Confidence Bounds for Statistical Model Checking of Probabilistic Hybrid Systems

Christian Ellen[1], Sebastian Gerwinn[1], and Martin Fränzle[1,2]

[1] OFFIS, R&D Division Transportation, Escherweg 2 - 26121 Oldenburg - Germany
{ellen,gerwinn}@offis.de
[2] Carl von Ossietzky Universität, 26111 Oldenburg - Germany
fraenzle@informatik.uni-oldenburg.de

Abstract. Model checking of technical systems is a common and demanding task. The behavior of such systems can often be characterized using hybrid automata, leading to verification problems within the first-order logic over the reals. The applicability of logic-based formalisms to a wider range of systems has recently been increased by introducing quantifiers into satisfiability modulo theory (SMT) approaches to solve such problems, especially randomized quantifiers, resulting in stochastic satisfiability modulo theory (SSMT). These quantifiers combine non-determinism and stochasticity, thereby allowing to represent models such as Markov decision processes. While algorithms for exact model checking in this setting exist, their scalability is limited due to the computational complexity which increases with the number of quantified variables. Additionally, these methods become infeasible if the domain of the quantified variables, randomized variables in particular, becomes too large or even infinite. In this paper, we present an approximation algorithm based on confidence intervals obtained from sampling which allow for an explicit trade-off between accuracy and computational effort. Although the algorithm gives only approximate results in terms of confidence intervals, it is still guaranteed to converge to the exact solution. To further increase the performance of the algorithm, we adopt search strategies based on the upper bound confidence algorithm UCB originally used to solve a similar problem, the multi-armed bandit. Preliminary results show that the proposed algorithm can improve the performance in comparison to existing SSMT solvers, especially in the presence of many randomized quantified variables.

1 Introduction

In safety analysis, one is often interested in guaranteeing certain behavioral properties of complex systems. Such systems are usually described using hybrid automata, which are capable of expressing the continuous dynamics of an environment using differential equations, together with discrete/continuous controllers. As the exact dynamics may not be known, these hybrid automata can contain non-deterministic choices, which have to be resolved. Additionally, due to environmental influences or failure probabilities, the system is likely to be

M. Jurdziński and D. Ničković (Eds.): FORMATS 2012, LNCS 7595, pp. 123–138, 2012.

exposed to stochastic type of non-determinism leading to probabilistic hybrid systems[1]. Safety properties for these systems can be formalized as reachability problems, that is, unsafe sets of states must not be reached. For the corresponding verification, a common technique is to use bounded model checking [1,2], which evolves the dynamics of the system up to a given number of transitions and checks the reachability for this unrolled depth. As transitions usually reflect only the switching decision, the reachability problem still has to respect the continuous dynamics of the system. The formalism of Satisfiability Modulo Theories (SMT) together with the corresponding solvers can in turn be used to solve this remaining satisfiability problem.

Recent developments with Conflict-Driven Clause Learning (CDCL) solvers enabled the analysis of large hybrid systems by learning conflicts which provide information from one path about a set of other paths. Formally, these hybrid systems can be modeled and analyzed using a combination of SMT and bounded model checking [3,4]. Analogous to Markov decision processes, it still remains to decide which transition path to evaluate in the presence of probabilism and non-determinism. To this end the formalism of SMT can be extended to Stochastic Satisfiability Modulo Theory (SSMT) which contains quantifiers that allow the encoding of different type of transitions [5]. Such SSMT solvers iterate the tree of possible assignments to the quantified variables and solve the resulting SMT problems at the leaves. The exponential size of this decision tree is one of the main problems for SSMT solvers. Therefore, methods are needed which search the tree efficiently and do not expand the tree completely.

Mathematically, the probability of satisfaction can also be formulated as a nested optimization. In this formulation, the problem of one existential followed by a randomized quantifier is known as the multi-armed bandit problem [6], where bandits are choices of the existential quantifier and expected rewards are the averages from the randomized quantifiers. For this problem, there exists a number of algorithms, most notably the Upper Bound Confidence algorithm (UCB), which has been proven to solve the multi-armed bandit problem in a minimax-optimal way [7].

In this paper, we present an algorithm which combines a sampling based approach for the generation of confidence intervals with the exploitation-scheme of the UCB-algorithm. The resulting algorithm allows for an explicit trade-off between accuracy (desired confidence level and precision in terms of the width of the confidence interval) and efficiency (solving of SMT formulas at the leaves of a decision tree). Additionally, due to the sampling of random variables, the algorithm is also applicable for randomized quantifiers with large or even infinite domains.

This paper is organized as follows. In Section 2 we give a short introduction to the SSMT-formalism and briefly review related work on SSMT-solvers as well as the relevant statistics literature followed by the description of the proposed algorithm including the bound propagation and the selection rules. In Section 3

[1] As an illustrative example for such a system, we use a simple cooling controller throughout this paper, see Figure 1.

we evaluate the algorithm on randomly generated SSMT formulas and on the example hybrid system in Figure 1. We conclude and give an outlook to future work in Section 4.

2 Methods

As mentioned previously, we are interested in satisfiability problems concerning probabilistic hybrid systems. which we illustrate with a simple example in Figure 1. For this particular system, we might be interested in guaranteeing that the probability of overheating is lower than a given threshold. To this end, we use satisfiability modulo theory (SMT) to formalize the satisfiability problem, given a particular path of transitions/decisions $(x_1, \ldots, x_n)$ obtained by unrolling the dynamics for a given number n of transitions. Note that this does not necessarily imply that the time within a state is fixed. For the example in Figure 1, transitions are only possible between `cooling` and `not cooling`. We denote the SMT formula which indicates the satisfiability for the given transition by $\phi(x_1, \ldots, x_n)$. The main task for the SMT solver is then to determine the existence of a satisfying variable assignment for a given SMT formula $\phi(x_1, \ldots, x_n)$. In the cooling example, this corresponds to an assignment of a temperature trajectory, given a state and a starting temperature.

