# Verification Using Simulation*

Antoine Girard and George J. Pappas

Department of Electrical and Systems Engineering,
University of Pennsylvania,
Philadelphia, PA 19104
{agirard, pappasg}@seas.upenn.edu

**Abstract.** Verification and simulation have always been complementary, if not competing, approaches to system design. In this paper, we present a novel method for so-called metric transition systems that bridges the gap between verification and simulation, enabling system verification using a finite number of simulations. The existence of metrics on the system state and observation spaces, which is natural for continuous systems, allows us to capitalize on the recently developed framework of approximate bisimulations, and infer the behavior of neighborhood of system trajectories around a simulated trajectory. For nondeterministic linear systems that are robustly safe or robustly unsafe, we provide not only a completeness result but also an upper bound on the number of simulations required as a function of the distance between the reachable set and the unsafe set. Our framework is the first simulation-based verification method that enjoys completeness for infinite-state systems. The complexity is low for robustly safe or robustly unsafe systems, and increases for nonrobust problems. This provides strong evidence that robustness dramatically impacts the complexity of system verification and design.

## 1 Introduction

Given a system model and a desired specification, system designers rely on both analysis and simulation methods. Simulation-based approaches ensure that a finite number of user-defined system trajectories meet the desired specification. Even though computationally inexpensive simulation is ubiquitous in system design, it suffers from completeness as it is impossible or impractical to test all system trajectories. Furthermore, simulation-based testing is semi-automatic since the user must provide a large number of test cases. On the other hand, automated verification methods enjoy completeness by showing that all system trajectories satisfy the desired property. Despite great progress on verification tools for discrete software and hardware systems, the algorithmic complexity of verification tools makes them applicable to smaller scale problems.

The gap between simulation and verification is more extreme when considering systems with infinite states, such as continuous or hybrid systems.

* This research is partially supported by the Région Rhône-Alpes (Projet CalCel) and the NSF Presidential Early CAREER (PECASE) Grant 0132716.

J. Hespanha and A. Tiwari (Eds.): HSCC 2006, LNCS 3927, pp. 272–286, 2006.

Whereas traditional simulation techniques for discrete and continuous systems can be naturally extended for hybrid systems [1, 2, 3], verification techniques have been much more challenging to extend due to the complexity of computing reachable sets for continuous systems. This has resulted in a variety of computationally intensive approaches for hybrid system verification using predicate abstraction [4, 5], barrier certificates [6], level sets [7], and exact arithmetic [8]. Even though these approaches can handle low-dimensional hybrid systems, for the class of uncertain linear systems, promising scalable results have been obtained using zonotope computations [9].

In this paper, we present a novel method that bridges the gap between verification and simulation methods, enabling system verification using a finite number of simulations. This is achieved for so-called metric transition systems, that are transition systems that are equipped with metrics on the system state and observation spaces. Whereas choosing metrics may not be natural for purely combinatorial discrete problems, they are very natural for continuous and hybrid systems. Having a notion of distance between states and observations, enables us to build on the recently developed framework of approximate bisimulation metrics [10, 11, 12, 13]. Bisimulation metrics measure how far two states are from being bisimilar, thus enabling the quantification of error between trajectories originating from approximately bisimilar states.

Equipping transition systems with bisimulation metrics enables the development of a simulation-based verification algorithm by inferring the behavior of neighborhood of system trajectories around a simulated trajectory, resulting in more robust simulations. By appropriately sampling the set of initial states, we can verify or falsify the desired property for all system trajectories. The more robust the simulations, the less simulations we have to perform. The pre-computed bisimulation metric is used for automatically guiding the choice of trajectories that will be simulated.

For the class of metric transition systems generated by nondeterministic linear systems that are robustly safe or robustly unsafe, a completeness result is provided. Our framework is the first simulation-based verification method that enjoys completeness for continuous systems. Furthermore, we obtain an upper bound on the number of simulations required as a function of the distance between the reachable set of the system and the unsafe set. Naturally, the complexity of our approach is low for robustly safe or robustly unsafe systems, and increases for nonrobust problems. This provides strong evidence that robustness dramatically impacts the complexity of system verification and design.

## 2 Bisimulation Metrics for Transition Systems

We consider the class of metric transition systems defined as follows:

**Definition 1 (Metric transition system).** *A transition system with observations is a tuple* $T = (Q, \rightarrow, Q_0, \Pi, \langle\langle . \rangle\rangle)$ *that consists of:*

- *a (possibly infinite) set $Q$ of states,*
- *a transition relation $\rightarrow \subseteq Q \times Q$,*
- *a (possibly infinite) set $Q_0 \subseteq Q$ of initial states,*
- *a (possibly infinite) set $\Pi$ of observations,*
- *an observation map $\langle\langle . \rangle\rangle : Q \rightarrow \Pi$.*

*If $(Q, d_Q)$ and $(\Pi, d_\Pi)$ are metric spaces, then $T$ is called a metric transition system.*

A metric transition system is therefore a possibly nondeterministic transition system equipped with metrics for states and observations. A transition $(q, q') \in \rightarrow$ will be denoted $q \rightarrow q'$. The successor map is defined as the set valued map given by

$$\forall q \in Q,\ \mathrm{Post}(q) = \{q' \in Q |\ q \rightarrow q'\}\,.$$

We assume the set of initial values $Q_0$ is a compact subset of $Q$ and for all $q \in Q$, $\mathrm{Post}(q)$ is a compact subset of $Q$. A state trajectory of $T$ is a finite sequence of transitions, $q_0 \rightarrow \cdots \rightarrow q_k$, where $q_0 \in Q_0$. For $N \in \mathbb{N}$, $S_N(T)$ denotes the set of state trajectories of length less or equal to $N$. The reachable set of $T$ within $N$ transitions is the subset of $\Pi$ defined by:

$$\mathrm{Reach}_N(T) = \{\pi \in \Pi |\ \exists\ q_0 \rightarrow \cdots \rightarrow q_k \in S_N(T),\ \langle\langle q_k \rangle\rangle = \pi\}\,.$$

An important problem for transition systems is the safety verification problem which asks whether the intersection of $\mathrm{Reach}_N(T)$ with an unsafe set $\mathcal{U} \subseteq \Pi$ is empty or not. When considering metric transition systems, more robust (*i.e.* quantitative) versions of this property can be formulated:

**Definition 2.** *A metric transition system $T$ is robustly safe if there exists $\delta > 0$ such that*

$$\forall \pi \in \mathrm{Reach}_N(T),\ \mathcal{N}_\Pi(\pi, \delta) \cap \mathcal{U} = \emptyset,$$

*and is robustly unsafe if there exists $\delta > 0$ such that*

$$\exists \pi \in \mathrm{Reach}_N(T),\ \mathcal{N}_\Pi(\pi, \delta) \subseteq \mathcal{U},$$

*where $\mathcal{N}_\Pi(\pi, \delta)$ denotes the $\delta$-neighborhood of observation $\pi$ for the metric $d_\Pi$. The supremum of the set of $\delta$ such that one of these equations holds is called the coefficient of robustness of $T$. If $T$ is neither robustly safe nor robustly unsafe, we say that $T$ is not robust with respect to the safety property.*

*Remark 1.* Robustness with respect to the safety property is generic for metric transition systems. Indeed, $T$ is not robust with respect to the safety property if and only if the intersection of the interior of $\mathrm{Reach}_N(T)$ and $\mathcal{U}$ is empty while the intersection of their closure is not.

