From Simulation Models to Hybrid Automata Using Urgency and Relaxation *

Stefano Minopoli
Univ. Grenoble Alpes, VERIMAG
Centre Équation - 2, avenue de Vignate
38610 GIÉRES
stefano.minopoli@imag.fr

Goran Frehse
Univ. Grenoble Alpes, VERIMAG
Centre Équation - 2, avenue de Vignate
38610 GIÉRES
goran.frehse@imag.fr

ABSTRACT

We consider the problem of translating a deterministic *simulation model* (like Matlab-Simunk, Modelica or Ptolemy models) into a *verification model* expressed by a network of hybrid automata. The goal is to verify safety using reachability analysis on the verification model. Simulation models typically use transitions with urgent semantics, which must be taken as soon as possible. Urgent transitions also make it possible to decompose systems that would otherwise need to be modeled with a monolithic hybrid automaton. In this paper, we include urgent transitions in our verification models and propose a suitable adaptation of our reachability algorithm. However, the simulation model, due to its imperfections, may be unsafe even though the corresponding hybrid automata are safe. Conversely, set-based reachability may not be able to show safety of an ideal formal model, since complex dynamics necessarily entail overapproximations. Taken as a whole, the formal modeling and verification process can both falsely claim safety and fail to show safety of the concrete system.

We address this inconsistency by relaxing the model as follows. The standard semantics of hybrid automata is a mathematical idealization, where reactions are considered to be instantaneous and physical measurements infinitely precise. We propose semantics that relax these assumptions, where guard conditions are sampled in discrete time and admit measurement errors. The relaxed semantics can be translated to an equivalent relaxed model in standard semantics. The relaxed model is realistic in the sense that it can be implemented on hardware fast and precise enough, and in a way that safety is preserved. Finally, we show that overapproximative reachability analysis can show safety of relaxed models, which is not the case in general.

*The authors gratefully acknowledge financial support by the European Commission project UnCoVerCPS under grant number 643921.

HSCC'16, April 12 - 14, 2016, Vienna, Austria

ISBN 978-1-4503-3955-1/16/04...$15.00

DOI: http://dx.doi.org/10.1145/2883817.2883825

Keywords

Hybrid Systems, Hybrid Automata, Reachability Analysis, Numerical Analysis, Urgency

1. INTRODUCTION

A hybrid system describes the interaction of both discrete and continuous components over time. This combination can quickly lead to complex behaviors that are difficult to predict and control. The model based design of hybrid systems is commonly based on *simulation models* for tools like Matlab/Simulink [16], Modelica [17], Ptolemy [5], and many more. It is very hard to exhaustively test models using numerical simulation, so critical behaviors may go undetected. Set-based verification methods, on the other hand, can cover all possible behaviors in a single analysis. But they require a *verification model*, typically in the form of a *Hybrid Automaton* (HA) [12]. Hybrid Automata extend traditional state machines with continuous variables, governed by differential equations for modeling the continuous evolution of physical activities. We assume that verification models are used to verify safety or bounded liveness properties using approximative, set-based reachability algorithms, which are applicable to systems with piecewise-affine dynamics and hundreds of continuous variables [8, 7]. In this paper, we propose a relaxation of hybrid automata such that they are conservative abstractions of simulation models, and show how standard reachability techniques can be applied without necessarily compromising the computational cost of the analysis.

In the standard semantics of hybrid automata, transitions are nondeterministic: The system may take a transition when it satisfies the guard condition of a transition or remain in the same discrete state as long as the invariant (staying condition) holds. Simulation models are typically deterministic, since the simulator needs to be able to compute what happens in the next step. Simulation models therefore use urgent semantics, where a transition is taken as soon as the guard condition is satisfied. Urgent transitions make it possible to decompose systems that would otherwise need to be modeled with a monolithic hybrid automaton. This greatly facilitates the translation from deterministic models to hybrid automata, since it can be done component-wise. This is important in practice, since the structure of the model has a profound impact on several aspects of safety-critical model development, see [25]. Urgent semantics are not covered by standard reachability algorithms, since the computation of the states reachable by time elapse is more complex. In addition, instantaneous reactions can not be guaranteed by

combining invariants and transitions guards. For example when the variables in the guard condition have a derivative equal to zero, the system may remain forever on the border between invariant and guard and this is in contrast with the instantaneous reaction requirement. We fill this gap by generalizing the urgent time elapse operator in [19] from piecewise constant dynamics to piecewise affine dynamics. We show that the time elapse computation with urgency can be reduced to time elapse with a nonconvex invariant. Since the standard reachability algorithms work on convex invariants, we propose a a time elapse operator for nonconvex invariants. The idea is to consider a coverage of the invariant, where the elements are convex and closed. Then the standard time elapse is applied recursively on each of the elements.

The introduction of urgency in hybrid automata allows us a relatively straightforward translation from simulation models, e.g. the translation tool *SL2SX* [21, 18]. But the obtained hybrid automata are a mathematical idealization, where reactions (the effects of discrete events) are considered to be instantaneous and the variables have infinite precision. Clearly this is not the case for simulation, which computes an approximation of the state while taking discrete time steps. When the safety of a verification model relies on the assumption of instantaneous reaction, it can not be always implemented by a simulation model. In other words, the safety of the ideal hybrid automaton does not prove the safety of the corresponding simulation model. This should not be simply dismissed as a defect of the simulation model. In industrial practice, simulation models are developed over years and finely tuned up to the point that the simulator output is considered a faithful reflection of reality [9]. Furthermore, if the system can not be simulated numerically, it stands to reason that it can also not be implemented on a digital controller.

We illustrate the point with a variation of a *DC-to-DC switched-mode power converter* from [23], implemented in Simulink/Stateflow. The state variables are currents and voltages whose continuous dynamics are specified by switched linear ordinary differential equations. We add an urgent transition to change the dynamics when the voltage reaches the value $x = 15$ exactly. In the Simulink output shown in Fig. 1(a), the simulator does not detect any of the three crossings of $x = 15$. This could be considered a bug in the model, since we could have used one of Simulink's special blocks for detecting zero crossings. It could also point to a bug in the controller design, since an implementation of the system with a digital controller may miss the crossing just like the simulator. Either way, one should hope to detect this behavior using formal verification.

We convert the simulink model to a hybrid automaton using the translator *SL2SX* [21, 18], and run the verification tool SpaceEx [8]. In the hybrid automaton model, the urgent transition is always and instantaneously fired when the guard $x = 15$ is satisfied, as shown in Fig. 1(b). As a consequence, the reachability set is different and can not be used to prove safety of the simulation model. In the following we propose a relaxed hybrid automaton, which is an abstraction of the simulation model and therefore contains both behaviors, as shown in Fig. 1(c). Usually this is achieved by conservative abstraction, in the sense that it covers all behaviors. But in practice this is often overly conservative, such that the analysis either returns false negatives or becomes infeasible due to state explosion. The mathematical reason for this is that in numerical simulation the approximation errors may cancel each other out, while in set-based reachability they accumulate. Our approach can deal with the simulation model without adding overly conservative error terms. This makes it applicable to more complex systems and properties. Since this approach does not add any additional variables, locations or transitions to the model, it therefore does not fundamentally change scalability compared to standard HA models.

In this paper, we propose to relax the semantics of the hybrid automaton to capture that the simulated system

(i) may check guard conditions at discrete time points, assuming a bound $\Delta \geq 0$ on the time step, and

(ii) may check guard conditions with a precision error ϵ.

Point (i) is the relaxation of the instantaneous reaction assumption and point (ii) is the relaxation of the infinite precision assumption. For $\Delta = 0$ and $\varepsilon = 0$, we obtain the standard semantics. We show that the behaviors generated by relaxed semantics are equivalent to a relaxed model with standard semantics. The relaxation consists of enlarging the guards and shrinking urgent guard conditions accordingly. Our reachability algorithm can then be applied on the relaxed model.

