots, \exists x^0_{0,q_m}, \\
\exists x^1_{1,q_m}, \ldots, \exists x^0_{k,q_m}, \exists x^1_{k,q_m}, \forall t'_{k,q_m} \in [0,t_{k,q_m}] : \\
(\bigvee_{q \in Q} (\text{init}_q(r,x^0_{0,q}) \land \text{flow}_q(r,x^0_{0,q},x^1_{0,q})) \land \\
(\bigvee_{q \in Q} (\text{jump}_{q,q'}(r,x^0_{1,q},x^1_{1,q'}) \land \\
(\bigvee_{q \in Q} \text{unsafe}_q(r,x^0_{k,q}) \land \text{jump}_{q,q'}(r,x^0_{k,q},x^1_{k,q'}) \lor \\
(t_{k,q_m} \geq T))))))
\]

If the formula evaluates to true then the system does not reach the unsafe region on the \( k \)-th step. Then set \( B \) can be defined as the collection \([r] : \phi([r]) \land (\neg \phi^C([r]))\).

**Remark.** The last term in formula (16) ensures that the system either does not reach the unsafe region on the \( k \)-th step before it can make a transition to the successor mode or it reaches the time bound before reaching the unsafe region. This should not be confused with reaching the time bound in any of the preceding modes as it means that the system fails to reach the \( k \)-th step and should be, therefore, unsatisfiable.

### C. Probabilistic reachability

By integrating a probability density function of a random variable we obtain intervals such that size of the enclosure containing the exact value of a partial sum for each of them is smaller than a local error. Then a decision about the relation between each of the obtained intervals and the (unknown) Borel set \( B \) should be taken. In order to achieve this we use a \( \delta \)-complete decision procedure, *i.e.*, dReal, to verify formulas (13) and (16). The following three outcomes are considered:

- \( \phi([r]) \) is \( \delta \)-sat and \( \phi^C([r]) \) is unsat

- \( \phi([r]) \) is unsat

- \( \phi([r]) \) is \( \delta \)-sat and \( \phi^C([r]) \) is \( \delta \)-sat

In the first case, we can conclude that \( [r] \cap B = [r] \). Therefore, \( [r] \subseteq B \) and the value of the partial sum on this interval should be added to the resulting value of the integral. The second case allows concluding that \( [r] \cap B = \emptyset \). Hence, this interval can be completely removed from the integration domain. Finally, in the last case, \( [r] \cap B \neq \emptyset \) and \( [r] \cap B \neq [r] \). In other words, the interval \([r]\) is not fully contained in the set \( B \). Hence, this interval should be split into two subintervals and each of them should be checked in the same way.

After each run through the set of intervals \([r]\), the over-approximated \([I_{\text{over}}]\) and under-approximated \([I_{\text{under}}]\) interval values of the integral are calculated. The under-approximation value is calculated only over the intervals fully contained in set \( B \) while the over-approximation of the integral involves values of partial sum on the intervals which are either fully or partially contained in set \( B \). The exact value of the integral will always be within the interval \([I_{\text{under}}, I_{\text{over}}]\). The algorithm terminates when the length of the interval \([I_{\text{under}}, I_{\text{over}}]\) is smaller than \( \delta \). Pseudo-code for the procedure is presented in Algorithm 2.

**Algorithm 2: Probabilistic \( \delta \)-reachability**

```plaintext
input: f(x), [a, b], \delta_1, \delta_2, \phi, \phi^C;
output: [I];
// stack for all the intervals
B.push((a, b));
// stack only for the intervals and
// partial sum satisfying a local
T = \emptyset;
while size(B) > 0 do
    [x] = B.pop();
    [S][([x]) = width([x]);
    [f]([x]) + 4 [f]([mid([x])]) +
    \frac{[f]}{[T]} = \frac{width([x])^2}{2 \delta_0} [f]([x]);
    if width([S])([x]) > \delta_2 \cdot \frac{width([x])}{2} then
        B.push([x, mid([x])]);
        B.push([mid([x]), T]);
    else
        T.push([x], [S])([x]));
end
end
[I_{\text{under}}] = [0,0,0,0];
[I_{\text{over}}] = [0,0,0,0];
[I] = [0,0,1,0];
// stack containing extra divisions of
// the intervals
D = \emptyset;
while size(T) > 0 do
    [x], [S][([x]) = T.pop();
    if \phi([x]) then
        [I_{\text{over}}] = [I_{\text{over}}] + [S][([x])];
    if \phi^C([x]) then
        D.push([x, mid([x])], [S][([x], mid([x])])];
        D.push([mid([x]), T], [S][([mid([x])], T])];
    end
    else
        [I_{\text{under}}] = [I_{\text{under}]} + [S][([x])];
    end
end
[I] = [I_{\text{under}}, I_{\text{over}}];
[I_{\text{over}}] = [I_{\text{under}}];
T = D;
D = \emptyset;
end
end
return [I];
```
VIII. EXPERIMENTS