For systems with a finite number of states, the safety verification problem can be solved by exhaustive simulation of the transition system. Though effective for systems with a reasonable number of states, this approach becomes much more

computationally demanding when the number of states increases. For systems with a finite but large number of states, notions of systems refinement and equivalence, based on language inclusion, simulation and bisimulation relations [14], have been useful for simplifying the safety verification problem.

**Definition 3 (Bisimulation relation).** *A relation* $\sim \subseteq Q \times Q$ *is a bisimulation relation if for all* $q_1 \sim q_2$*:*

1. $\langle\langle q_1 \rangle\rangle = \langle\langle q_2 \rangle\rangle$,
2. *for all* $q_1 \to q_1'$*, there exists* $q_2 \to q_2'$*, such that* $q_1' \sim q_2'$,
3. *for all* $q_2 \to q_2'$*, there exists* $q_1 \to q_1'$*, such that* $q_1' \sim q_2'$.

From a bisimulation relation, we can construct an equivalent (but smaller) transition system $T'$ defined on the quotient set of states $Q/\sim$. Particularly, the reachable sets of $T$ and $T'$ are equal and therefore the safety verification problem of both systems are equivalent though much simpler to solve for $T'$.

For transition systems with an infinite number of states such as those generated by dynamical and hybrid systems, exhaustive simulation is generally not possible. Extensions of the notion of simulation and bisimulation relations have recently been developed [15, 16, 17]. Though simpler, the quotient system generally still has an infinite number of states for which exhaustive simulation would require to compute an infinite number of trajectories. In the following, we show that an approach based on the computation of a finite number of trajectories is possible using more robust relations defined by metrics.

**Definition 4 (Bisimulation metric).** *A continuous function* $d_{\mathcal{B}} : Q \times Q \to \mathbb{R}+$ *is a bisimulation metric if it is a pseudo-metric:*

1. *for all* $q \in Q$, $d_{\mathcal{B}}(q, q) = 0$,
2. *for all* $q_1, q_2 \in Q$, $d_{\mathcal{B}}(q_1, q_2) = d_{\mathcal{B}}(q_2, q_1)$,
3. *for all* $q_1, q_2, q_3 \in Q$, $d_{\mathcal{B}}(q_1, q_3) \leq d_{\mathcal{B}}(q_1, q_2) + d_{\mathcal{B}}(q_2, q_3)$,

*and if in addition, there exists* $\lambda > 1$*, such that for all* $q_1, q_2 \in Q$*,*

$$d_{\mathcal{B}}(q_1, q_2) \geq \max(\, d_{\Pi}(\langle\langle q_1 \rangle\rangle, \langle\langle q_2 \rangle\rangle),\ \lambda \sup_{q_1 \to q_1'} \inf_{q_2 \to q_2'} d_{\mathcal{B}}(q_1', q_2') \,). \tag{1}$$

The notion of bisimulation metrics extends the notion of bisimulation relations. A bisimulation metric measures how far two states are from being bisimilar:

**Proposition 1.** *The zero set of a bisimulation metric is a bisimulation relation.*

*Proof.* Let $q_1, q_2 \in Q$ such that $d_{\mathcal{B}}(q_1, q_2) = 0$. Then, from equation (1), we have that $d_{\Pi}(\langle\langle q_1 \rangle\rangle, \langle\langle q_2 \rangle\rangle) = 0$. Since $d_{\Pi}$ is a metric, this implies that $\langle\langle q_1 \rangle\rangle = \langle\langle q_2 \rangle\rangle$. Now, let $q_1 \to q_1'$, then since $d_{\mathcal{B}}$ is continuous and $\mathrm{Post}(q_2)$ is compact, equation (1) implies that there exists $q_2 \to q_2'$ such that $d_{\mathcal{B}}(q_1', q_2') = 0$. Similarly, since $d_{\mathcal{B}}(q_1, q_2) = d_{\mathcal{B}}(q_2, q_1)$, we have that for all $q_2 \to q_2'$, there exists $q_1 \to q_1'$ such that $d_{\mathcal{B}}(q_2', q_1') = d_{\mathcal{B}}(q_1', q_2') = 0$. ■

*Remark 2.* The branching distance defined in [10] and [11] as the smallest function (but not necessarily metric) $d$ satisfying the functional equation:

$$d(q_1,q_2) = \max(\ d_{\Pi}(\langle\langle q_1\rangle\rangle, \langle\langle q_2\rangle\rangle), \\ \lambda \sup_{q_1 \to q_1'} \inf_{q_2 \to q_2'} d(q_1', q_2'),\ \lambda \sup_{q_2 \to q_2'} \inf_{q_1 \to q_1'} d(q_1', q_2')\ )$$

is a bisimulation metric. Moreover, we have shown [11] that it is the smallest (or minimal) bisimulation metric which is the analog for metrics of the largest (or maximal) bisimulation relation for relations. Though it is possible to compute the minimal bisimulation metric for systems with a finite number of states [10], it becomes more problematic for systems with an infinite number of states. In that case, the relaxed conditions of Definition 4 allows to make computations easier [12, 13].

Given a bisimulation metric $d_{\mathcal{B}}$, we define state neighborhoods associated to this metric. For all $q \in Q$ and $\delta > 0$, $\mathcal{N}_{\mathcal{B}}(q,\delta) = \{q' \in Q \mid d_{\mathcal{B}}(q,q') \leq \delta\}$.

## 3 Simulation-Based Reachability Computation

In this section, we show that for metric transition systems equipped with a bisimulation metric, we can compute an approximation (with any desired precision) of the reachable set of a metric transition system by simulating a finite number of its state trajectories. Let us assume that we have a discretization function $Disc$ which associates to a compact set $\mathcal{C} \subseteq Q$ and a real number $\varepsilon > 0$, a finite set of points $Disc(\mathcal{C},\varepsilon) = \{q_1,\ldots,q_r\} \subseteq \mathcal{C}$ such that

$$\text{for all } q \in \mathcal{C},\ \text{there exists } q_i,\ \text{such that } d_{\mathcal{B}}(q,q_i) \leq \varepsilon.$$

Since $d_{\mathcal{B}}$ is assumed to be continuous, such a function always exists[1].