Notice that our assumptions (see also Sec. 3.2) cover all simulation runs with a time step of Δ or smaller. Even zero crossing detection is covered by these assumptions since it amounts to reducing the time step for particular sections of a trajectory. If one has enough confidence in the zero crossing detection algorithm to be sure that no zeros are missed above some time step Δ^* (e.g., the minimal step size of the zero detection algorithm), then one can substitute Δ^* for Δ.

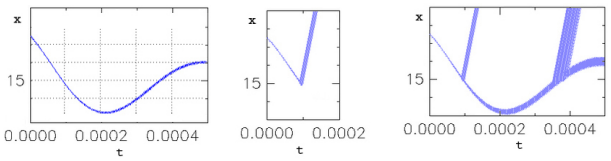
The relaxed semantics with urgency have additional interesting properties, some of them similar to *almost asap semantics* from [26]. First, *fast is better*, meaning that if the relaxed model with sampling time Δ and precision error ϵ is safe, then it is also safe for any smaller sampling time Δ' using the same precision error ϵ. Second, *precision is quality*, meaning that if the relaxed model with sampling time Δ and error ϵ is safe, then it is also safe for Δ and any smaller precision error ϵ'. Third, relaxed models are realistic in the sense that they can be implemented on hardware *fast and precise enough*, and in a way that safety is preserved. Indeed, our approach compensates for time discretization and approximation errors incurred in numerical simulation. Controllers implemented in digital hardware suffer from exactly the same flaws, if we consider measurement errors part of the approximation error. Safety of the simulation model therefore implies safety of a hardware system with high enough overall accuracy (including computation and measurement errors).

Moreover, we show that overapproximative reachability analysis can show safety of relaxed models, which is not the case in general.

For lack of space, some proofs were omitted from this paper; they can be found in [20].

Related Work.

Time-elapse computations for nonconvex invariants have already been tackled for hybrid automata with piecewise



(a) Simulink (b) ideal HA (c) relaxed HA

Figure 1: The simulation model may differ from an idealized hybrid automata model; a relaxed hybrid automaton can capture all cases

constant dynamics (LHA). The technique proposed by [14, 4] consist of modeling the nonconvex invariant by splitting locations, each one with a single convex component of the invariant. The technique in [19] is similar, but skips the construction of auxiliary locations by iterating over a convex cover of the invariant. It also resolves technicalities arising from strict inequalities. We follows this last approach, extending the results to piecewise affine dynamics.

A detailed and formal discussion of urgency can be found in [10] and the references therein. A general class of hybrid automata with urgency conditions is described in [24]. Urgency conditions are also part of the *Computational Interchange Format for Hybrid Systems (CIF)* [3], and a restricted class of urgent guard conditions can be handled by the classic reachability tool HyTech [13]. Computing the reachable states for the general case of urgency is not trivial. An algorithm for an effective computation of the time-elapse for HA with urgency was proposed for linear hybrid automata (LHA) in [19]. This time elapse computation with urgency is reduced to time-elapse with nonconvex invariants. We follows a similar approach, extending the algorithm to piecewise affine dynamics.

Our timed relaxation is closely related to [26], which introduces a parametric semantics for timed controllers called *Almost ASAP semantics (AASAP)*. These semantics relax the assumption of instantaneous reaction by imposing the controller to react within Δ time units when an urgent action takes place. Similar to our case, the AASAP semantics is such that faster is better and such that any controller proven to be correct for some $\Delta > 0$ can be implemented. Our semantics can be seen as an extension of the this semantics from timed to hybrid automata with finite precision on the measurements.

The formalism of *Lazy Hybrid Automata* [2, 1, 15] also relaxes the ideal assumptions. It refers to HA whose dynamics is governed by a vector of constants rates, and where the plant state evolves continuously while the controller samples the plant state and changes the control state discretely at specific sampling times T_i. The relaxation of the instantaneous reaction hypothesis is done by distinguishing the delays δ_h on sensors and δ_g on actuators, while the finite precision is modeled by considering neighborhood of the guards. Transition may be fired when guards are satisfied at least one time in intervals, next to sampling points, of length δ_h. This does not ensure that transitions are fired also with smaller sensors delay (i.e. by setting $\delta'_h < \delta_h$) and then the property faster is better is not valid for lazy HA. The analysis of the behavior of lazy HA is based on state discretization, which may compromise scalability.

2. REACHABILITY FOR HYBRID AUTOMATA WITH URGENCY

In this section, we give our definition of *Hybrid Automata (HA)* with urgency. They include in each location an urgency condition represented by a nonconvex polyhedron. As in [19], we also include non-convex invariants.

2.1 Hybrid Automata with Urgency

We first need to define some notation. A *convex polyhedron* is a subset of $\mathbb{R}^n$ that is the intersection of a finite number of strict and non-strict affine half-spaces. A *polyhedron* is a subset of $\mathbb{R}^n$ that is the union of a finite number of convex polyhedra. For clarity, we write $\widehat{P}$ if P is convex. For a general (i.e., not necessarily convex) polyhedron $G \subseteq \mathbb{R}^n$, we denote by $[\![G]\!] \subseteq 2^{\mathbb{R}^n}$ its representation as a (minimal) finite union of convex polyhedra. The topological closure of P is denoted by $cl(P)$. Given an ordered set $X = \{x_1, \ldots, x_n\}$ of variables, a *valuation* is a function $v : X \to \mathbb{R}$. Let $Val(X)$ denote the set of valuations over X. There is an obvious bijection between $Val(X)$ and $\mathbb{R}^n$, allowing us to extend the notion of (convex) polyhedron to sets of valuations. We denote by $CPoly(X)$ (resp., $Poly(X)$) the set of convex polyhedra (resp., polyhedra) on X. Moreover, we denote by $SPoly(X)$ the subset of $\mathbb{R}^X$ that can be obtained by finite disjunction of closed convex polyhedra.

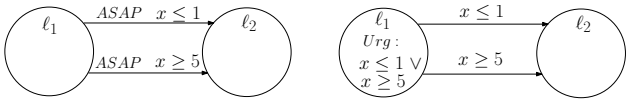
We use $\dot{X}$ to denote the set $\{\dot{x}_1, \ldots, \dot{x}_n\}$ of dotted variables, used to represent the first derivatives, and X' to denote the set $\{x'_1, \ldots, x'_n\}$ of primed variables, used to represent the new values of variables after a discrete transition. Arithmetic operations on valuations are defined in the straightforward way. An *activity* over X is a function $f : \mathbb{R}^{\geq 0} \to Val(X)$ that is continuous on its domain and differentiable except for a finite set of points. Let $Acts(X)$ denote the set of activities over X. The *derivative* $\dot{f}$ of an activity f is defined in the standard way and it is a partial function $\dot{f} : \mathbb{R}^{\geq 0} \to Val(\dot{X})$.