Source code (C++) for both the validated integration procedure, probabilistic δ-reachability, and the model files are available on https://github.com/shmarovfedor/ISSRE. The implementation uses the CAPD library [13] for computing the various interval extensions required by the integration procedure. All the experiments were carried out on a Intel Xeon E5-2690 2.90GHz system running Linux Ubuntu 12.04LTS.

A. Bouncing ball

The ball is launched from the initial point $(S_x = 0, S_y = 0)$ with initial speed $v_0 \sim N(20, 1), i.e.,$ normal distribution with mean 20 and variance 1, and angle to horizon $\alpha = 0.7854$. The system can be modeled as a hybrid system with one mode with dynamics governed by a system of ODEs:

$$S'_x(t) = v_0 \cos \alpha$$
$$S'_y(t) = v_0 \sin \alpha - gt$$

We consider the following cases of a bouncing ball:

- without energy loss (the speed remains constant)
- with energy loss (after each jumps the speed is reduced by 0.9)

The goal of the experiment is to calculate the probability of reaching the region $S_x(t) \geq 100$ within 0, 1, 2 and 3 jumps. The results are presented in Table I.

The obtained result was validated using a simple Monte Carlo method in MATLAB. The ODE system was solved symbolically and used in the simulation. Random samples were distributed normally with mean $\mu = 20$ and standard deviation $\sigma = 3$; we calculated a confidence interval using the sample size returned by the Chernoff-Hoeffding [15] bound $N = \frac{\log \frac{1}{c^2}}{\epsilon^2}$, where $\epsilon$ is the interval half-width and $c$ is the coverage probability. For our simulation we used $\epsilon = 0.00001$ and $c = 0.99999$, which required a sample size of 57,564,627,325. Table I contains the results of Monte Carlo simulation.

B. Thermostat

The main purpose of the system is to keep the temperature within the desired range. The system is modelled by a two mode hybrid system [3] (Fig. 2). The temperature is changing exponentially and it is decreasing in the first mode and increasing in the second mode. The system starts in mode 1 with the initial temperature $T_0$ which is normally distributed $(\mu = 30$ and $\sigma = 1)$. When the temperature drops to the minimum level $T_{min} = 18$, the system makes a transition to mode 2, where the temperature increases until it reaches a maximum level $T_{max} = 22$. Then the system makes a jump to mode 1 and the loop repeats again. In the model we use function $\tau(t)$ to represent the global time, as the current time is reset when the system makes a discrete transition.

The goal of the experiment is to calculate the probability of reaching the region $T(t) \in [19.9, 20.1]$ in mode 2 at various time points $\tau$. The results are presented in Table III.

We can extend the 2-mode thermostat model to a 4-mode version by adding two delay modes (Fig. 3). The initial mode of the system is mode 1. Modes 1 and 3 are equivalent to...
 modes 1 and 2 in the 2-mode thermostat. Modes 2 and 4 model a delay of 0.1 seconds. The goal of the experiment is to calculate the probability of reaching the region $T(t) \in [19.9, 20.1]$ in mode 3 at various time points $\tau$. The results are presented in Table IV.

C. Controlled bouncing ball

Consider a 2-mode hybrid system (Fig. 4) modelling a controlled bouncing ball [16]. In mode 1, a ball of mass $m = 7$ is dropped on a platform attached to a stiff spring and a damper from a random height $H_0$, which is distributed normally ($\mu = 9$ and $\sigma = 1$). When the ball reaches the platform ($H = 0$) the system makes a transition to mode 2, where the ball is reflected from the platform and it jumps back to mode 1 when the height of the ball is greater than 0.