The reachable set of $T$ can then be approximated with arbitrarily precision using a finite set of simulations, *i.e.* by computing a finite number of state trajectories of $T$. Let $\delta > 0$ be the desired precision of approximation for the reachable set. Let $\varepsilon$ be a discretization parameter such that $\delta \geq \delta/\lambda + \varepsilon$. First, start with the discretization of the set of initial states of $T$, $P_0 = Disc(Q_0,\delta)$. Then, we compute some state trajectories of $T$ from the following iteration:

$$P_{i+1} = P_i \cup \left( \bigcup_{q \in P_i} Disc(\mathrm{Post}(q),\varepsilon) \right),\ i = 0,\ldots,N-1.$$

**Theorem 1.** *Let us consider the finite set*

$$\mathrm{Reach}_N^{\delta,\varepsilon}(T) = \{\pi \in \Pi \mid \exists q \in P_N,\ \langle\langle q\rangle\rangle = \pi\}.$$

*Then, the following inclusions hold*

$$\mathrm{Reach}_N^{\delta,\varepsilon}(T) \subseteq \mathrm{Reach}_N(T) \subseteq \mathcal{N}_{\Pi}\left(\mathrm{Reach}_N^{\delta,\varepsilon}(T),\delta\right).$$

[1] The proof is not stated here because of the lack of space.

*Proof.* The first inclusion is obvious since $P_N$ is obtained from simulations of $T$. Let $\pi \in \text{Reach}_N(T)$ and $q_0 \to \cdots \to q_k$ be a state trajectory of $T$, such that $\langle\langle q_k \rangle\rangle = \pi$ $(k \leq N)$. Since $q_0 \in Q_0$, there exists $p_0 \in Disc(Q_0, \delta) = P_0$ such that $d_{\mathcal{B}}(p_0, q_0) \leq \delta$. Then, from equation (1), there exists $p_1' \in \text{Post}(p_0)$ such that $d_{\mathcal{B}}(p_1', q_1) \leq \delta/\lambda$. There also exists $p_1 \in Disc(\text{Post}(p_0), \varepsilon) \subseteq P_1$ such that $d_{\mathcal{B}}(p_1, p_1') \leq \varepsilon$. From the triangular inequality, $d_{\mathcal{B}}(p_1, q_1) \leq \delta/\lambda + \varepsilon \leq \delta$. Recursively, we can show that there exists $p_k \in P_k \subseteq P_N$ such that $d_{\mathcal{B}}(p_k, q_k) \leq \delta$. Then, from equation (1), we also have $d_{\Pi}(\langle\langle p_k \rangle\rangle, \langle\langle q_k \rangle\rangle) \leq \delta$ which finally leads to the second inclusion. ■

If we assume that the number of elements $Disc(\text{Post}(q), \varepsilon)$ is always greater than an integer $r > 1$, then the set $P_N$ contains $\mathcal{O}(r^N)$ elements. Then, the number of trajectories that we need to compute grows exponentially with the time horizon $N$. To overcome this problem and design a more efficient reachability algorithm, one can think of using an approach similar to the systematic simulation algorithm proposed in [2]. The main idea consists in merging, at each iteration, *neighbor* states in $P_i$. The algorithm in [2] used general ellipsoidal neighborhoods and requires several ellipsoidal operations at each step to determine which states need to be merged. An implementation of this method with neighborhoods associated to the bisimulation metric seems promising and will be explored in the future.

## 4 Simulation-Based Safety Verification

The method presented in the previous section can be dramatically improved in the context of safety verification. First, it is seldom the case that we need a uniform approximation (in space) of the reachable set. Whereas the previous approach uniformly covers the reachable set with $\delta$ neighborhoods, an approach allowing rough approximations where it is possible (*i.e.* far from the unsafe set) and an accurate estimation where it is necessary (*i.e.* near the unsafe set) would definitely give more accurate results for equivalent computations. Second, if the approximation with $\delta$ neighborhoods does not allow concluding the safety of the transition system $T$, the previous approach does not give any guidance for refining our approximation other than choose a smaller $\delta$ and start over. Motivated by these two remarks, we propose an algorithm for safety verification for the class of metric transition systems generated by discrete-time linear systems of the form:

$$\Sigma : \begin{cases} x(k+1) = Ax(k) + Bu(k), & x(k) \in \mathbb{R}^n,\ u(k) \in U,\ x(0) \in I, \\ y(k) \quad\ = Cx(k), & y(k) \in \mathbb{R}^p, \end{cases} \tag{2}$$

where $U$ is a compact subset of $\mathbb{R}^m$ and $I$ is a compact subset of $\mathbb{R}^n$. The input $u(.)$ is to be thought as a disturbance rather than a control.

*Remark 3.* The distance between the reachable set of a continuous-time system and the reachable set of its sampled version can be quantified. The presented approach can therefore be adapted for safety verification of a continuous-time system at the expense of a quantifiable error.

### 4.1 Bisimulation Metrics for Linear Systems

In the spirit of [16], the linear system can be written as a nondeterministic transition system $T = (Q, \rightarrow, Q^0, \Pi, \langle\langle . \rangle\rangle)$ where

- the set of states is $Q = \mathbb{R}^n$,
- the transition relation is given by

$$x \rightarrow x' \iff \exists u \in U \text{ such that } x' = Ax + Bu,$$

- the set of initial states is $Q_0 = I$,
- the set of observations is $\Pi = \mathbb{R}^p$,
- the observation map is given by $\langle\langle x \rangle\rangle = Cx$.

The set of states and observations are equipped with the traditional Euclidean metric. Our approach requires a bisimulation metric for our transition system. Following [12], we search for bisimulation metrics of the form:

$$d_{\mathcal{B}}(x_1, x_2) = \sqrt{(x_1 - x_2)^T M (x_1 - x_2)} \tag{3}$$

where $M$ is a positive semi-definite symmetric matrix.

**Theorem 2.** *Let $M$ be a positive semi-definite symmetric matrix, $\lambda > 1$ such that the following linear matrix inequalities hold:*

$$M \geq C^T C, \tag{4}$$

$$M - \lambda^2 A^T M A \geq 0. \tag{5}$$

*Then, the function $d_{\mathcal{B}}(x_1, x_2)$ given by equation (3) is a bisimulation metric.*

*Proof.* It is clear that $d_{\mathcal{B}}$ is pseudo-metric. The linear matrix inequality (4) implies that

$$d_{\mathcal{B}}(x_1, x_2) \geq \sqrt{(x_1 - x_2)^T C^T C (x_1 - x_2)} = \|Cx_1 - Cx_2\|.$$

The linear matrix inequality (5) implies that for all $u \in U$,

$$\begin{aligned} \lambda d_{\mathcal{B}}(Ax_1 + Bu, Ax_2 + Bu) &= \lambda \sqrt{(x_1 - x_2)^T A^T M A (x_1 - x_2)} \\ &\leq \sqrt{(x_1 - x_2)^T M (x_1 - x_2)} = d_{\mathcal{B}}(x_1, x_2). \end{aligned}$$

It follows that $d_{\mathcal{B}}(x_1, x_2) \geq \lambda \sup_{x_1 \rightarrow x_1'} \inf_{x_2 \rightarrow x_2'} d_{\mathcal{B}}(x_1', x_2')$. ∎

Thus, a bisimulation metric can be computed by solving a set of linear matrix inequalities which can be done efficiently using semi-definite programming [18]. Moreover, for the class of asymptotically stable linear systems, bisimulation metrics of the form (3) are universal.