Formally, a hybrid automaton

$$H = (Loc, X, Lab, Inv, Urg, Flow, Trans, Init)$$

consists of the following components:

- a finite set Loc of *locations*;
- a finite set $X = \{x_1, \ldots, x_n\}$ of real-valued *variables*. A *state* is a pair $\langle \ell, v \rangle$ of a location ℓ and a valuation $v \in Val(X)$;
- a finite set of labels Lab;
- a set $Inv \subseteq Loc \times \mathbb{R}^n$, called the *invariant*. The system may only remain in a location ℓ as long as the state is inside its *invariant* $Inv(\ell)$.
- a set $Urg \subseteq Loc \times \mathbb{R}^n$, called the *urgency condition*. The urgency condition impedes time elapse, i.e., no continuous activities in a location ℓ continue from a valuation that satisfies the condition $Urg(\ell)$;
- a set $Flow \subseteq Loc \times \mathbb{R}^n \times \mathbb{R}^n$, defined over the first derivative of the variables, which determines how variables can change over time.
- a finite set $Trans$ of *discrete transitions* that describes instantaneous changes of locations, in the course of



(a) ASAP transitions explicitly modeled

(b) ASAP transitions modeled by urgent condition

Figure 2: ASAP transitions modeled by urgent

which variables may change their value. Each transition $(\ell, \alpha, \mathcal{G}, Asgn, \ell') \in Trans$ consists of a *source location* ℓ, a *target location* ℓ', a label $\alpha \in Lab$, a *guard* $\mathcal{G} \subseteq \mathbb{R}^n$ and an *assignment* $Asgn : \mathbb{R}^n \to 2^{\mathbb{R}^n}$. A state $\langle \ell, v \rangle$ can jump to $\langle \ell', v' \rangle$ if $v \in \mathcal{G}$ and $v' \in Asgn(v)$.

- a set $Init \subseteq Loc \times \mathbb{R}^n$, contained in the invariant, defining the *initial states* of the automaton. All behavior originates from these states.

The set of states of H is $S = Loc \times Val(X)$. Given a set of states A and a location ℓ, we denote by $A\,|_\ell$ the projection of A on ℓ, i.e. $A\,|_\ell = \{v \in Val(X) \mid \langle \ell, v \rangle \in A\}$.

The kind of dynamics modeled by the component *Flow* determines the special class of Hybrid Automata. When *Flow* is expressed by polyhedral inclusion, i.e. $Flow : Loc \to CPoly(\dot{X})$, we are talking about the subclass of *Linear Hybrid Automata*. Otherwise, the subclass of *Affine Hybrid Automata* is defined by affine continuous dynamics with uncertain inputs of the form

$$\dot{x}(t) = Ax(t) + u(t), u \in \mathcal{U},$$

where $x(t) \in \mathbb{R}^n$, A is a real-valued $n \times n$ matrix and $\mathcal{U} \subseteq \mathbb{R}^n$ is a closed and bounded convex set.

In both subclasses, the transition assignments *Asgn* are of the form $x' = Rx + w, w \in \mathcal{W}$, where R is a real-valued $n \times n$ matrix, and $\mathcal{W} \subset \mathbb{R}^n$ is a closed and bounded set of nondeterministic inputs.

Urgent Conditions and ASAP Transitions.

Urgency is used to model the so-called *must semantics* where, unlike the *may semantics* of standard HA, transitions must be taken immediately when associated guards are satisfied. Because urgent conditions impede elapsing of time, they can be satisfied only in a single time point. This point is the exact moment when the urgency is met (i.e. the frontier of the urgent condition).

In our definition, the urgency condition is defined for each location. An alternative approach, popular mainly because of its syntactical simplicity, is to designate each discrete transition as urgent or not. This is also referred to as *as-soon-as-possible (ASAP) transitions*. As shown by Fig. 2, urgent transitions can easily be translated to an urgency condition: let $Trans_U \subseteq Trans$ be the set of urgent transitions. Then the equivalent urgency condition is the union of the outgoing guards,

$$Urg(\ell) = \bigcup \{\mathcal{G} | (\ell, \alpha, \mathcal{G}, Asgn, \ell') \in Trans_U\}.$$

2.2 Parallel Composition

One attractive feature of urgency is that a model can be decomposed for cases where this is not possible without urgency. Consider the example of an automaton for the plant and an automaton for the controller. Without urgency, the controller automaton can in general not prevent time elapse in the plant automaton, unless an additional clock is introduced and that clock is sampled periodically.

We give a brief formal definition of parallel composition with urgency for the case where both automata range over the same variables. The key here is that the urgency condition of the composition is the union of the urgency conditions of the operands. Given a hybrid automata H_1, H_2 with $H_i = (Loc_i, X, Lab_i, Inv_i, Urg_i, Flow_i, Trans_i, Init_i)$, their *parallel composition* is the HA $H = (Loc_1 \times Loc_2, X, Lab_1 \cup Lab_2, Edg, Flow, Inv, Urg, Init)$, written as $H = H_1 \| H_2$, where $Flow(l_1, l_2) = Flow_1(l_1) \cap Flow_2(l_2)$; $Inv(l_1, l_2) = Inv_1(l_1) \cap Inv_2(l_2)$; $Urg(l_1, l_2) = Urg_1(l_1) \cup Urg_2(l_2)$; $Init(l_1, l_2) = Init_1(l_1) \cap Init_2(l_2)$ and transitions $((l_1, l_2), \alpha, \mathcal{G}, Asgn, (l_2', l_2')) \in Edg$ iff

- $\alpha \in Lab_1 \cap Lab_2$, for $i = 1, 2$, $(l_i, \alpha, \mathcal{G}_i, Asgn_i, l_i') \in Edg_i$, $\mathcal{G} = \mathcal{G}_1 \cap \mathcal{G}_2$, $Asgn = Asgn_1 \cap Asgn_2$, or
- $\alpha \notin Lab_1$, $l_2' = l_2$, $(l_1, \alpha, \mathcal{G}, Asgn, l_1') \in Edg_1$, or
- $\alpha \notin Lab_2$, $l_1' = l_1$, $(l_2, \alpha, \mathcal{G}, Asgn, l_2') \in Edg_2$.

2.3 Run Semantics

The behavior of a HA is based on two types of steps: *discrete* steps correspond to the *Trans* component, and produce an instantaneous change in both the location and the variable valuation; *timed* steps describe the change of the variables over time in accordance with the *Flow* component.

Definition 1 (Discrete Step). Given two states s, s', and a transition $e = (loc(s), \alpha, \mathcal{G}, Asgn, loc(s')) \in Trans$, there is a *discrete step* $s \xrightarrow{e} s'$ with *source* s and *target* s' iff

(i) $s, s' \in Inv$,
(ii) $val(s) \in \mathcal{G}$, and
(iii) $val(s') = Asgn(val(s))$.

Whenever (ii) holds, we say that e is *enabled* in s. Given a state $s = \langle \ell, v \rangle$, let $loc(s) = \ell$ and $val(s) = v$. An activity $f \in Acts(X)$ is called *admissible from* s if (i) $f(0) = v$ and (ii) for all $\delta \geq 0$, if $\dot{f}(\delta)$ is defined then $\dot{f}(\delta) \in Flow(\ell)$. We denote by $Adm(s)$ the set of activities that are admissible from s.

To take into account the urgency condition, which affects timed steps, we need to know the maximum amount of time δ such that the system, by following an activity f, is allowed to remain in a give location ℓ. Formally we define, for an activity $f \in Adm(s)$, the *Switching Time* of f in ℓ, denoted by $SwitchT(f, Urg(\ell))$, as the value $\delta \geq 0$ such that, for all $0 \leq \delta' < \delta$, $f(\delta') \notin Urg(\ell)$ and $f(\delta) \in Urg(\ell)$. When for all $\delta \geq 0$ it holds that $f(\delta) \notin Urg(\ell)$, we write $SwitchT(f, Urg(\ell)) = \infty$.

Definition 2 (Timed Step). Given two states s, s', there is a *timed step* $s \xrightarrow{\Delta, f} s'$ with *duration* $\Delta \in \mathbb{R}^{\geq 0}$ and activity $f \in Adm(s)$ iff

(i) for all $0 \leq \delta' \leq \Delta$, $(\langle \ell, f(\delta') \rangle) \in Inv$,
(ii) $s' = \langle loc(s), f(\Delta) \rangle$, and
(iii) $\Delta \leq SwitchT(f, Urg(loc(s)))$.