SSMT is an extension of (SMT) [4] consisting of a decision problem for first-order logical formulas over a given background theory (*e.g.,* the arithmetic theories over real numbers, including multiplication). Specifically, an SSMT formula Φ extends an SMT formula ϕ by adding a prefix of quantified variables $Q_1 X_i, ..., Q_n X_n$. Every quantifier Q_i of the prefix binds one variable X_i of ϕ and

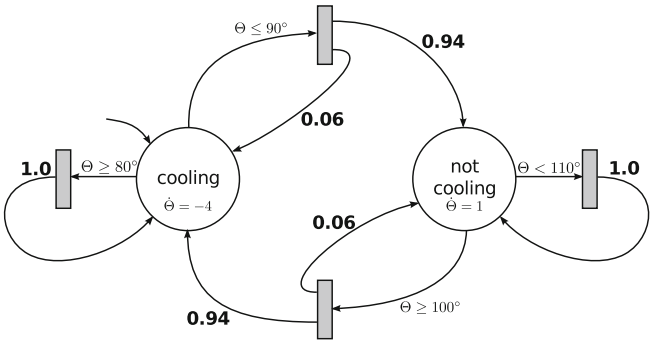

Fig. 1. Example of a probabilistic hybrid system modeling a simple cooling system. A cooling device can either be in the state `cooling` or `not cooling`. Withing the cooling state, the temperature Θ is decreased constantly whereas in the `not cooling` state, the temperature rises. Bold numbers at the edges reflect transition probabilities for the given probabilistic transitions, which can be activated once the guards (inequalities) hold. For this system, we might be interested in the probability of overheating (*e.g.,* $\Theta > 115°$) within a given time-frame.

is either randomized ($\bowtie_{X_i}$), existential ($\exists_{X_i}$), or universal ($\forall_{X_i}$). Every quantified variable has a finite domain: $\mathcal{X}_i$. In the randomized case, every value $x_j \in \mathcal{X}_i$ is associated with a probability $P(X_i = x_j)$, modeling the likelihood that the corresponding transition is chosen. The other quantifier types model different ways to resolve non-determinism: $\exists$ by maximizing the probability of satisfying the remaining formula over all domain values and analogously $\forall$ by minimizing the probability (see definition 1).

Definition 1. *The semantics on an SSMT formula Φ are defined recursively using Q for the remainder for the quantifier prefix (cf. [5]):*

1. $P(\quad \epsilon : \phi \;) = 0$ *if ϕ is unsatisfiable.*
2. $P(\quad \epsilon : \phi \;) = 1$ *if ϕ is satisfiable.*
3. $P(\; \exists_{X_i} Q : \phi \;) = \max_{x \in \mathcal{X}_i} P\left(Q : \phi[X_i = x]\right).$
4. $P(\; \forall_{X_i} Q : \phi \;) = \min_{x \in \mathcal{X}_i} P\left(Q : \phi[X_i = x]\right).$
5. $P(\; \bowtie_{X_i} Q : \phi \;) = \sum_{x \in \mathcal{X}_i} P(X_i = x | X_{\setminus i}) \cdot P\left(Q : \phi[X_i = x]\right).$

Here, we used the shorthand notation $X_{\setminus i}$ for all variables except the i-th one. From definition 1, we see that the satisfiability problem can be written as a nested optimization-expectation problem. For example, if Φ consists of one existential quantifier followed by a randomized one, the satisfiability problem can be written as follows:

$$P(\exists_x \bowtie_y \phi(x,y)) = \max_{x \in \mathcal{X}} \mathbb{E}_{y|x}\left[P(\phi(x,y))\right] = \max_{x \in \mathcal{X}} \left(\sum_{y \in \mathcal{Y}} P(\phi(x,y)) P(y|x) \right) \quad (1)$$

For the cooling example, this would correspond to first selecting a transition from a cooling state to one of the randomized states (rectangles in Figure 1) and then choosing a transition at random, according to the probabilities (bold numbers). The existential quantifier corresponds to a pessimistic choice of transitions in terms of consequences with respect to overheating. Note that in equation (1), we used $P(y|x)$ to indicate that the probability distribution associated with the random variable Y potentially depends on the value of other variables within the prefix. Although it is straightforward to extend the formalism developed in this paper to this general case, we assume independence of the different variables to simplify the notation.

Equation (1) suggests we should approximate the expectation via a sampling based scheme, if the set $\mathcal{Y}$ is too large. If we use the average of samples generated according to $P(y|x)$, we observe only a noisy estimate $\hat{\mathbb{E}}_{y|x}$ of the true underlying function $\mathbb{E}_{y|x}$. By using confidence intervals for this estimation, we obtain an uncertainty estimate for the expectation, *i.e.,* randomized quantifier which has then to be propagated through other quantifiers to obtain an overall uncertainty estimate on Φ. Computationally, we are interested in an efficient way of calculating equation (1), that is to efficiently search for promising x to evaluate.