**Theorem 3.** *If $\Sigma$ is asymptotically stable (i.e. all the eigenvalues of $A$ lie inside the open unit disk), then there exists a bisimulation metric of the form (3).*

The proof is omitted here but a similar result has been proved in [12].

### 4.2 Safety Verification Algorithm

Let $T$ be a metric transition system generated by a stable discrete-time linear system and $d_{\mathcal{B}}$ a bisimulation metric of form (3). We propose a safety verification algorithm consisting of two main phases. First, by simulating a single trajectory of $T$, we compute a rough finite-state abstraction $T_{\mathcal{A}}$ of our transition system. Then, the algorithm automatically decides which new trajectories need to be simulated (choice of the initial value and of the sequence of inputs) in order to refine the abstraction $T_{\mathcal{A}}$ and conclude the safety of $T$.

The states of our abstraction are of the form $q = (x, \mu)$ with $x \in \mathbb{R}^n$ and $\mu \geq 0$ and should be thought of as representing the points of the neighborhood $\mathcal{N}_{\mathcal{B}}(x, \mu)$. The abstraction of $T$ is a transition system $T_{\mathcal{A}} = (Q_{\mathcal{A}}, \rightarrow_{\mathcal{A}}, Q_{\mathcal{A},0}, \Pi_{\mathcal{A}}, \langle\langle . \rangle\rangle_{\mathcal{A}})$ where the set of states $Q_{\mathcal{A}}$ is a finite subset of $\mathbb{R}^n \times \mathbb{R}^+$, the set of observations is $\Pi_{\mathcal{A}} = \Pi$ and the observation map is given by $\langle\langle (x, \mu) \rangle\rangle_{\mathcal{A}} = \langle\langle x \rangle\rangle$. We also need a set $Q_{\text{safe}} \subseteq Q_{\mathcal{A}}$ consisting of *safe* states of $T_{\mathcal{A}}$.

Algorithm 1 shows the structure of our safety verification algorithm. In the following, each step of the method is detailed.

```
Compute the initial abstraction T_A
while Reach_N(T_A) ∩ U = ∅ and Q_A ≠ Q_safe do
    - Main refinement loop:
    Determine the states to split S ⊆ Q_A \ Q_safe
    foreach q ∈ S do
        Split the state q          - refinement operation
    end
end
if Reach_N(T_A) ∩ U ≠ ∅ then
    return "The system is unsafe"
else
    return "The system is safe"
end
```

**Algorithm 1.** Safety verification algorithm

**Computation of the initial abstraction.** The initial abstraction is computed according to the following procedure. Initially, the set of states $Q_{\mathcal{A}}$, the set of initial states $Q_{\mathcal{A},0}$ the set of *safe* states $Q_{\text{safe}}$ as well as the transition relation $\rightarrow_{\mathcal{A}}$ are empty.

First, we choose an initial state $z_0 \in I$ and compute $\mu_0$ such that for all $x_0 \in I$, $d_{\mathcal{B}}(x_0, z_0) \leq \mu_0$. We insert $(z_0, \mu_0)$ in $Q_{\mathcal{A}}$ and $Q_{\mathcal{A},0}$. Then, we choose an input $v \in U$ and compute $\varepsilon$ such that for all $u \in U$, $d_{\mathcal{B}}(Bv, Bu) \leq \varepsilon$. For $i = 1 \ldots N$, we compute $z_i = Az_{i-1} + Bv$ and $\mu_i = \mu_{i-1}/\lambda + \varepsilon$. Note that this essentially consists in simulating system $T$ for the initial state $z_0$ and the constant input $v$. We insert $(z_i, \mu_i)$ in $Q_{\mathcal{A}}$ and $((z_{i-1}, \mu_{i-1}), (z_i, \mu_i))$ in the transition relation $\rightarrow_{\mathcal{A}}$.

The second step consists in inserting *safe* states in $Q_{\text{safe}}$. A state $(z_i, \mu_i)$ of the abstraction is *safe* if $\mathcal{N}_{\Pi}(\langle\langle z_i \rangle\rangle, \mu_i) \cap \mathcal{U} = \emptyset$ (*i.e.* it is safe now) and its succesors are *safe* (*i.e.* it is safe in the future). We start from the state

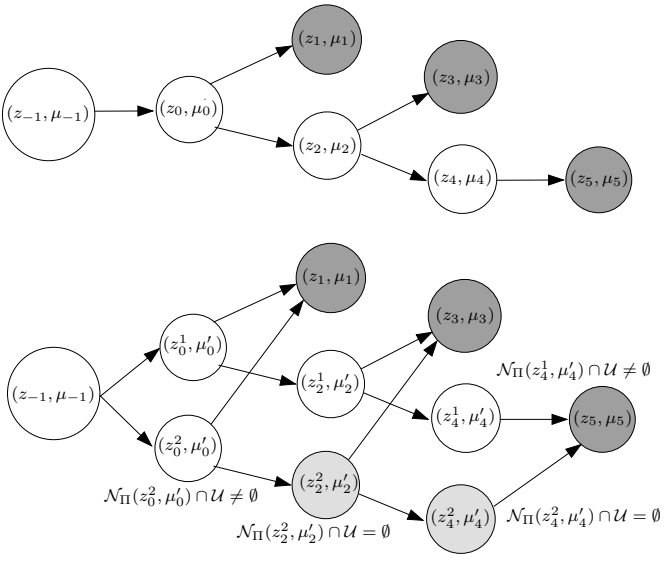


**Fig. 1.** Illustration of state splitting, the new states are obtained by simulation of the system. The grey states are *safe* states (elements of $Q_{\text{safe}}$): dark grey for states that were in $Q_{\text{safe}}$ before state splitting and light grey for those that were added during state splitting.

$(z_N,\mu_N)$, if $\mathcal{N}_{\Pi}(\langle\langle z_N\rangle\rangle,\mu_N)\cap\mathcal{U}=\emptyset$, then we insert $(z_N,\mu_N)$ in $Q_{\text{safe}}$. We repeat this procedure for $(z_{N-1},\mu_{N-1})$ and so on until we find $(z_{N'},\mu_{N'})$ such that $\mathcal{N}_{\Pi}(\langle\langle z_{N'}\rangle\rangle,\mu_{N'})\cap\mathcal{U}\neq\emptyset$.