Conditions (i) says that the system always remains in the invariant $I = Inv(\ell)$ during the duration Δ of the step,

while condition (iii) says that during this step the system can satisfy the urgency condition only at time Δ.

A *run* is a sequence

$$r = s_0 \xrightarrow{\delta_0, f_0} s_0' \xrightarrow{e_0} s_1 \xrightarrow{\delta_1, f_1} s_1' \xrightarrow{e_1} s_2 \cdots s_n \ldots \quad (1)$$

of alternating timed and discrete steps. If the run r is finite, we define $len(r) = n$ to be the length of the run, otherwise we set $len(r) = \infty$. The set $Runs(H)$ denotes the set of all runs of the automaton H.

Given a state $s \in S$ and a hybrid automaton H with initial set of states $Init$, s is said to be *reachable* in H if there exists a finite *run* $r = s_0 \xrightarrow{\delta_0, f_0} s_0' \xrightarrow{e_0} s_1 \xrightarrow{\delta_1, f_1} s_1' \xrightarrow{e_1} s_2 \cdots s_n$, such that $s_0 \in Init$ and $s_n = s$. We denote the set of reachable states by $Reach(H)$.

2.4 Reachability Computation

In this section we propose an extension of the algorithm proposed by [19] in order to compute the time-elapse operator for the class for automata with piecewise affine dynamics with urgent conditions. This problem is reduced to the computation of the time-elapse for non-convex invariants, for which so far no algorithm is available for piecewise affine dynamics.

2.4.1 Standard Post Operators

A classic algorithm to compute the set $Reach(H)$ is a fixed-point procedure based on a *continuous post operator* and on a *discrete post operator*: given a set of states $S' \subseteq S$, the continuous post is used to compute the states reachable from S' by following an admissible trajectory, while the discrete post is used to compute the states reachable from S' via discrete transitions.

Given a hybrid automaton H, a location $\ell \in Loc$, a set of valuations $P, I \subseteq Inv(\ell)$, $U = Urg(\ell)$, and a time horizon $T \geq 0$, the *Δ-timed horizon continuous post operator* $\Delta Post_\ell(P, I, T)$ contains the set of all valuations $v \in Val(X)$ reachable from some $u \in P$, within the time T by never leaving I:

$$\Delta Post_\ell(P, I, T) = \big\{ v \in val(X) \big| \exists 0 \leq \delta \leq T, u \in P, f \in Adm(\langle \ell, u \rangle) : \forall 0 \leq \delta' \leq \delta : f(\delta') \in I \text{ and } f(\delta) = v \big\}. \quad (2)$$

The standard *continuous post operator* $Post_\ell(P, I)$ contains the set of all valuations $v \in Val(X)$ reachable from some $u \in P$, without leaving I:

$$Post_\ell(P, I) = \bigcup_{\delta \geq 0} \Delta Post_\ell(P, I, \delta) \quad (3)$$

The *discrete post operator* $Post_e(P)$ contains the set of all valuations $v \in Val(X)$ reachable from some $u \in P$ by taking the discrete transition $e = (\ell, \mathcal{G}, Asgn, \ell')$:

$$Post_e(P) = \big\{ v \in val(X) \big| \exists u \in P \cap \mathcal{G} \wedge v \in Inv(\ell') \big\}. \quad (4)$$

Notice that urgency does not affect discrete steps, hence the above definition of discrete post is still valid in case of urgency. From these operators on valuations we obtain the continuous and discrete post operators for a set of states S by iterating over all locations and transitions:

$$Post_c(S) = \bigcup_{\ell \in Loc} \{\ell\} \times Post_\ell(S\lfloor_\ell, Inv(\ell)), \quad (5)$$

$$Post_d(S) = \bigcup_{(\ell, \alpha, \mathcal{G}, Asgn, \ell') \in Trans} \{\ell'\} \times Post_{(\ell, \alpha, \mathcal{G}, Asgn, \ell')}(S\lfloor_\ell). \quad (6)$$

Note that definition (3) is valid regardless whether I is convex or not. For the sake of clarity, we will denote the continuous post with $Post_\ell(P, I)$ when I is convex, and with $ncPost_\ell(P, I)$ when I is nonconvex.

The reachable states $Reach(H)$ are computed as the smallest fixed point of the sequence $S_0 = Post_c(Init)$ and $S_{k+1} = S_k \cup Post_c(Post_d(S_k))$.

2.4.2 Computation of ncPost

We describe here an approach for computing the continuous post when the invariant I of a location is a non-convex and closed polyhedron. The technique is similar to the one proposed in [19] for piecewise constant dynamics. However, the proof differs since [19] relies entirely on straight-line trajectories.

We propose a characterization of $ncPost_\ell$ based on the standard $Post_\ell$ evaluated iteratively over a convex cover of I. Given an affine HA $\mathcal{H}$ and let $\ell \in Loc$, $I = Inv(\ell)$ and consider the initial set $P \subseteq I$. For each convex component $\widehat{I}' \in [\![I]\!]$, we apply the standard continuous post operator with $P \cap \widehat{I}'$ as initial set and $\widehat{I}'$ as invariant. The procedure is applied recursively by building a sequence $W_0 \subseteq W_1 \subseteq \ldots W_k$ of reachable valuations, with $W_0 = P$. Then $ncPost_\ell(P, I)$ is the fixed point of that sequence. This is formalized by the following theorem:

Theorem 1. Given a location $\ell \in Loc$ and let $I = Inv(\ell)$ be the invariant in location ℓ. For every set of valuations $P \subseteq I$ such that the sequence $W_0 = P$,

$$W_{k+1} = \bigcup_{\widehat{I}' \in [\![I]\!]} Post_\ell(W_k \cap \widehat{I}', \widehat{I}')$$

satisfies $W_k = W_{k+1}$ for some k, then $ncPost_\ell(P, I) = W_k$.

Proof. (Sketch – for a full proof see [20]). To show that $ncPost_\ell(P, I) \subseteq W_k$, we show that $v \in ncPost_\ell(P, I)$ implies that $v \in W_{i+1}$ for some $i \leq k$. Let u be a valuation in P. We proceed by induction over the number of crossings between convex components of I to get from u to v along some activity. Since the activities are analytic functions, the number of crossings is finite (analytic functions have isolated zeros).

The induction hypothesis is as follows: If $v_i \in W_i$ be reachable through $i-1$ crossings of convex components of I, then any valuation reachable from v_i by crossing one more convex component belongs to W_{i+1}. The base case for $i = 1$ is straightforward, since it refers to states reachable inside a single convex component of I, i.e., the classic convex post operation. Suppose that during a run leading from u to v, the last crossing involves the convex components $\widehat{I}_{i-1}, \widehat{I}_i \in I$. Hence there exists a valuation $u' \in \widehat{I}_{i-1} \cap \widehat{I}_i$ reachable with $i-1$ crossings and then by inductive hypothesis $u' \in W_i$. But since the last crossing is from $\widehat{I}_{i-1}$ to $\widehat{I}_i$, then from u' to v the system always remains inside $\widehat{I}_i$, meaning that $v \in Post_\ell(W_i \cap \widehat{I}_i, \widehat{I}_i) \subseteq W_{i+1}$.