A common representation of an SSMT formula uses a decision tree for the variables in the quantifier prefix, where the nodes are the variables (in order of their occurrence in the prefix) and the decisions are the domain values. The leaves of the tree are replications of ϕ, where every quantified variable is substituted according to the path in the decision tree. Therefore, every leaf represents an SMT problem of its own. As solving these SMT problems at the leaves of the decision tree is a time-consuming problem, most existing work focuses on minimizing the number of evaluations of the leaves by carrying information from one leaf-evaluation to another by using conflict learning.

In the following, we aim at calculating confidence bounds l, u, such that the following holds:

$$P(l \leq P(\Phi) \leq u) \geq 1 - \alpha \tag{2}$$

Where α is a given confidence level, which specifies the quality of the calculated bounds. Here, the outer probability originates from random samples generated during the execution of the algorithm and play the role of classical confidence intervals, while the inner probability is the quantity we wish to estimate, namely the probability of satisfaction.

2.1 Related Work

In the following, we briefly review existing work on SSMT solving on the one hand and on the corresponding statistics literature on the other hand.

SSMT Solving. Based on the iSAT[8] algorithm for SMT problems, an algorithm called SiSAT [5] has been developed to solve SSMT problems efficiently. It implements a fully symbolic solving procedure based on the traversal of the prefix tree, using extended CDCL procedures and pruning rules. The computed probability of satisfaction comes with absolute certainty, that is SiSAT terminates, if the probability is guaranteed to be larger than a given threshold θ or it has been computed exactly. The pruning rules allow SiSAT to ignore parts of the quantifier tree if the outcome of the decisions could be inferred or has no impact on the result (*e.g.,* if the target threshold has already been exceeded or cannot be reached anymore). Otherwise, the algorithm has to perform an exhaustive search over the state space. Due to this exhaustive search, the number of leaves in the tree – and hence the number of SMT problems to solve – depends exponentially on the number of quantified variables in the prefix. Although SiSAT has to examine exponentially many leaves in the worst case, the memory usage is still limited, as the tree is searched in a depth-first manner.

Statistical Model Checking. Inspired from classical hypothesis testing based on a set of data-samples, statistical model checking uses generated traces of the system under investigation to estimate if a given property holds. This can be done, if the system is given only in terms of a trace-generator [9,10,11], or if

more structure of the generative model is known (*e.g.,* a continuous time Markov chain [12]). Also, additional information, for example in terms of associated costs, can be incorporated [13,14]. Instead of hypothesis testing, one can also impose a prior distribution on the probability of satisfaction and update a Bayesian believe sequentially as new samples are drawn from the generative model [15]. However, all these approaches are only applicable, if no decisions are needed to resolve other than random non-determinism.

Stochastic Optimization. Problems in the form of equation (1) have been studied extensively in the statistics literature. This special case of an SSMT problem is known as the multi-armed bandit problem [6]. Algorithms which solve this problem approximately have been presented in [7] and also continuous extensions thereof [16]. The idea is to use the stochastic information obtained from a sample $\phi(y, x) \sim \phi(y, x)p(y|x)$ (or any noisy measurement of the expectation) to favor those points x which are more likely to attain the maximal expectation. Due to the sample-based nature of such information, one cannot guarantee hard bounds for the probability of satisfaction. However, even in the finite sample-size regime soft-bounds in terms of confidence intervals are available that are based on Hoeffding's inequality [17]. Extensions to problems similar to nested quantifiers are available for Markov Decision Processes, most notably the upper bound confidence algorithm for trees UCT [18,19]. In the UCT case, the search tree is sequentially expanded if needed. In this roll-out based technique, an estimate of the probability of satisfaction is needed, when a path is not completely traversed to the leaves and hence only parts of the variables are assigned specific values. Additionally, certain drift conditions have to be guaranteed and even if they hold, only asymptotic bounds for the root node are available.

2.2 Bound Propagation

In this section, we describe how the bounds for the full SSMT formula can be obtained from bounds for the intermediate values at the leaves of the decision tree. First, we show how to obtain confidence bounds for the probability of satisfaction at the root node using bound propagation, and then present search strategies for an efficient way to decrease these bounds.

As mentioned previously, we are interested in calculating bounds on the probability of satisfaction. To obtain these bounds (equation (2)), we propagate bounds together with their confidence level from the leaves of the decision tree up to the root (see Figure 2). As the bounds together with confidence levels are propagated from leaves to the root, at each point during the computation, we need bounds as well as confidence levels at the leaves. Without loss of generality, the last quantifier in the decision tree is a randomized quantifier, otherwise we can shift all existential and universal quantifiers into the formula at the leaves as the probability of satisfaction is either 0 or 1. However, in this case we use the SiSAT algorithm to compute these remaining SSMT problems. Therefore, we can obtain the bounds and confidence levels for the leaves of the decision tree by drawing samples from the probability distribution associated with the