**Refinement operation: state splitting.** If the initial abstraction is not sufficient to conclude safety ($Q_{\mathcal{A}}=Q_{\text{safe}}$) or unsafety ($\text{Reach}_N(T_{\mathcal{A}})\cap\mathcal{U}\neq\emptyset$) then the abstraction needs to be refined by splitting states. Let $\rho\in(0,1)$ be a refinement parameter that determines how many states result from state splitting; the smaller $\rho$, the more new states are inserted in the abstraction. For simplicity, we assume that all the states in $Q_{\mathcal{A}}\setminus Q_{\text{safe}}$ have at most one predecessor for the transition relation $\rightarrow_{\mathcal{A}}$[2]. We split any state $q_0=(z_0,\mu_0)\in Q_{\mathcal{A}}\setminus Q_{\text{safe}}$ according to the following procedure. State splitting is illustrated on Figure 1.

The first step consists in splitting the state $q_0$ into several states. If $q_0$ is an initial state (*i.e.* $q_0\in Q_{\mathcal{A},0}$), then let $\mu_0'=\rho\mu_0$ and $\{z_0^1,\ldots,z_0^r\}=Disc(\mathcal{N}_{\mathcal{B}}(z_0,\mu_0)\cap I,\rho\mu_0)$. We replace $(z_0,\mu_0)$ by $(z_0^1,\mu_0'),\ldots,(z_0^r,\mu_0')$ in $Q_{\mathcal{A}}$ and $Q_{\mathcal{A},0}$. If $q_0$ is not an initial state (*i.e.* $q_0\notin Q_{\mathcal{A},0}$), then let $(z_{-1},\mu_{-1})\in Q_{\mathcal{A}}$ be the predecessor of $q_0$ (*i.e.* $(z_{-1},\mu_{-1})\rightarrow_{\mathcal{A}}(z_0,\mu_0)$) and let $\{w_{-1}^1,\ldots,w_{-1}^r\}=Disc(\mathcal{N}_{\mathcal{B}}(Bv_{-1},\varepsilon_{-1})\cap BU,\rho\varepsilon_{-1})$ where $\varepsilon_{-1}=\mu_0-\mu_{-1}/\lambda$ and $v_{-1}\in U$ is the input which leads $T$ from $z_{-1}$ to $z_0$. Let $v_{-1}^1,\ldots,v_{-1}^r\in U$ be inputs such that $Bv_{-1}^j=w_{-1}^j$ $(j=1,\ldots,r)$. Let $z_0^j=Az_{-1}+Bv_{-1}^j$ and $\mu_0'=\mu_{-1}/\lambda+\rho\varepsilon_{-1}$, we replace

[2] This assumption is not restrictive since we are performing finite-horizon verification and thus such cases can be handled by duplicating states.

$(z_0, \mu_0)$ by $(z_0^1, \mu_0'), \ldots, (z_0^r, \mu_0')$ in $Q_{\mathcal{A}}$ and $((z_{-1}, \mu_{-1}), (z_0, \mu_0))$ is replaced by $((z_{-1}, \mu_{-1}), (z_0^1, \mu_0')), \ldots, ((z_{-1}, \mu_{-1}), (z_0^r, \mu_0'))$ in the transition relation $\rightarrow_{\mathcal{A}}$.

We update $Q_{\mathcal{A}}$ and $\rightarrow_{\mathcal{A}}$, so that each sequence of transitions of the form $(z_0, \mu_0) \rightarrow_{\mathcal{A}} (z_1, \mu_1) \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} (z_k, \mu_k)$ such that $(z_k, \mu_k) \notin Q_{\text{safe}}$ is replaced by $r$ sequences:

$$(z_0^j, \mu_0') \rightarrow_{\mathcal{A}} (z_1^j, \mu_1') \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} (z_k^j, \mu_k'),\ j = 1, \ldots, r$$

such that $z_{i+1}^j = Az_i^j + Bv_i$ where $v_i \in BU$ is the input which leads the system $T$ from $z_i$ to $z_{i+1}$ and $\mu_{i+1}' = \mu_i'/\lambda + \varepsilon_i$ where $\varepsilon_i = \mu_{i+1} - \mu_i/\lambda$. Hence, for each trajectory initiating from $z_0$ (associated to a sequence of inputs $v_0, \ldots, v_{k-1}$), we need to simulate the trajectories starting in $z_0^1, \ldots, z_0^r$ for the same sequence of inputs. For each *safe* successors of $(z_k, \mu_k)$, $(z_{k+1}, \mu_{k+1}) \in Q_{\text{safe}}$, the transition $((z_k, \mu_k), (z_{k+1}, \mu_{k+1}))$ is replaced by the transitions $((z_k^1, \mu_k'), (z_{k+1}, \mu_{k+1})), \ldots, ((z_k^r, \mu_k'), (z_{k+1}, \mu_{k+1}))$ in $T_{\mathcal{A}}$. The main idea is that since we already know that $(z_{k+1}, \mu_{k+1})$ is safe, there is no need to split this state.

Finally, we update the set of *safe* states $Q_{\text{safe}}$. For each new state $(z, \mu)$ of $T_{\mathcal{A}}$, if all its successors are safe and $\mathcal{N}_{\Pi}(\langle\langle z \rangle\rangle, \mu) \cap \mathcal{U} \neq \emptyset$, then we insert $(z, \mu)$ in $Q_{\text{safe}}$. The same process is repeated for the predecessor of $(z, \mu)$.

*Remark 4.* The number of new states introduced by the splitting of $q_0$ depends critically on how many transitions separate $q_0$ from a state in $Q_{\text{safe}}$. For instance if all the successors of $q_0$ are in $Q_{\text{safe}}$ then the refinement operation adds $r$ new states. On the other hand, if $q_0$ is an initial state the total number of states of the abstraction can multiplied by $r$.

At each iteration of the main loop of Algorithm 1, we choose a set $\mathcal{S} \subseteq Q_{\mathcal{A}} \setminus Q_{\text{safe}}$ of states to be split according to a refinement policy. Then, we apply state splitting to each state in $\mathcal{S}$. The order in which we split the elements of $\mathcal{S}$ is in backward manner, that is we split an element $q$ of $\mathcal{S}$ if all the states $q' \in \mathcal{S}$ such that $q \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} q'$ have already been split.

The refinement procedure defined by state splitting is such that after each refinement, the states $q$ which remain in $Q_{\mathcal{A}} \setminus Q_{\text{safe}}$ are those for which there exists a sequence of transitions $q \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} q'$ such that $q' = (z', \mu')$ and $\mathcal{N}_{\Pi}(\langle\langle z' \rangle\rangle, \mu) \cap \mathcal{U}$ is not empty. This means that all states of $Q_{\mathcal{A}} \setminus Q_{\text{safe}}$ are states which potentially lead to an unsafe state. Since only these states are refined, this approach is similar to counterexample guided abstraction refinement [19, 4, 5].

**Soundness and completeness.** Before stating results on soundness and completeness of Algorithm 1, we need two approximation results.