We now show $W_k \subseteq ncPost_\ell(P, I)$. $v \in W_{i+1}$ means that there exists a convex component $\widehat{I}'$ and a valuation $u' \in W_i$ such that $v \in Post_\ell(W_i \cap \widehat{I}', \widehat{I}')$. This means that v is reachable from some $u' \in W_i \cap \widehat{I}'$ and by inductive

hypothesis $u' \in ncPost_\ell(P, I)$ and there exists a valuation $u \in P$ such that u' is reachable from u with $i-1$ crossings. Suppose that u' also belongs to another convex component $\widehat{I}''$, this means that in order to reach v from u' the system needs to complete a further crossing from $\widehat{I}''$ to $\widehat{I}'$ and hence $v \in ncPost_\ell(P, I)$ by performing i crossings. Otherwise v already belongs to W_i and then we can apply the inductive hypothesis to conclude the proof. □

Theorem 1 provides an effective way to compute an overapproximation of the post with non-convex invariants. Indeed several algorithms for computing overapproximation of the standard post with convex invariants are available, and they can be used to compute overapproximation of the sets W_i. The *overapproximative post operator*, whose error is bounded by the non-negative real number $\upsilon \geq 0$, is denoted by $Post_\ell^\upsilon$. The definition of the *overapproximative reachability set* denoted by $Reach^\upsilon$, is straightforward. The following result is a direct consequence of Theorem 1.

Corollary 1. Given a location $\ell \in Loc$ and let $I = Inv(\ell)$ be the non-convex invariant in location ℓ and $\upsilon \geq 0$ a non-negative number. For every set of valuations $P \subseteq I$ such that the sequence $W_0 = P$,

$$W_{k+1} \quad = \quad \bigcup_{\widehat{I}' \in [\![I]\!]} Post_\ell^\upsilon(W_k \cap \widehat{I}', \widehat{I}')$$

reaches fixed point in k iterations, then W_k is an overapproximation of $ncPost_\ell(P, I)$ with error $k\upsilon$.

2.4.3 Computation of UPost

We now define the continuous post operator under urgency conditions, and show that it can be approximated using the post operator for nonconvex invariants. The *urgent continuous post operator* is

$$\begin{aligned} UPost_\ell(P, I, U) = \big\{ v \in val(X) \,\big|\, & \exists \delta \geq 0, u \in P, \\ & f \in Adm(\langle \ell, u \rangle) : \forall 0 \leq \delta' \leq \delta : f(\delta') \in I, \\ & f(\delta) = v, \delta \leq SwitchT(f, U) \big\}. \end{aligned} \quad (7)$$

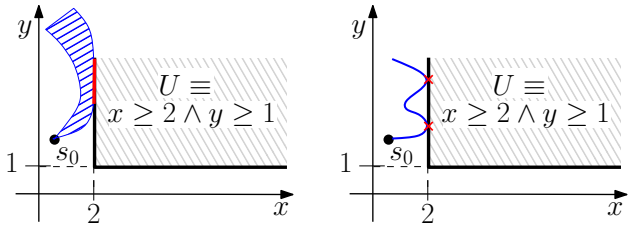
The intuition is that, since an urgent condition U blocks activities that inside U, the complement of U can be added to the invariant. This allows time-elapse only outside U. However, the system can reach the border of U, which is not included in the complement of U. Hence we include the topological closure of $\overline{U}$ in the invariant, at the price of including spurious behavior. The spurious behavior arises from admissible activities that continue after touching the border of the urgent condition, as illustrated in Figure 3. They are excluded if U is enlarged even slightly. Let the *ε-ball* of some norm $\|\cdot\|$ be $\mathcal{B}_\varepsilon = \{x \mid \|x\| \leq \varepsilon\}$.

Theorem 2. Given a location $\ell \in Loc$. Let P be a set of valuations, $I = Inv(\ell)$ be the invariant in location ℓ and $U = Urg(\ell)$ be the urgent condition of ℓ. The urgent post operator $UPost_\ell(P, I, U)$ can be overapproximated by

$$UPost_\ell(P, I, U) \subseteq ncPost_\ell(P, I \cap cl(\overline{U})),$$

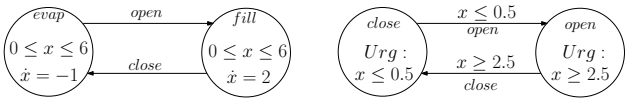
and any $\varepsilon > 0$,

$$ncPost_\ell(P, I \cap cl(\overline{U \oplus \mathcal{B}_\varepsilon})) \subseteq UPost_\ell(P, I, U).$$



(a) the red line is computed but is not in $UPost_\ell$ (b) the red crosses are computed but are not in $UPost_\ell$

Figure 3: The two cases when the computation of $UPost_\ell$ is an approximation



(a) tank (b) controller

Figure 4: Hybrid automata modeling the Water Tank Controller (WTC)

3. RELAXATIONS OF HYBRID AUTOMATA

The standard semantics of hybrid automata, as presented in Sect. 2, may be problematic if the goal is to verify a given simulation model. They are a mathematical idealization, while numerical simulation is based on time discretization and finite precision arithmetic.

ASSUMPTION 1. *An execution of the simulation model approximates each activity $f(t)$ at time points $t_0, t_1, \ldots$ with a sequence of valuations $v_0, v_1, \ldots$ such that for all $i = 0, 1, \ldots$,*

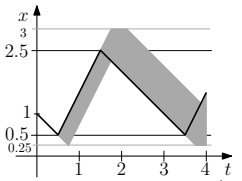
(i) $t_{i+1} - t_i \leq \Delta$ *(bounded time step), and*

(ii) $\|f(t_i) - v_i\| \leq \varepsilon$ *(bounded precision).*

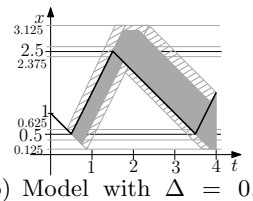
For the purpose of this paper, assumption *(i)* can be relaxed so that it applies only to states within a neighborhood of the urgent guards. Assumption *(ii)* is satisfied by most ODE solvers, except that the global bound ε is usually not known in absolute terms. Instead, solvers guarantee convergence as the time step goes to zero, e.g., $\varepsilon = \mathcal{O}(\Delta^p)$, where p is the order of the solver. ODE solvers that guarantee global bounds are available [22], but convergence and scalability are more challenging when conservative error bounds are imposed.

Example 1. We use a simplification of the Water Tank Control (*WTC*) model reported in [11] to illustrate why a verification model can be safe while its simulation model can be instead unsafe. The WTC consists of a single tank with a valve that can be controlled to add water. The water is subject to natural evaporation. The level of the water inside the tank is monitored by a sensor. A verification model of the WTC is used to check whether, via a sequence of opening and closing of the valve, it is possible to maintain the level of the water within given bounds.

The verification model consists of the hybrid automata in Fig. 4. The tank model is shown in Fig. 4(a). A continuous variable x models the water level. Two locations are used



(a) Verification model (solid line) and simulation model with $\Delta = 0.25$ sec. (gray area)



(b) Model with $\Delta = 0.25$ and measurement error of $\epsilon = 0.125$

Figure 5: Reachability sets (over the first 4 sec.) for verification and simulation models for WTC

to model when x is decreasing (i.e. closed valve and evaporation) and increasing (i.e. open valve). The automaton for the controller is shown by Fig. 4(b). It has two locations for modeling the open and close commands for the valve. Since the desired safety property is to maintain x inside the interval $S = [0.5, 2.5]$, the urgent (ASAP) guards are $x \leq 0.5$ for opening and $x \geq 2.5$ for closing the value; the urgency conditions of the locations are the union of the urgent guard conditions.

The reachability analysis under standard semantics shows the safety of the verification model. The solid black line of Fig. 5(a) is a graphical representation of the reachability set from initial condition $x = 1$, over the first 4 sec., and it is easy to see that x never exceeds the limits 0.5 and 2.5.

A numerical simulation with an upper bound Δ on the time step may recognize an urgent guard condition during the next sample time, thus by delaying the reaction by Δ units of times. This delay leads the system to unsafe states, as shown the gray area of Fig. 5(a). For $\Delta = 0.25$ sec., the system may exceed the upper limit by 0.5 units by delaying the switch Δ extra time with flow $\dot{x} = 2$. Similarly, the lower limit may be exceeded by 0.25 units.