The goal of the experiment is to calculate the probability that the ball reaches the region $H > 7$ in mode 1 after making one bounce. The results are presented in Table V.

| Table IV. Probabilistic reachability of the thermostat model (4 modes) |
|---|
| $#$ | $k$ | Probability interval | $CPU$ |
| 1 | 2 | $[0.0007687433606520627, 0.0007687433607436878]$ | 53 |
| 2 | 6 | $[9.585015171225825e-08, 9.684797129694618e-08]$ | 343 |
| 3 | 6 | $[0.003967491767795972, 0.003967492552568959]$ | 708 |

$k$ = number of discrete transitions, $\tau$ = global time, $CPU$ = CPU time in seconds

The obtained result was validated using Monte Carlo simulation in MATLAB (Table VI). The time variable was discretised over the interval $[0, 5]$ in mode 1 and $[0, 0.5]$ in mode 2, and the system was simulated using random samples distributed normally with mean $\mu = 9$ and standard deviation $\sigma = 1$. The number of samples was calculated using the

| Table V. Probabilistic reachability in the controlled bouncing ball model |
|---|
| $#$ | $k$ | Probability interval | $CPU$ |
| 1 | 2 | $[0.2049030211646857, 0.2049030221625531]$ | 6,906 |

$k$ = number of discrete transitions, $CPU$ = CPU time in seconds
Chernoff–Hoeffding bound \( \delta \) with \( \epsilon = 0.01 \) and \( c = 0.99 \), which returned \( N = 23,026 \).

| \( k \) | \( P \) | \( CI \) | \( CPU \) |
|---|---|---|---|
| 2 | 0.2048120 | [0.1948120, 0.2148120] | 10.915 |

\( k \) = number of discrete transitions, \( P \) = probability estimate, \( CI \) = confidence interval and \( CPU \) = CPU time in seconds. Sample size = 23,026.

IX. CONCLUSION AND FURTHER WORK

We gave a formal definition of hybrid systems with random initial parameters, and the associated bounded probabilistic \( \delta \)-reachability problem. We showed how to verify a subclass of hybrid systems with explicit continuous dynamics and a single random initial parameter, and we presented results of verification for an example of such a system. Next we considered a larger class of hybrid systems involving a random initial parameter and dynamics that is represented via ordinary differential equations. This motivated us to implement a validated procedure for integration of functions over an arbitrary Borel set. In particular, we implemented an interval-based version of Simpson’s integration rule and applied it for solving the bounded probabilistic \( \delta \)-reachability problem. We remark that our technique will output a small interval which is guaranteed to contain the probability that the system reaches the unsafe region.

We believe that our technique can be fruitfully utilised for model checking \( \delta \) actual implementations of probabilistic algorithms. In particular, by adopting a similar approach as with bounded model checking for C code \( \delta \), temporal logic properties over programs would be transformed into SMT formulae by ‘unrolling’ the program source. Of course, these programs should satisfy suitable conditions such as, e.g., having finite loops only. We think that our approach would be useful for verifying embedded software implementing nonlinear control laws in (continuous) environments described by complex ODEs.

Also, our technique can produce extremely accurate results, returning intervals of the order of \( 10^{-8} \) and smaller even when computing very small probabilities (see, e.g., the thermostat model studied). This represents a significant advantage with respect to verification techniques based on Monte Carlo approaches, such as statistical model checking \( \delta \). In fact, it is well-known that Monte Carlo methods suffer from the rare-event problem: to estimate reliably very small probabilities, extremely large sample sizes (i.e., system simulations) are needed \( \delta \). \( \delta \).

In the future we plan to explore the application of our technique for probabilistic software verification. Furthermore, we plan to tackle an even larger class of hybrid systems. A first extension is to allow multiple random parameters. Solving \( \delta \)-reachability problems with multiple parameters involves multidimensional validated integration. A second extension is to allow probabilistic jumps in the system (discrete) dynamics. In such systems, transitions between modes may depend in general on the continuous state of the system. Finally, there is a big class of hybrid systems whose continuous state can randomly change over the time. Such dynamics is usually defined by a system of stochastic differential equations (SDE). These find much application in the modelling of complex cyber-physical systems, e.g., modelling of white noise in control systems. Hence, solving bounded \( \delta \)-reachability in general stochastic hybrid systems requires a validated SDE solver.