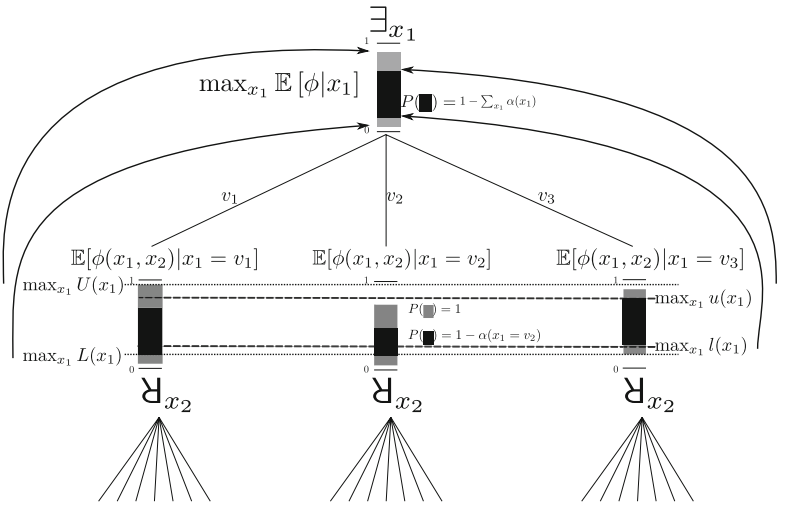

Fig. 2. Illustration of the bound propagation through the decision tree. Confidence intervals with confidence level $\alpha = \alpha(x_1)$ are obtained at the leaves (here for the random variable x_2) via equation (3). These are propagated to the next level to derive confidence intervals with worse confidence levels $\sum \alpha$. In this illustrative example $v_i, i = 1, 2, 3$ are the possible values for the variable x_1. Additionally to the α-confidence intervals, we could also propagate guaranteed ($\alpha = 1$) confidence bounds.

randomized quantifier and apply a Hoeffding type of inequality similar to the UCT algorithm. If we assume that the domain of the randomized quantifier is finite, we could also get tighter bounds by drawing samples without replacement and using the appropriate inequality (see [20]). However, as such sampling without replacement increases memory usage for large domains, we use the version with replacement [17]. Specifically, we set for the lower l and upper u bounds at the leaves, which depend on the values set for other variables set earlier in the decision tree $x_1, \ldots, x_{k-1}$:

$$
\begin{aligned}
l(x_{1:k-1}) := \hat{P}(\Phi|x_{1:k-1}) - w, \qquad & u(x_{1:k-1}) := \hat{P}(\Phi|x_{1:k-1}) + w \\
\hat{P}(\Phi|x_{1:k-1}) &= \frac{1}{m} \sum_{x^i \sim P_k} P\left(\phi(x_1, \ldots, x_{k-1}, x_k = x^i)\right) \\
P(\Phi|x_{1:k-1}) &= \sum_{x_k \in \mathcal{X}_k} P_k(x_k) P\left(\phi(x_1, \ldots, x_{k-1}, x_k)\right),
\end{aligned}
\tag{3}
$$

where w specifies the uncertainty in the estimate $\hat{P}$ of the true (unknown) probability of satisfaction P. m is the number of samples x^i we have drawn from the probability distribution P_k of the random variable X_k. Here, we have written $P_k(x_k)$ as a shorthand for $P(X_k = x_k|X_1 = x_1, \ldots, X_{k-1} = x_{k-1})$. Using these

setting and applying Hoeffding's inequality, we have for the confidence level of the specified width:

$$\begin{aligned} P\left(l(x_{1:k-1}) \leq P(\Phi|x_{1:k-1})\right) &\geq 1 - e^{\left(-2mw^2\right)} := 1 - \frac{1}{2}\alpha(x_1, \ldots, x_k) \\ P\left(P(\Phi|x_{1:k-1}) \leq u(x_{1:k-1})\right) &\geq 1 - e^{\left(-2mw^2\right)} := 1 - \frac{1}{2}\alpha(x_1, \ldots, x_k) \end{aligned} \tag{4}$$

where α is the confidence level, which depends on the number of samples and the width w. The bound in equation (4) makes a statement about the confidence that we obtain as a consequence of the random sampling. As mentioned previously, we could, obtain a "hard bound" (that holds with probability 1) on the probability of satisfaction if we additionally make use of the explicitly known structure of the randomized quantifier in terms of the associated probability distribution. The confidence bounds apply for the random sampling at the leaves. However, we are interested in calculating bounds for the root of the decision tree. To this end, we state the following propagation rules from level k to the above level $k-1$:

$$b(x_1, x_2, \ldots, x_{k-1}) = \begin{cases} \max_{x_k \in \mathcal{X}_k} b(x_1, \ldots, x_{k-1}, x_k) & \text{if } Q_k = \exists_{X_k} \\ \min_{x_k \in X_k} b(x_1, \ldots, x_{k-1}, x_k) & \text{if } Q_k = \forall_{X_k} \\ \sum_{x_k \in X_k} P(x_k) b(x_1, \ldots, x_{k-1}, x_k) & \text{if } Q_k = \text{ꓤ}_{X_k} \end{cases} \tag{5}$$

$$\alpha(x_1, \ldots, x_{k-1}) = \sum_{x_k \in \mathcal{X}_k} \alpha(x_1, \ldots, x_{k-1}, x_k) \tag{6}$$

Here, b are the bounds, which are function of the decision variables and could be either lower or upper bounds that we would like to propagate to the preceding level.

This process of propagating intervals is illustrated in Figure 2. In the case of Figure 2 we have two quantifiers, an existential and a randomized one. Given that we have drawn samples for the randomized quantifier, we can calculate confidence intervals via equation (3),(4) for a given confidence level.We can then combine these confidence intervals to obtain confidence intervals for the value at the existential quantifier using equation (5). Note that the confidence level for the "soft" bounds is worse than those at the lower level. In fact, if we want to reach a desired confidence level at the root of the decision tree, *e.g.,* 95%, we have to impose a much higher confidence level at the leaves. For example, in the case of Figure 2, we would have to impose $\alpha(x_1) = \frac{0.05}{3}$ to get the desired 95% confidence interval at the top.