To capture the finite precision computation, let the approximate value be $\hat{x} = x + \epsilon'$, where $\epsilon' \in \varepsilon = [-\epsilon, +\epsilon]$. The guard check of the simulator evaluates $\hat{x} \in \mathcal{G}$. As a consequence, the resulting reachable set contains some extra valuations, as depicted in Fig. 5(b) by the tiled gray area for $\Delta = 0.25$ sec. and $\epsilon = 0.125$.

3.1 Relaxed Semantics

In this section we define relaxed semantics for hybrid automata. The semantics uses as parameters a bound Δ on the time step of the simulation model and a bound ϵ on the precision error, as defined in Assumption 1.

To formally define the relaxed semantics, we need some extra notation. Let P be a set of valuations and $\epsilon \geq 0$. The *ε-enlargement* of P is the set

$$\lceil P \rceil \varepsilon = \{w \mid \exists v \in P : \|v - w\| \leq \varepsilon\} = P \oplus \mathcal{B}_\varepsilon,$$

where $\oplus$ denotes the Minkowski sum and $\mathcal{B}_\varepsilon$ is the ball of size ε of the chosen norm. The *ε-shrinkage* of P is the set

$$\lfloor P \rfloor_\varepsilon = \bigcap_{b \in \mathcal{B}_\varepsilon} \{v - b \mid v \in P\} = P \ominus \mathcal{B}_\varepsilon.$$

We now define the relaxed semantics. Let $\Delta \geq 0$ be the bound on the time step and $\epsilon \geq 0$ be the precision error. A *discrete step* in relaxed semantics is obtained from a discrete step in standard semantics, Def. 1, by replacing condition (ii), the check of the guard condition, with the following condition that accounts for the precision error:

(ii)* $\exists e \in \mathcal{B}_\varepsilon$: $val(s) + e \in \mathcal{G}$.

To define a relaxed time step, we use a relaxed definition of the switching time, which takes into account that the urgent condition is checked only at discrete time points, and that the check is subjected to the precision error. First, time may elapse until the urgency condition U is satisfied even when taking into account all possible precision errors. Second, since the urgency condition is only checked at discrete time points, the urgency condition must be satisfied for at least Δ time to be sure that a check takes place. Both phenomena are captured by the following relaxation of the switching time. The *relaxed switching time* of an activity $f \in Adm(s)$ in location ℓ, denoted by $SwitchT_\varepsilon^\Delta(f, Urg(\ell))$, is the value $T + \Delta \geq 0$ such that, for all $0 \leq \delta' < T$, there exists some precision error $e \in \mathcal{B}_\varepsilon$ such that $f(\delta') + e \notin Urg(\ell)$, and for all $T \leq \delta' \leq T + \Delta$ and for all precision errors $e \in \mathcal{B}_\varepsilon$, $f(\delta') + e \in Urg(\ell)$. When such a value does not exists, we write $SwitchT_\varepsilon^\Delta(f, Urg(\ell)) = \infty$.

A *timed step* in relaxed semantics is obtained from a timed step in standard semantics, Def. 2, by replacing condition (iii), the check of the urgency condition via the switching time, with the relaxed switching time:

(iii)* $\delta \leq SwitchT_\varepsilon^\Delta(f, Urg(loc(s)))$.

The definition of a *relaxed run* is straightforward, as well as the resulting definition of reachability. We denote the set of the run under relaxed semantics by $Runs_\epsilon^\Delta(H)$ and the set of reachable states under relaxed semantics by $Reach_\epsilon^\Delta(H)$.

The following properties about relaxed semantics are intuitive, and the related proofs can be found in [20].

Property 1. [Faster is better] Let H be a hybrid automaton and given $\epsilon \geq 0$. For any $\Delta_1, \Delta_2 \in \mathbb{R}^{\geq 0}$ such that $\Delta_1 \leq \Delta_2$, it holds that $Reach_\varepsilon^{\Delta_1}(H) \subseteq Reach_\varepsilon^{\Delta_2}(H)$.

Property 2. [Precision is a quality] Let H be a hybrid automaton and given $\Delta \geq 0$. For any $\epsilon_1, \epsilon_2 \in \mathbb{R}^{\geq 0}$ such that $\epsilon_1 \leq \epsilon_2$, it holds that $Reach_{\varepsilon_1}^\Delta(H) \subseteq Reach_{\varepsilon_2}^\Delta(H)$.

Property 3. [Preserving Safety] Let H be a hybrid automaton, $\Delta' > 0$ be a bound on the sampling time and $\epsilon' > 0$ be a bound on the precision error. If H is safe in relaxed semantics, then there is a bound $0 < \Delta \leq \Delta'$ on the sampling time and a bound on the precision error $0 < \varepsilon \leq \varepsilon'$ such that all executions of the simulation model are safe under Assumption 1.

3.2 Relaxed Hybrid Automata

In this section we will show that the relaxed semantics can be translated to an equivalent relaxed model evaluated under standard semantics. The advantage of having this equivalence is that the correctness of a simulation model could be proved by using the standard reachability on the relaxed model.

We already show the reachability set $Reach_\epsilon^\Delta(H)$ for the WTC example obtained with a time step bounded by $\Delta = 0.25$ sec and a global error measurement bounded by $\epsilon = 0.125$ (see the gray area depicted by Fig. 5(b)).

Consider now another version of controller modeled by the linear hybrid automaton H', obtained from the by replacing

guards and urgent conditions of automaton H depicted by Fig. 4(b). Now the urgent conditions are less restrictive (i.e. $x \leq 0.125$ and $x \geq 3.125$ for the two locations, respectively), as well as the guards (i.e. $x \leq 0.625$ and $x \geq 2.375$, respectively). The obtained relaxed model is such that, under the standard semantics, the model may perform a discrete transition when $x \in (0.125, 0.625]$ or $x \in [2.375, 3.125)$, while it must jump when $x = 0.125$ or 3.125. Accordingly the reachability set $Reach(H')$ of such a model is clearly bounded by the interval $\in [0.125, 3.125]$, and moreover is completely equivalent to the relaxed reachability set $Reach_\epsilon^\Delta(H)$. In other words, the standard reachability set on the relaxed model is equal to the relaxed reachability set on the standard model. The key point behind this equivalence is that the relaxed automaton was obtained by properly relaxing urgent conditions and guards of transitions. In particular, guard are enlarged according to the interval error ε. Instead urgent condition are shrunk according to ε and in addition in a way that allows time-elapse for further Δ units of time, where Δ is the bound of the time step. The following definition is the formalization of the urgency relaxation.

Definition 3 (Urgency *Relaxation*). Given a location $\ell \in Loc$, and let $U = Urg(\ell)$ be the urgent condition location ℓ, $\Delta \geq 0$ and $\epsilon \geq 0$ be bounds on time step and precision error, respectively.

The *relaxation* of the urgency condition U is the set of valuations

$$\begin{aligned} \lfloor U \rfloor_\varepsilon^\Delta = \big\{ & p \in \lfloor U \rfloor_\varepsilon | p \notin Post_\ell(\overline{\lfloor U \rfloor_\varepsilon}, \mathbb{R}^n) \text{ or} \\ \exists u \in \overline{\lfloor U \rfloor_\varepsilon}, f \in Adm(\langle \ell, u \rangle)), 0 < \delta_1 < \delta_2 \text{ with } \delta_2 \geq \delta_1 + \Delta : \\ & \forall 0 \leq \delta' < \delta_1, f(\delta') \notin \lfloor U \rfloor_\varepsilon, \\ & \forall \delta_1 \leq \delta' \leq \delta_2, f(\delta') \in \lfloor U \rfloor_\varepsilon, \text{ and} \\ & f(\delta_2) = p \big\} \end{aligned} \quad (8)$$

There is an immediate relation between the relaxed switching time of an urgent condition U and the standard switching time of a corresponding relaxed condition, as stated by the following lemma.