**Lemma 1.** *Let $T_{\mathcal{A}}$ be an abstraction of $T$ obtained from the initial abstraction by a finite sequence of state splittings, let $x_0 \rightarrow \cdots \rightarrow x_k \in S_N(T)$. Then, there exists $(z_0, \mu_0) \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} (z_k, \mu_k) \in S_N(T_A)$ such that one of the following holds*

1. $(z_k, \mu_k) \in Q_{safe}$ *and* $d_{\mathcal{B}}(x_k, z_k) \leq \mu_k$,
2. $(z_k, \mu_k) \notin Q_{safe}$, $d_{\mathcal{B}}(x_0, z_0) \leq \mu_0$ *and* $d_{\mathcal{B}}(Bu_i, Bv_i) \leq \varepsilon_i$ $(i = 0, \ldots, k-1)$, *where* $\varepsilon_i = \mu_{i+1} - \mu_i/\lambda$ *and* $u_0, \ldots, u_{k-1}$ *(respectively* $v_0, \ldots, v_{k-1}$*) is the sequence of inputs associated to the trajectory* $x_0 \to \cdots \to x_k$ *(respectively* $z_0 \to \cdots \to z_k$*).*

*Proof.* Let $x_0 \to \cdots \to x_k \in S_N(T)$, let $(z_0, \mu_0) \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} (z_k, \mu_k)$ be the unique trajectory of length $k$ of the initial abstraction $T_{\mathcal{A}}$. By construction, we have $d_{\mathcal{B}}(x_0, z_0) \leq \mu_0$ and $d_{\mathcal{B}}(Bu_i, Bv_i) \leq \varepsilon_i$ $(i = 0, \ldots, k-1)$. Then,

$$\begin{aligned} d_{\mathcal{B}}(x_{i+1}, z_{i+1}) &\leq d_{\mathcal{B}}(Ax_i + Bu_i, Ax_i + Bv_i) + d_{\mathcal{B}}(Ax_i + Bv_i, Az_i + Bv_i) \\ &\leq \varepsilon_i + d_{\mathcal{B}}(x_i, z_i)/\lambda. \end{aligned}$$

By induction, we have that $d_{\mathcal{B}}(x_k, z_k) \leq \mu_k$. Hence, it is clear that the property holds for the initial abstraction. Let us assume that it holds after a finite sequence of refinements, and let $(z_0, \mu_0) \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} (z_k, \mu_k)$ be the associated element of $S_N(T_{\mathcal{A}})$. We apply state splitting to an element $q \in Q_{\mathcal{A}} \setminus Q_{\text{safe}}$. If $(z_k, \mu_k)$ was in $Q_{\text{safe}}$ before state splitting, then it is clear that the first assertion of the lemma still holds after state splitting. Let us assume that $(z_k, \mu_k)$ is not in $Q_{\text{safe}}$ and that the second assertion of the lemma holds before state splitting. Particularly, it can be shown by induction that $d_{\mathcal{B}}(x_k, z_k) \leq \mu_k$. If for all $i \in \{0, \ldots, k\}$, $q \neq (z_i, \mu_i)$ then $(z_0, \mu_0) \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} (z_k, \mu_k)$ is still a trajectory of $T_{\mathcal{A}}$ after state splitting and one of the two assertions of the lemma holds. If $q = (z_i, \mu_i)$ (for some $i \in \{1, \ldots, k\}$, the case $i = 0$ being similar), then after state splitting, we know by construction that there exists a trajectory of $T_{\mathcal{A}}$ of the form $(z_0, \mu_0) \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} (z_{i-1}, \mu_{i-1}) \to_{\mathcal{A}} (z_i^j, \mu_i') \cdots \to_{\mathcal{A}} (z_k^j, \mu_k')$ such that $d_{\mathcal{B}}(x_0, z_0) \leq \mu_0$ and $d_{\mathcal{B}}(Bu_0, Bv_0) \leq \varepsilon_0, \ldots, d_{\mathcal{B}}(Bu_{i-2}, Bv_{i-2}) \leq \varepsilon_{i-2}$, $d_{\mathcal{B}}(Bu_{i-1}, Bv_{i-1}^j) \leq \varepsilon_{i-1}' = \rho\varepsilon_{i-1}$, $d_{\mathcal{B}}(Bu_i, Bv_i) \leq \varepsilon_i, \ldots, d_{\mathcal{B}}(Bu_{k-1}, Bv_{k-1}) \leq \varepsilon_{k-1}$. Note that this also implies that $d_{\mathcal{B}}(x_k, z_k^j) \leq \mu_k'$. Therefore, one of the assertions of the lemma holds after state splitting. Hence, Lemma 1 is proved by induction. ■

**Theorem 4.** *Let* $T_{\mathcal{A}}$ *be an abstraction of* $T$ *obtained from the initial abstraction by a finite sequence of state splittings. Let us define the following set*

$$\widetilde{\text{Reach}}_N(T_{\mathcal{A}}) = \{\pi \in \Pi \mid \exists (z, \mu) \in Q_{\mathcal{A}},\ d_\pi(\langle\langle z \rangle\rangle, \pi) \leq \mu\}.$$

*Then, the following inclusions hold*

$$\text{Reach}_N(T_{\mathcal{A}}) \subseteq \text{Reach}_N(T) \subseteq \widetilde{\text{Reach}}_N(T_{\mathcal{A}}).$$

*Proof.* The first inclusion is obvious because the states of the abstraction are computed by simulation of $T$. Let $\pi \in \text{Reach}_N(T)$ and $x_0 \to \cdots \to x_k$ be a state trajectory of $T$, such that $\langle\langle x_k \rangle\rangle = \pi$ $(k \leq N)$. From Lemma 1, there exists $(z_0, \mu_0) \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} (z_k, \mu_k) \in S_N(T_A)$ such that one assertion of the lemma holds. Let us remark that in both cases, we have $d_{\mathcal{B}}(z_k, x_k) \leq \mu_k$ and therefore $d_\pi(\langle\langle z_k \rangle\rangle, \pi) \leq \mu_k$. ■

The following soundness result is straightforward:

**Theorem 5.** *If Algorithm 1 terminates, then it provides the correct answer to the safety verification problem.*

*Proof.* If at the termination of Algorithm 1, we have $\mathrm{Reach}_N(T_{\mathcal{A}}) \cap \mathcal{U} \neq \emptyset$, then from the first inclusion of Theorem 4, we have that $T$ is unsafe. If at the termination of Algorithm 1, we have $Q_{\mathcal{A}} = Q_{\mathrm{safe}}$, this particularly means that for all $(z, \mu) \in Q_{\mathcal{A}}$, $\mathcal{N}_{\pi}(\langle\langle z \rangle\rangle, \mu) \cap \mathcal{U} = \emptyset$. From the second inclusion of Theorem 4, we have that $T$ is safe. ■

Guaranteed termination of Algorithm 1 requires defining more precisely the refinement policy. If at each iteration of the main loop of Algorithm 1, we split all the states of $T_{\mathcal{A}}$ ($\mathcal{S} = Q_{\mathcal{A}} \setminus Q_{\mathrm{safe}}$), then we have the following completeness result:

**Theorem 6.** *If we apply the refinement policy $\mathcal{S} = Q_{\mathcal{A}} \setminus Q_{safe}$, and if the metric transition system $T$ is either robustly safe or robustly unsafe with coefficient of robustness $\delta$, then Algorithm 1 terminates after at most $\lceil (\log(\delta) - \log(\bar{\mu}_0))/\log(\rho) \rceil$ iterations where $\bar{\mu}_0 = \max\{\mu |\ (z, \mu) \in Q_{\mathcal{A}} \setminus Q_{safe}$ in the initial abstraction$\}$.*

*Proof.* Let $\bar{\mu}_i = \max\{\mu |\ (z, \mu) \in Q_{\mathcal{A}} \setminus Q_{\mathrm{safe}}$ after the i-th refinement loop$\}$. Since at each refinement loop, state splitting is applied to all the states in $Q_{\mathcal{A}} \setminus Q_{\mathrm{safe}}$, it is not hard to see that $\bar{\mu}_{i+1} \leq \rho \bar{\mu}_i$. Then, $\bar{\mu}_i \leq \rho^i \bar{\mu}_0$. It follows that for $i \geq (\log(\delta) - \log(\bar{\mu}_0))/\log(\rho)$, $\bar{\mu}_i \leq \delta$. Let us assume that Algorithm 1 did not terminate after $i$ iterations. Then, there exists $(z, \mu) \in Q_{\mathcal{A}} \setminus Q_{\mathrm{safe}}$ with $\mu \leq \delta$ and such that $\langle\langle z \rangle\rangle \notin \mathcal{U}$ and $\mathcal{N}_{\Pi}(\langle\langle z \rangle\rangle, \mu) \cap \mathcal{U}$ is not empty. From Theorem 4, we have that $\langle\langle z \rangle\rangle \in \mathrm{Reach}_N(T)$, it follows that $T$ cannot be robustly safe. If $T$ was robustly unsafe, there would be a $\pi \in \mathrm{Reach}_N(T)$ such that $\mathcal{N}_{\Pi}(\pi, \delta) \subseteq \mathcal{U}$, from Theorem 4, there exists $\pi' \in \mathrm{Reach}_N(T_A)$ such that $d_{\Pi}(\pi, \pi') \leq \bar{\mu}_i$. Hence, $\pi' \in \mathcal{N}_{\Pi}(\pi, \bar{\mu}_i) \subseteq \mathcal{N}_{\Pi}(\pi, \delta) \subseteq \mathcal{U}$ which contradicts the fact that Algorithm 1 did not terminate. ■

We can see that the more robust with respect to the safety property a system is, the less refinements are needed resulting in fewer computations and easier safety verification. Note that particularly, if $\bar{\mu}_0 \leq \delta$, no refinement is needed to decide wether $T$ is safe or unsafe. This is an important advantage of the method.

In practice, it is seldom necessary to apply state splitting to all the states of the abstraction. Moreover, we have seen that applying state splitting to states that are separated by a large number of transitions of a state in $Q_{\mathrm{safe}}$ may result in a large increase of the number of states in $Q_{\mathcal{A}}$. Hence, from this point of view it is better to apply state splitting to states that are within a small number of transitions from elements in $Q_{\mathrm{safe}}$. A different refinement policy can be defined by $\mathcal{S} = \mathcal{P}(Q_{\mathcal{A}}, p)$ which consists of the states $q = (z, \mu) \in Q_{\mathcal{A}}$ such that there exists a sequence of transition of the form $q \rightarrow_{\mathcal{A}} q_1 \rightarrow_{\mathcal{A}} \cdots \rightarrow_{\mathcal{A}} q_k$ with $q_k \in Q_{\mathrm{safe}}$ and $k \leq p$. Note that for this refinement policy, Theorem 6 does not hold even if Algorithm 1 shows better performances in practice. For theoretical completeness, we can use a refinement policy which alternates $\mathcal{S} = \mathcal{P}(Q_{\mathcal{A}}, p)$ and $\mathcal{S} = Q_{\mathcal{A}} \setminus Q_{\mathrm{safe}}$. In that case, a result similar to Theorem 6 holds.

*Remark 5.* It is clear from Lemma 1 that the abstraction $T_{\mathcal{A}}$ not only allows to approximate the reachable set of $T$ but also its language. This is strong evidence that our approach can be generalized for the simulation-based verification of more complex properties such as those expressible in linear temporal logic [14].

### 4.3 Experimental Results

Let us consider the following continuous-time linear system:

$$\begin{cases} \dot{x}_1(t) = \quad 3x_1(t) + 20x_2(t) \\ \dot{x}_2(t) = -2x_1(t) - 9x_2(t) + x_3(t) + u(t) \\ \dot{x}_3(t) = -4x_3(t) + 2u(t) \end{cases}$$

For piecewise constant inputs with sampling period $\tau = 0.1$, the sampled system dynamics are given by

$$x(k+1) = Ax(k) + Bv(k), \text{ where}$$

$$A = \begin{bmatrix} 1.17 & 1.47 & 0.07 \\ -0.15 & 0.28 & 0.05 \\ 0 & 0 & 0.67 \end{bmatrix}, \; B = \begin{bmatrix} 0.09 \\ 0.07 \\ 0.16 \end{bmatrix}, \; x(k) = \begin{bmatrix} x_1(k\tau) \\ x_2(k\tau) \\ x_3(k\tau) \end{bmatrix},$$

and $v(k) = u(k\tau)$. Only the variable $x_2$ is observed (*i.e.* $C = [0\ 1\ 0]$). The set of initial states $I$ and of inputs $U$ are given by $I = [-0.05, 0.05] \times [9.95, 10.05] \times \{0\}$ and $U = [0, 2.5]$. $T$ denotes the associated metric transition system. The safety

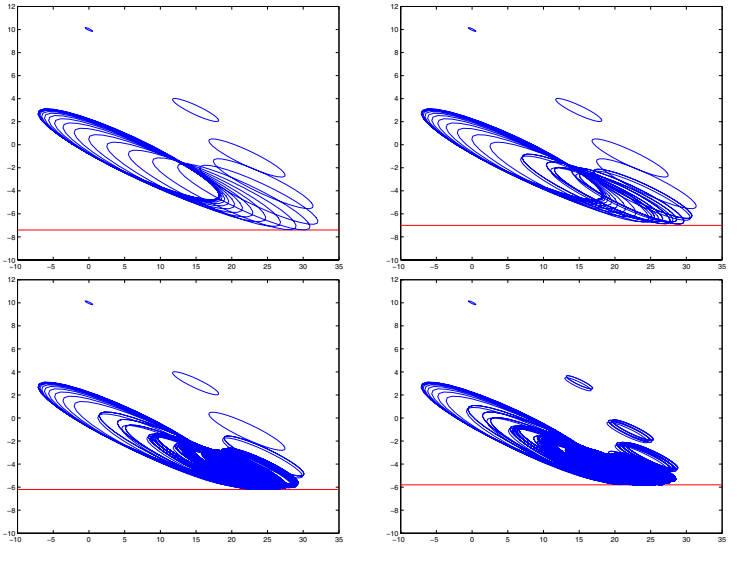

**Fig. 2.** Over approximation of the set reachable by $x_1$ and $x_2$, the red line represent the border of the unsafe set. The quality of the approximation is adapted automatically to the safety property we want to verify.

property we want to check is wether the reachable set $\text{Reach}_{20}(T)$ intersects the set $\{y \leq \theta\}$ or not. In order to apply our safety verification algorithm, we need to compute a bisimulation metric. This is done by solving linear matrix inequalities (4) and (5). Our safety verification algorithm has been implemented in MATLAB and used for several values of the parameter $\theta$ with the refinement parameter and the refinement policy $\mathcal{S} = \mathcal{P}(Q_{\mathcal{A}}, 5)$.