ACKNOWLEDGMENT

This work has been supported by award N00014-13-1-0090 of the US Office of Naval Research.

REFERENCES

[1] A. Tiwari, “Formal semantics and analysis methods for Simulink Stateflow models,” SRI International, Tech. Rep., 2002, http://www.csl.sri.com/~tiwari/stateflow.html.
[2] A. Agrawal, G. Simon, and G. Karsai, “Semantic translation of Simulink/Stateflow models to hybrid automata using graph transformations,” Electron. Notes Theor. Comput. Sci., vol. 109, pp. 43–56, 2004.
[3] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho, “Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems,” in Hybrid Systems, 1992, pp. 209–229.
[4] R. Alur and D. L. Dill, “Automata for modeling real-time systems,” in ICALP, 1990, pp. 322–335.
[5] D. Richardson, “Some undecidable problems involving elementary functions of a real variable,” J. Symb. Log., vol. 33, no. 4, pp. 514–520, 1968.
[6] S. Gao, J. Avigad, and E. M. Clarke, “Delta-decidability over the reals,” in LICS, 2012, pp. 305–314.
[7] ——, “Delta-complete decision procedures for satisfiability over the reals,” in IJCAR, 2012, pp. 286–300.
[8] S. Gao, S. Kong, and E. M. Clarke, “dReal: An SMT solver for nonlinear theories over the reals,” in CADE, 2013, pp. 208–214.
[9] K.-I. Ko, Complexity Theory of Real Functions. Birkauser, 1991.
[10] M. Davis and H. Putnam, “A computing procedure for quantification theory,” J. ACM, vol. 7, no. 3, pp. 201–215, Jul. 1960.
[11] F. Benhammou and L. Granvilliers, “Continuous and interval constraints,” in Handbook of Constraint Programming, F. Rossi, P. van Beek, and T. Walsh, Eds. Elsevier, 2006, ch. 16.
[12] S. Gao, S. Kong, W. Chen, and E. M. Clarke, “Delta-complete analysis for bounded reachability of hybrid systems,” CoRR, vol. arXiv:1404.7171, 2014.
[13] S. Galdino, “Interval integration revisited,” Open Journal of Applied Sciences, vol. 2, no. 4B, pp. 108–111, 2012.
[14] M. Lerch, G. Tischler, J. W. V. Gudenberg, W. e. Hofschuster, and W. Krämer, “FILIB++, a fast interval library supporting containment computations,” ACM Trans. Math. Softw., vol. 32, no. 2, pp. 299–324, Jun. 2006.
[15] W. Hoeffding, “Probability inequalities for sums of bounded random variables,” J. Amer. Statist. Assoc., vol. 58, no. 301, pp. 13–30, 1963.
[16] P. J. Mosterman, J. Zander, G. Hamon, and B. Denckla, “Towards computational hybrid system semantics for time-based block diagrams,” in 3rd IFAC Conference on Analysis and Design of Hybrid Systems (ADHS’09), 2009, pp. 376–385.
[17] E. M. Clarke and E. A. Emerson, “Design and synthesis of synchronization skeletons using branching-time temporal logic,” in Logic of Programs, ser. LNCS, D. Kozen, Ed., vol. 109, pp. 43–56, 2004.
[18] E. Clarke, D. Kroening, and F. Lerda, “A tool for checking ANSI-C programs,” in Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2004), ser. LNCS, vol. 2988, 2004, pp. 168–176.
[19] H. L. S. Younes and R. G. Simmons, “Statistical probabilistic model checking with a focus on time-bounded properties,” Inf. Comput., vol. 204, no. 9, pp. 1368–1409, 2006.
[20] R. Y. Rubinstein and D. Kroese, Simulation and the Monte Carlo Method. Wiley, 2008.
[21] P. Zuliani, C. Baier, and E. M. Clarke, “Rare-event verification for stochastic hybrid systems,” in HSCC, 2012, pp. 217–226.