Lemma 1. Given a location $\ell \in Loc$, and let $U = Urg(\ell)$ be a urgent condition, $\Delta, \epsilon \geq 0$ be upper bounds on time step and precision error, respectively. Then for all $u \in \lfloor U \rfloor_\varepsilon \cap$ and $f \in Adm(\langle \ell, u \rangle)$,

$$SwitchT(f, \lfloor U \rfloor_\varepsilon^\Delta) = SwitchT_\varepsilon^\Delta(f, U).$$

A proof of Lemma 1 can be found in [20].

Now we are ready to formalize the automaton relaxed by considering a reaction delay Δ and a global measurement error ϵ.

Definition 4 (Relaxed Hybrid Automaton). Let $\mathcal{H} = (Loc, X, Lab, Inv, Urg, Flow, Trans, Init)$ be a hybrid automaton, $\Delta, \epsilon \geq 0$ be upper bounds on time step and error precision, respectively. Then the *relaxed hybrid automaton* $\mathcal{H}_\varepsilon^\Delta = (Loc, X, Lab, Inv, Urg', Flow, Trans', Init)$ is such that:

- $Urg' = \big\{ \langle \ell, \lfloor U \rfloor_\varepsilon^\Delta \rangle | \langle \ell, U \rangle \in Urg \big\}$
- $Trans' = \big\{ \langle \ell_1, \alpha, \lceil \mathcal{G} \rceil_\varepsilon, Asgn \rangle \mid \langle \ell_1, \alpha, \mathcal{G}, Asgn \rangle \in Trans \big\}$

As already informal discussed, there is a direct relation that connects an automaton $\mathcal{H}$ evaluated under relaxed semantics and its relaxation $\mathcal{H}_\varepsilon^\Delta$ evaluated on standard semantics. As the intuition suggests, the corresponding sets of all the runs are equivalent, as formalized by the following theorem (the proof can be found in [20]).

Theorem 3. [Relaxed Semantics Equivalence] Given a hybrid automaton $\mathcal{H} = (Loc, X, Lab, Inv, Urg, Flow, Trans, Init)$. Let $\Delta \geq 0$ be a reaction delay, and $\epsilon \geq 0$ be a global measurement error. The relaxed automaton $\mathcal{H}_\varepsilon^\Delta = (Loc, X, Lab, Inv, Urg', Flow, Trans', Init)$ is such that

$$Runs_\epsilon^\Delta(\mathcal{H}) = Runs(\mathcal{H}_\varepsilon^\Delta).$$

From Theorem 3 directly follows the reachability equivalence, as stated by the following corollary

Corollary 2. Given a hybrid automaton $\mathcal{H} = (Loc, X, Lab, Inv, Urg, Flow, Trans, Init)$. Let $\Delta \geq 0$ be a reaction delay, and $\epsilon \geq 0$ be a global measurement error. The relaxed automaton $\mathcal{H}_\varepsilon^\Delta = (Loc, X, Lab, Inv, Urg', Flow, Trans', Init)$ is such that

$$Reach_\epsilon^\Delta(\mathcal{H}) = Reach(\mathcal{H}_\epsilon^\Delta).$$

Corollary 2 has a practice consequence on the task of proving safety w.r.t. a safe state S of a simulation model via reachability of a corresponding verification model H. Indeed, Prop. 3 says that this can by done by checking whether $Reach_\epsilon^\Delta(\mathcal{H}) \cap S$ is empty or not. But by Corollary 2 this is equivalent to check whether $Reach(\mathcal{H}_\varepsilon^\Delta) \cap S$ is empty or not. In other words, the safety of a simulation model can be proved via the standard reachability on the corresponding relaxed automaton as formalized by the following corollary

Corollary 3. Given a time step bounded by $\Delta > 0$, an error measurement bounded by $\epsilon > 0$ and a safe state S. Let $\mathcal{H}_\epsilon^\Delta$ be a relaxed automaton. Then there exists a safe implementation of $\mathcal{H}_\epsilon^\Delta$ by a simulation model with time step bounded by $0 < \Delta' \leq \Delta$ and an error measurement bounded by $0 < \epsilon' \leq \epsilon$ if and only if

$$Reach(\mathcal{H}_\epsilon^\Delta) \cap S = \emptyset.$$

Theorem 3 establishes the equivalence between relaxed semantics on ideal models and standard semantics on relaxed models. Now we present how to compute the corresponding relaxed automaton.

The following lemmas gives a way to compute the enlargement and shrinkage operators for the case of polyhedral sets. The exact solution would be of exponential complexity in the number of variables. To keep the computation scalable, we use conservative approximations.

Lemma 2. [Enlargement and Shrinkage] Given a convex polyhedron $P = \big\{ x \in \mathbb{R}^n \big| \bigwedge_i a_i^T x \leq b_i \big\}$ and a real number $\epsilon \geq 0$,

$$\lceil P \rceil_\varepsilon \subseteq \big\{ x \in \mathbb{R}^n \mid \bigwedge_i a_i^T x \leq b_i + \epsilon \|a_i\| \big\}, \text{ and}$$

$$\big\{ x \in \mathbb{R}^n \mid \bigwedge_i a_i^T x \leq b_i - \epsilon \|a_i\| \big\} \subseteq \lfloor P \rfloor_\varepsilon.$$

While the enlargement and shrinkage operations on polyhedra in Lemma 2 are only approximative, the approximation error goes to zero as $\epsilon \to 0$. This is sufficient for the purposes of this paper and the results that follow. The next lemma establishes how to compute the relaxation of urgent conditions.

Lemma 3. [Computation for relaxed urgency] The relaxed urgency condition $\lfloor U \rfloor_{\varepsilon}^{\Delta}$ can be computed as:

$$\lfloor U \rfloor_{\varepsilon}^{\Delta} = \lfloor U \rfloor_{\epsilon} \setminus \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta).$$

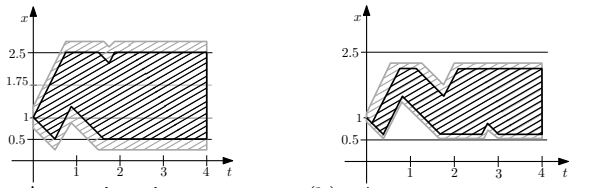
PROOF. [$\subseteq$] Let p be a valuation belonging to $\lfloor U \rfloor_{\varepsilon}^{\Delta}$. The definition of relaxation implies that $p \in \lfloor U \rfloor_{\varepsilon}$ and either (a) $p \notin Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n)$ or (b) there exists a valuation $u \in \overline{\lfloor U \rfloor_{\varepsilon}}$, an activity $f \in Adm(\langle \ell, u \rangle)$ and times $0 < \delta_1 < \delta_2$ with $\delta_2 \geq \delta_1 + \Delta$ such that for all $0 \leq \delta' < \delta_1$ it holds that $f(\delta') \notin \lfloor U \rfloor_{\varepsilon}$, for all $\delta_1 \leq \delta' \leq \delta_2$ it holds that $f(\delta') \in \lfloor U \rfloor_{\varepsilon}$, and $f(\delta_2) = p$.