We now show that the bounds calculated in equation (5) indeed hold for the confidence level in equation (6). We start with the bounds in the case of an existential quantifier. In this case, we can simplify the notation, by ignoring variables that are fixed for the propagation and therefore writing $u(x), l(x)$ for the upper and lower confidence bounds for the level that we would like propagate and $\alpha(x)$ for the corresponding confidence level at the lower level. Similarly, we write $\Phi(x)$ for the true (unknown) values at the leaves of the existential quantifier. With this notation, we have:

Lemma 1. *Let the lower and upper confidence bound hold for each child x of an existential quantifier,* i.e., $P\left(\Phi(x) > u(x)\right) \leq \frac{1}{2}\alpha(x) \forall x$. *Then the following bound hold for the existential quantifier node:*

$$P\left(\max_x l(x) \leq \max_x \Phi(x) \leq \max_x u(x)\right) \geq 1 - \sum_x \alpha(x)$$

Proof. From the lower level, we know $P\left(\Phi(x) > u(x)\right) \leq \frac{1}{2}\alpha(x)$ for each x individually. Applying a simple union bound (cf. [21]), we find:

$$\begin{aligned} P\left(\max_x \Phi(x) > \max_x u(x)\right) &\leq P\left(\exists_x : \Phi(x) > u(x)\right) \\ &= \sum_x \underbrace{P\left(\Phi(x) > u(x)\right)}_{\leq \frac{1}{2}\alpha(x)} - \underbrace{P\left(\forall_x : \Phi(x) > u(x)\right)}_{\geq 0} \qquad (7) \\ &\leq \frac{1}{2}\sum_x \alpha(x) \end{aligned}$$

Similarly, we get $P\left(\max_x \Phi(x) < \max_x l(x)\right) \leq \frac{1}{2}\sum_x \alpha(x)$. Combining upper and lower bound completes the proof. □

By maximizing the negative value instead of the minimization for the universal quantifier and inverting the role of upper and lower bound, the same argument can be made for the case of an universal quantifier. For the case of a randomized quantifier, we have:

Lemma 2. *Using the same notation as in Lemma 1, the following bound holds.*

$$P\left(\sum_x P(x)l(x) \leq \sum_x P(x)\Phi(x) \leq \sum_x P(x)u(x)\right) \geq 1 - \sum_x \alpha(x)$$

Proof. As $\forall_x : \Phi(x) \leq u(x) \Rightarrow \sum_x P(x)\Phi(x) \leq \sum_x P(x)u(x)$ $(*)$:

$$\begin{aligned} P\left(\sum_x P(x)\Phi(x) > \sum_x P(x)u(x)\right) &= 1 - P\left(\sum_x P(x)\Phi(x) \leq \sum_x P(x)u(x)\right) \\ &\overset{*}{\leq} 1 - P\left(\forall_x : \Phi(x) \leq u(x)\right) = P\left(\exists_x : \Phi(x) > u(x)\right) \overset{(7)}{\leq} \frac{1}{2}\sum_x \alpha(x) \end{aligned}$$

The lower bounds follow analogously. □

2.3 Search Strategies

So far, we have not considered any search strategies for the optimized selection of children to expand at a quantifier node in order to increase the confidence at the root of the decision tree. By selecting the paths to evaluate in a strategic manner, we aim for gaining as much information as possible about the probability of satisfaction of the SSMT-formula at hand. To this end, we propose several simple

search strategies. As an existential quantifier corresponds to a maximum operation (see definition 1), we select the child with the maximal upper confidence bound for exploration. Analogously, we select the child with the minimal confidence bound for universal quantifiers. This strategy is known under the upper confidence bound algorithm. However, additionally to the lower and upper bound from equation (5), we can calculate the upper bound, according to the number of times, the current node x has been visited $n(x)$, compared to the overall number of samples within the current quantifier n analogously to the UCT-algorithm:

$$\mu(x)\pm = \sqrt{\frac{2\log(n)}{n(x)}}, \qquad \mu(x) := l(x) + \frac{u(x) - l(x)}{2}. \tag{8}$$

Note that using the bounds from equation (5) typically leads to smaller bounds compared to the UCB, as it uses more information on the structure below the node. For leaf-nodes, however, they are identical. For a randomized quantifier, we propose the following two strategies:

1. Sample a child to explore according to the associated probability distribution
2. Construct a probability distribution by weighting the probability of each child with the width of its confidence interval

By using combinations of these selection rules, we obtain 4 different selection rules in total.

2.4 Algorithmic Description

Although we have presented the basic components – initial bounds, bound propagation, and search strategies – of our proposed algorithm in the previous sections, we present a more detailed algorithmic description in this section. In particular, this includes how deductions obtained from the solver at the leaves of the decision tree can be used for pruning and incorporated into the calculation of the confidence bounds. We explain the three different phases (see Algorithm 1: selection phase, sampling phase, and propagation phase) in more detail in the following. First, we partition the decision tree into a three parts: A trailing part containing all trailing non-randomized quantifiers, a (non-empty) set of randomized quantifiers, and a leading part which may contain any quantifier. To lighten the description, we collapse the randomized quantifiers in the second part to a single randomized quantifier with more leaves and expand the SMT formula by adding the trailing non-randomized quantifiers to the formula. Hence, we can assume for the following a decision tree/SSMT formula in which the last quantifier is a randomized one. Additionally, to obtain the desired confidence level at the root of the decision tree, we calculate the necessary α for the randomized part by counting the number of leaves L of the first part of the decision tree:

$$\alpha(x_1, \ldots, x_n) = 1 - \frac{1}{L}(1 - \alpha)$$

Algorithm 1. Confidence Bound Generation and Propagation

Data: Quantifier Prefix $Q_1, \ldots, Q_n$,
SMT Forumla $\phi(x_1, \ldots, x_n)$,
confidence level α
Result: Confidence interval $[\underline{p}, \overline{p}]$ for satisfiability of: $Q_1, \ldots, Q_n \phi(x_1, \ldots x_n)$ with given confidence
while *terminal condition not reached* **do**
  for $i \leftarrow 1$ **to** $n-1$ **do** ; `// Selection phase`
    Select x_i for Variable $X_i | x_1, \ldots, x_{i-1}$;
  end
  for $k \leftarrow 1$ **to** m **do** ; `// Sampling phase: draw` m `samples`
    sample $x_n^k \sim P_n(X_n)$;
    $\text{sat}_k \ni \{0,1\} = \text{solveSMT}(\phi(X_1 = x_1, \ldots, X_n = x_n^k))$
  end
  update bounds for $u(x_1, \ldots, x_{n-1})$ using eq. (3);
  for $i \leftarrow n-1$ **to** 1 **do** ; `// Propagation phase`
    collect $l(x_1, \ldots, x_i, X_{i+1} = x), u(x_1, \ldots, x_i, X_{i+1} = x)$ for all x;
    use deductions to collapse some of these bounds to 0;
    use equation (5) to calculate $l(x_1, \ldots, x_i), u(x_1, \ldots, x_i)$;
  end
end

Selection Phase. Within the selection phase, the quantifiers of first part are traversed and a new evaluation is selected based on the type of quantifier and the valuations so far:

$\exists, \forall$ These quantifiers are decided by selecting the value with the largest or smallest confidence bounds for existential or universal quantifiers respectively. Either the confidence bounds are used directly or they are computed based on the UCB-rule, see equation (8). Note that for some of the variables the bounds might be collapsed due to deduction from DPLL solvers for the SMT formula. The corresponding values will be ignored for the selection rule.

ꓤ As mentioned in section 2.3, there are two available selection rules for randomized quantifiers. To construct a weighted version of the probability distribution associated with the randomized variable X_k, we use the following:

$$P_s(x_k) = \frac{P_w(x_k)}{\sum_x P_w(x)}, \qquad P_w(x_k) = (u(x_k) - l(x_k))\, P(x_k) \tag{9}$$

Sampling Phase. In this phase, we generate m samples for last randomized quantifier. For each generated sample, we solve the remaining SMT formula and use the resulting data to calculate initial bounds or update the confidence bounds, if the current node as already been sampled in the past and has been selected for further evaluation.

Propagation Phase. After updated confidence bounds for the current path have been generated in the selection phase, the new information needs to be propagated to the root node. The updated bounds are propagated according to equation (5) and the number of visits for each existential and universal quantifier is increased by one. Additional to the evidence from the generated samples, DPLL solvers provide extra information in terms of deductions indicating subtrees which are guaranteed to evaluate to *false*. This supplemental information can be incorporated in the propagation of confidence bounds by collapsing the corresponding bound of the subtree to 0. As bound updates only occur on the current path, the influence of a deduction is only taken into account for this path. The effects of deductions for other subtrees not yet sampled will only be calculated once this subtree is visited.

Terminal Condition. As the algorithm never reaches a point interval with 100% confidence, we have to decide when to abort the sampling procedure. By adjusting combinations of confidence level and desired width of the computed confidence interval, we can set an explicit trade-off between accuracy (width of the interval together with desired confidence) and speed (number of SMT-evaluations). Sometimes we might be interested in verifying that the probability of satisfaction is larger, or lower than a certain threshold, independently of the width of the confidence interval. Therefore, we use the following two possibilities: Terminate if either the confidence interval for the full formula has reached a given width, or if the lower (upper) confidence bound has crossed a predefined threshold.

3 Evaluation

The computational complexity of Monte-Carlo sampling based methods usually do not scale with the dimensionality of the probability distribution at hand, but scale only with the number of samples (see equation (3)). Therefore, we expect the bound propagation to work best when the number of randomized quantified variables for the last part is large, as we obtain the initial confidence bound within this part. Additionally, as we do not use the full information available via deductions of the SMT-solver at the leaves of the decision tree, we expect existing solvers like SiSAT [5] to outperform the bound propagation algorithm in cases of only small proportions of randomized quantified variables. To test these hypotheses, we first evaluated the bound propagation on scenarios of randomly generated SMT formulas. Specifically, we used the same generation mechanism as in [22] called `makewff` to generate random formulas with 24 variables, 20 clauses and 3 variables per clause. By using these settings, we generated formulas which are likely to be satisfiable for some yet not all assignments. For the quantifier prefix, we tested two different settings, one with a large proportion of randomized quantifiers and one with a small proportion. For the first setting, we constructed a quantifier prefix with a high fraction of randomized quantifiers consisting of twice an $\exists - \mathbb{R}$ pair followed by 21 randomized quantifiers. The second setting is particularly disadvantageous for the bound propagation and consisted eight ($\exists - \exists - \mathbb{R}$) triplets. For each of these