Figure 2 represents the over approximation of the set reachable by $x_1$ and $x_2$ computed by our algorithm for different values of $\theta$. We can see that as the value of $\theta$ becomes larger, the system becomes less robustly safe and our over-approximation of the reachable set needs to be more precise. Let us remark that the state-splitting is effectively applied where it is needed, that is where the reachable set is close to the unsafe set. The results of our computations are presented in Figure 3. Experimentation confirms what we expected from Theorem 6. Indeed, we can check that if the system is very robust with respect to the safety property, the safety verification is performed using only one simulation and takes less than a second. As the system safety becomes less robust, the algorithm needs more time to decide if the system is safe or unsafe. On Figure 3, this is visible and we can expect that the curves of the CPU time and of the number of refinement loops have a vertical asymptote for some critical value of $\theta$.

| $\theta$ | Result | CPU time (s) | Refinements |
|---|---|---|---|
| -7.4 | Safe | 0.16 | 0 |
| -7.0 | Safe | 0.25 | 1 |
| -6.5 | Safe | 0.44 | 2 |
| -5.8 | Safe | 74.77 | 3 |
| -4.6 | Unsafe | 5.82 | 3 |
| -4.5 | Unsafe | 0.16 | 0 |

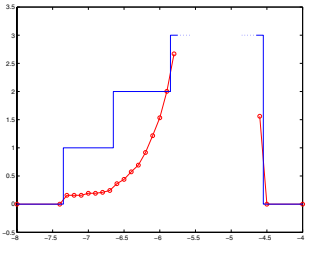

**Fig. 3.** Results of Algorithm 1 (table). Number of refinement loops needed by Algorithm 1 and CPU time in logarithmic scale against the parameter $\theta$ (figure).

## 5 Conclusion

In this paper we presented a simulation-based framework for verifying the safety of metric transition systems. Our algorithm critically relies on recently developed bisimulation metrics which can be used to approximate arbitrarily close the reachable set of metric transition systems by simulating a finite number of trajectories. For metric transition systems generated by nondeterministic linear systems, we proposed a safety verification algorithm which is complete for systems that are robustly safe or robustly unsafe. Future research will focus on lifting the completeness of the safety verification algorithm to general metric transition systems, including classes of hybrid systems.

## References


1. Alur, R., Grosu, R., Hur, Y., Kumar, V., Lee, I.: Modular specification of hybrid systems in charon. In: HSCC '00: Proceedings of the 3rd International Workshop on Hybrid Systems. Volume 1790 of Lecture Notes In Computer Science., Springer-Verlag (2000) 6 – 19
2. Kapinski, J., Krogh, B.H., Maler, O., Stursberg, O.: On systematic simulation of open continuous systems. In: Hybrid Systems: Computation and Control. Volume 2623 of LNCS., Springer (2003) 283–297
3. Lee, E.A., Zheng, H.: Operational semantics of hybrid systems. In: HSCC '05: Proceedings of the 8th International Workshop on Hybrid Systems. Volume 3414 of Lecture Notes In Computer Science., Springer-Verlag (2005) 25 – 53
4. Alur, R., Dang, T., Ivancic, F.: Counter-example guided predicate abstraction of hybrid systems. In: Tools and Algorithms for the Construction and Analysis of Systems. Volume 2619 of LNCS., Springer (2003) 208–223
5. Clarke, E., Fehnker, A., Han, Z., Krogh, B., Ouaknine, J., Stursberg, O., Theobald, M.: Abstraction and counterexample-guided refinement in model checking of hybrid systems. International Journal of Foundations of Computer Science **14**(4) (2003)
6. Prajna, S., Jadbabaie, A.: Safety verification of hybrid systems using barrier certificates. In: Hybrid Systems: Computation and Control. Volume 2993 of Lecture Notes in Computer Science., Springer (2004) 477 – 492
7. Mitchell, I., Tomlin, C.: Level set methods for computation in hybrid systems. In: Hybrid Systems: Computation and Control. Volume 1790 of LNCS., Springer (2000)
8. Frehse, G.: Phaver: Algorithmic verification of hybrid systems past hytech. In: HSCC '05: Proceedings of the 8th International Workshop on Hybrid Systems. Volume 3414 of Lecture Notes In Computer Science. (2005) 258–273
9. Girard, A.: Reachability of uncertain linear systems using zonotopes. In: Hybrid Systems: Computation and Control. Volume 3414 of Lecture Notes in Computer Science., Springer (2005) 291–305
10. de Alfaro, L., Faella, M., Stoelinga, M.: Linear and branching metrics for quantitative transition systems. In: ICALP'04. Volume 3142 of LNCS., Springer (2004) 1150–1162
11. Girard, A., Pappas, G.J.: Approximation metrics for discrete and continuous systems. Technical Report MS-CIS-05-10, Dept. of CIS, University of Pennsylvania (2005)
12. Girard, A., Pappas, G.J.: Approximate bisimulations for constrained linear systems. In: Proc. IEEE Conference on Decision and Control and European Control Conference, Seville, Spain (2005) 4700–4705
13. Girard, A., Pappas, G.J.: Approximate bisimulations for nonlinear dynamical systems. In: Proc. IEEE Conference on Decision and Control and European Control Conference, Seville, Spain (2005) 684–689
14. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press (2000)
15. Haghverdi, E., Tabuada, P., Pappas, G.J.: Bisimulation relations for dynamical, control, and hybrid systems. Theoretical Computer Science **342**(2-3) (2005) 229–262
16. Pappas, G.J.: Bisimilar linear systems. Automatica **39**(12) (2003) 2035–2047
17. van der Schaft, A.: Equivalence of dynamical systems by bisimulation. IEEE Transactions on Automatic Control **49**(12) (2004) 2160–2172
18. Sturm, J.F.: Using SEDUMI 1.02, a MATLAB toolbox for optimization over symmetric cones. Optimization Methods and Softwares **11-12** (1999) 625–653
19. Clarke, E., Grumberg, O., Jha, S., Lu, Y., Veith, H.: Counterexample-guided abstraction refinement. In: Computer Aided Verification. Volume 1855 of LNCS., Springer (2000) 154–169