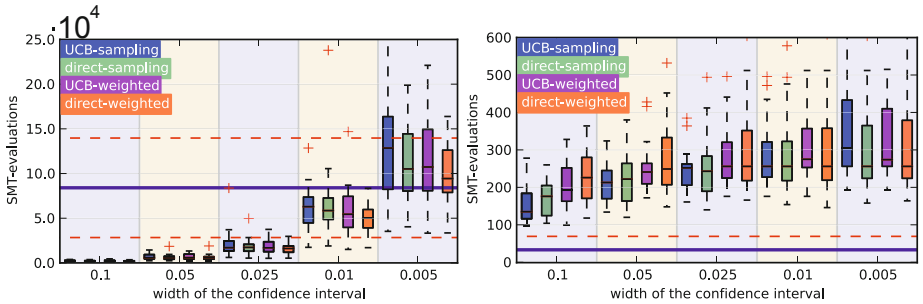

Fig. 3. Performance comparison as a function of the width of the confidence interval with fixed confidence level of 95%. **Left**: Results for the setting with high fraction of randomized quantifiers. **Right**: Disadvantageous setting with more existential quantifiers. The blue line shows the number of evaluations for the SiSAT algorithm (averaged across 16 randomly generated formulas) and in red the standard deviation over these repetitions. For each width we compared different selection rules, UCB stands for an UCB selection rule for the existential quantifier whereas 'direct' indicates a confidence bound selection rule on the confidence bounds according to equation (5). 'weighted' indicates that the probability distribution is weighed with the width of the confidence interval before a sample is drawn.

settings, we compared the SiSAT algorithm and the bound propagation in terms of the number of evaluations of the leaves, *i.e.,* number of solver calls averaged across 16 repetitions (as both the result as well the formula depend on stochastic quantities). For a fixed confidence level, the number of leaf-evaluations depends mainly on the width of the confidence interval used as termination condition. Therefore, we show the performance as a function of the width of the confidence interval in Figure 3. In the left panel, the results are shown for the advantageous setting of large proportions of randomized quantifiers, whereas in the right panel the disadvantageous setting is used. For the second setting (right panel) the probability of satisfaction can be inferred by only a few SMT-evaluations, as a large proportion of the tree can be pruned due to deductions made by the SMT solver. In fact, due to the maximum operation of the existential quantifier, only a few paths contribute to the root value, *i.e.,* it is sufficient to show the satisfiability of these few paths. As the bound propagation algorithm still samples some paths for the trailing randomized quantifier, it needs more SMT evaluations than the SiSAT algorithm. As the number of randomized variables is small, the sampling procedure is likely to sample the same valuation multiple times. These samples can be cached and hence no SMT solver needs to be evaluated. Therefore, we report only the number of actual SMT evaluations neglecting the evaluations, that can be obtained from the cache. For the first setting (left panel), both SiSAT and the four statistical variants need much more evaluations compared to the setting in the right panel, as in this case nearly all paths contribute to the root value. For this setting with a large proportion of randomized quantifiers, the bound propagation decreases the number of SMT evaluations tremendously compared to the SiSAT algorithm. If we fix, however, the width of the confidence

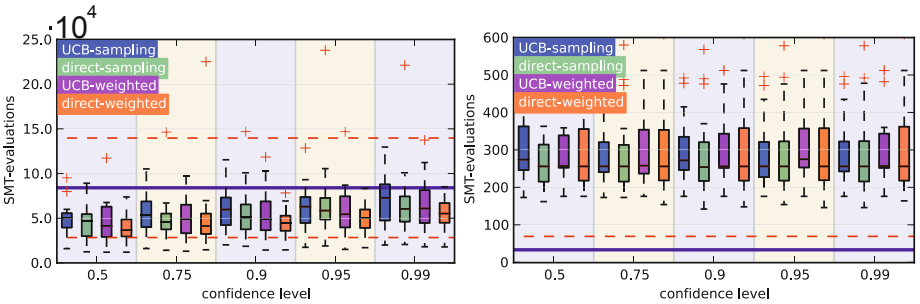

Fig. 4. Performance comparison as a function of the confidence level chosen. For all experiments the number of evaluations is counted until the confidence interval reached a width of ≤ 0.01. Left and right panels show the results for the different settings, analogous to Figure 3.

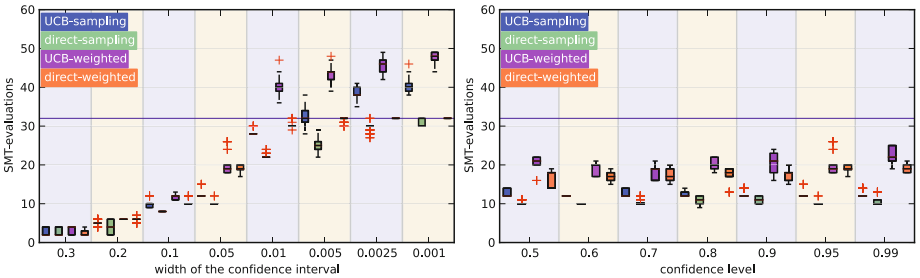

Fig. 5. Performance for the illustrative example of Figure 1. **Left**: Number of SMT evaluations with fixed level of confidence (95%) used as terminal condition. **Right**: Performance of the different search strategies as a function of the confidence level with fixed width of the confidence interval (0.05). For each combination of existential and randomized selection rule the statistics over 100 repetitions are plotted.

interval, the number of evaluations depends on the confidence level, see Figure 4. Analogous to Figure 3, the left and right panel show the results for the different proportions of the type of quantifiers. For this setting the same overall observation holds that the bound propagation gives a superior performance for the first set of quantifiers order (left panel). However, we note, that the width of the confidence interval has a much bigger impact on the number of SMT evaluations than the confidence level. Finally, we performed the same type of analysis for the illustrative example shown in Figure 1, the results for which can be found in Figure 5. The difficulty in this scenario can be increased by increasing the bounding depth in the BMC problem, *i.e.,* number of transition. The results plotted in Figure 5 are obtained by using a bounding depth of 8 steps. For this example, a similar behavior of the different algorithms can be observed. However, for this setting, much more information can be obtained through deductions, as can be seen by the small number of SMT evaluations needed to achieve the goal of the combination of interval width with a given confidence level. In the right panel

of Figure 5, we see that the number of SAT evaluations does not vary with the chosen confidence level. The actual confidence level is 100% due to deductions available during the processing, hence once the width of the confidence interval is below the chosen threshold, no more samples are needed to achieve the confidence level. Additionally, it can be observed that the direct confidence bound selection rule performs better than the UCB strategy, although the difference is not as big as in the previous setting.

4 Conclusion

We have presented an SSMT-solving algorithm based on statistical methods used for solving multi-armed bandit problems. As the algorithm allows to specify the desired accuracy in terms of confidence width and confidence level, the trade-off between accuracy and computational effort can be adjusted explicitly. By using the proposed search and sampling strategy, we were able to gain efficiency for certain SSMT problems. The improvement compared to existing SSMT-solvers (SiSAT) is larger, the higher the proportion of randomized quantifiers is within the tree. Indeed, the presented sampling based technique can also be extended to handle continuous valued random variables and thereby extends the model class which can be analyzed to hybrid systems including stochastic differential equations. Importantly, we can also make use of the pruning procedures, which are typically used in CDCL-based solvers. For the implementation, however, not all deductions obtained from these solvers can be exploited to prune the decision tree. Currently, we use the pruning rules to collapse the confidence intervals with confidence level α, although we could collapse the 100% confidence bounds thereby allowing to use worse confidence levels in other subtrees. Exploiting more of these pruning rules is subject to future research.

Acknowledgments. This research was supported by the European Commission under the project MoVeS, FP7-ICT-257005.

References

1. Biere, A., Cimatti, A., Clarke, E., Zhu, Y.: Symbolic Model Checking without BDDs. In: Cleaveland, W.R. (ed.) TACAS 1999. LNCS, vol. 1579, pp. 193–207. Springer, Heidelberg (1999)
2. Groote, J., van Vlijmen, S., Koorn, J.: The safety guaranteeing system at station hoorn-kersenboogerd. In: Proceedings of the Tenth Annual Conference on Systems Integrity, Software Safety and Process Security, Computer Assurance, COMPASS 1995, pp. 57–68. IEEE (1995)
3. Audemard, G., Bozzano, M., Cimatti, A., Sebastiani, R.: Verifying Industrial Hybrid Systems with MathSAT. Electronic Notes in Theoretical Computer Science 119(2), 17–32 (2005)
4. Fränzle, M., Herde, C.: HySAT: An efficient proof engine for bounded model checking of hybrid systems. Formal Methods in System Design 30(3), 179–198 (2007)

5. Teige, T., Eggers, A., Fränzle, M.: Constraint-based analysis of concurrent probabilistic hybrid systems: An application to networked automation systems. Nonlinear Analysis: Hybrid Systems (2010)
6. Robbins, H.: Some aspects of the sequential design of experiments. Bulletin of the American Mathematical Society 58(5), 527–536 (1952)
7. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2), 235–256 (2002)
8. Fränzle, M., Herde, C., Teige, T., Ratschan, S., Schubert, T.: Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure. Journal on Satisfiability, Boolean Modeling and Computation 1(3-4), 209–236 (2007)
9. Larsen, K.G., Skou, A.: Bisimulation through probabilistic testing. Information and Computation 94(1), 1–28 (1991)
10. Sen, K., Viswanathan, M., Agha, G.: Statistical Model Checking of Black-Box Probabilistic Systems. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 202–215. Springer, Heidelberg (2004)
11. Younes, H., Simmons, R.G.: Statistical probabilistic model checking with a focus on time-bounded properties. Information and Computation 204(9), 1368–1409 (2006)
12. Younes, H.L.S.: Ymer: A Statistical Model Checker. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 429–433. Springer, Heidelberg (2005)
13. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Wang, Z.: Time for Statistical Model Checking of Real-Time Systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 349–355. Springer, Heidelberg (2011)
14. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B., van Vliet, J., Wang, Z.: Statistical Model Checking for Networks of Priced Timed Automata. In: Fahrenberg, U., Tripakis, S. (eds.) FORMATS 2011. LNCS, vol. 6919, pp. 80–96. Springer, Heidelberg (2011)
15. Zuliani, P., Platzer, A., Clarke, E.M.: Bayesian Statistical Model Checking with Application to Stateflow/Simulink Verification. In: Johansson, K.H., Yi, W. (eds.) HSCC, pp. 243–252. ACM, Stockholm (2010)
16. Bubeck, S., Munos, R., Stoltz, G., Szepesvári, C.: X-armed bandits. Journal of Machine Learning Research 12, 1655–1695 (2011)
17. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association, 13–30 (1963)
18. Kocsis, L., Szepesvári, C.: Bandit Based Monte-Carlo Planning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 282–293. Springer, Heidelberg (2006)
19. Szepesvári, C.: In: Reinforcement Learning Algorithms for MDPs. John Wiley & Sons, Inc. (2010)
20. Serfling, R.J.: Probability inequalities for the sum in sampling without replacement. The Annals of Statistics 2(1), 39–48 (1974)
21. Srinivas, N., Krause, A., Kakade, S., Seeger, M.: Gaussian process optimization in the bandit setting: No regret and experimental design. In: Fürnkranz, J., Joachims, T. (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML 2010), pp. 1015–1022. Omnipress, Haifa (2010)
22. Littman, M.L., Majercik, S.M., Pitassi, T.: Stochastic boolean satisfiability. Journal of Automated Reasoning 27(3), 251–296 (2001)