Case (a) implies that $p \notin \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$, and hence $p \in \lfloor U \rfloor_{\epsilon} \setminus \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$. Otherwise, for case (b) suppose by contradiction that $p \in \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$, that is there exists a valuation $u \in \overline{\lfloor U \rfloor_{\varepsilon}}$, an activity $f \in Adm(\langle \ell, u \rangle)$ and a time $\delta \leq \Delta$ such that $f(\delta) = p$. Hence, for reaching $p \in \lfloor U \rfloor_{\varepsilon}$ from $u \in \overline{\lfloor U \rfloor_{\varepsilon}}$ the system spends at most $\delta < \Delta$ time inside $\lfloor U \rfloor_{\varepsilon}$. The last would imply $p \notin \lfloor U \rfloor_{\varepsilon}^{\Delta}$, that is in contrast with the hypothesis $p \in \lfloor U \rfloor_{\varepsilon}^{\Delta}$. Hence $p \notin \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$, and by recalling that $p \in \lfloor U \rfloor_{\varepsilon}$ we can write $p \in \lfloor U \rfloor_{\varepsilon} \setminus \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$.

[$\supseteq$] Let p be a valuation belonging to $\lfloor U \rfloor_{\epsilon} \setminus \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$. That is $p \in \lfloor U \rfloor_{\epsilon}$ and $p \notin \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$. Last one condition means that either (a) $p \notin Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n)$ or (b) there exists a time $\Delta_1 > \Delta$ such that $p \in \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta_1)$. By definition of relaxation of urgency, case (a) trivially implies that $p \in \lfloor U \rfloor_{\varepsilon}^{\Delta}$. Otherwise case (b) says that there exist a valuation $u \in \overline{\lfloor U \rfloor_{\varepsilon}}$, an activity $f \in Adm(\langle \ell, u \rangle)$ and a time δ with $\Delta < \delta \leq \Delta_1$, such that $f(\delta) = p$. On the trajectory leading from $u = f(0) \in \overline{\lfloor U \rfloor_{\varepsilon}}$ to $p = f(\delta) \in \lfloor U \rfloor_{\varepsilon}$ it is always possible to identify the last time δ^* with $\Delta < \delta^* \leq \delta$ such that for all $\delta^* \leq \delta' \leq \delta$ it holds that $f(\delta') \in \lfloor U \rfloor_{\varepsilon}$. If $\delta - \delta^* < \Delta$ then by definition $p \in \Delta Post_{\ell}(\overline{\lfloor U \rfloor_{\varepsilon}}, \mathbb{R}^n, \Delta)$ that contradicts the hypothesis. Hence $\delta - \delta^*$ must be at least equal to Δ and then by definition of relaxed urgency $p \in \lfloor U \rfloor_{\varepsilon}^{\Delta}$. □

3.3 Approximative Reachability

In general, the reachable states of a hybrid system with piecewise affine dynamics can only be computed approximatively. When checking safety properties, one can use conservative overapproximations to obtain soundness. But this may lead to false negatives, i.e., the analysis could indicate a violation of safety even if actually the system is safe. Concretely, if the reachability analysis indicates that the system is unsafe then it is not clear whether one should repeat the analysis with increased precision or whether one should conclude that the system is actually unsafe. It has long been argued, e.g., in [6], that real systems should robustly satisfy safety properties, and that including robustness assumptions in the semantics or the analysis can lead to decidability. A similar argument can be made for the relaxed semantics presented in this paper.



(a) Approximation error $\upsilon = 0.25$, delay $\Delta = 0.25$ and precision $\epsilon = 0.125$

(b) Approximation error $\upsilon = 0.1$, delay $\Delta = 0.1$ and precision $\epsilon = 0.1$

Figure 6: Reachability sets with different parameters for the relaxed WTC

For illustration, consider a safe relaxed hybrid automaton whose exact reachability set is depicted by the black area of Fig. 6(a). For this case, the reachability analysis with an approximation error of $\upsilon_1 = 0.25$ indicates a violation of safety even if the model is safe. A first possibility is to compute the reachability set with a smaller error, for example by setting $\upsilon' = 0.1$, but this may not be sufficient no matter how small υ'. In order to give more room to the approximation error, we can use smaller bound on the time step (for example $\Delta' = 0.1$ *sec*) and on the error measurement (for example $\epsilon' = 0.1$). With these parameters, the reachability analysis is now able to show the safety of the system, as shown by Fig. 6(b).

In order to formalize the feature described above, we introduce some extra notation. Let *Loc* be the set of locations, $S = \{\langle \ell, v \rangle \big\| \ell \in Loc, v \in Val(X)\}$ be a set of states and s be a state. With abuse of notation we use $bndry(s, S)$ to denote the boundary between the valuation of s and the set of valuations of S, that is $bndry(val(s), S\downharpoonright_{Loc})$.

The next assumption is enough to avoid false positive.

ASSUMPTION 2. *Let H_{ϵ}^{Δ} be a relaxed hybrid automaton with $\Delta > 0$ and $\epsilon > 0$, and S be a set of safe states. The reachability set of H_{ϵ}^{Δ} is such that each state $s \in Reach(H_{\epsilon}^{\Delta})$ with $bndry(s, \overline{S}) \neq \emptyset$ can only be reachable by a run $r = s_0 \xrightarrow{\delta_0, f_0} s_0' \xrightarrow{e_0} s_1 \xrightarrow{\delta_1, f_1} s_1' \xrightarrow{e_1} s_2 \cdots s_n' = s$ such that*

$$\exists \delta_i > 0 : \delta_i = SwitchT\Big(f_i, Urg\big(loc(s_i')\big)\Big).$$

Assumption 2 says that each reachable state s lying on the boundary with the unsafe set can be reachable only via a run where at least an urgent condition is satisfied after a non-zero elapsing of time. From a practical point of view we are assuming that critical behaviors of the systems are generated by those trajectories that at some point in time touch an urgent condition, and this is plausible if one considers that urgent conditions are used precisely in order to prevent unsafe behaviors. Moreover the condition of $\delta_i > 0$ implies that initial states do not belong to the unsafe state $\overline{S}$. And also this choice is plausible because initial states belonging to $\overline{S}$ make the system unsafe by definition.

Given a relaxed automaton H_{ϵ}^{Δ} and a set of safe states S, the practical sufficient condition to guarantee that H_{ϵ}^{Δ} satisfies Assumption 2 is the existence of an error location $err \in Loc$ accessible through urgent transitions, one from each location in the unsafe set. The urgent condition associated to these transitions is the topological closure of the unsafe set of valuations, that is $Z = cl(val(\overline{S}))$. Moreover, the set of initial valuations of H_{ϵ}^{Δ} must not belong to Z.

The following theorem is the formalization of the features described above, for a complete proof see [20].

Theorem 4. [Absorption of the overapproximation] Let H_ϵ^Δ be a relaxed hybrid automaton with $\Delta > 0$ and $\epsilon > 0$, and S be a set of safe states. If H_ϵ^Δ satisfies Assumption 2 and $Reach(H_\epsilon^\Delta) \cap \overline{S} = \emptyset$ then there exists an approximation error $\upsilon > 0$, a time step bound $0 < \Delta' < \Delta$ and a precision error bound $0 < \epsilon' < \epsilon$ such that $Reach^\upsilon(H_{\epsilon'}^{\Delta'}) \cap \overline{S} = \emptyset$.

4. REFERENCES

[1] M. Agrawal and P. Thiagarajan. Lazy rectangular hybrid automata. In R. Alur and G. Pappas, editors, *Hybrid Systems: Computation and Control*, volume 2993 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin Heidelberg, 2004.

[2] M. Agrawal and P. Thiagarajan. The discrete time behavior of lazy linear hybrid automata. In M. Morari and L. Thiele, editors, *Hybrid Systems: Computation and Control*, volume 3414 of *Lecture Notes in Computer Science*, pages 55–69. Springer Berlin Heidelberg, 2005.

[3] D. Beek, M.A., Reniers, R.R.H., Schiffelers, and J. Rooda. Foundations of a compositional interchange format for hybrid systems. In *HSCC'07*, volume 4416 of *LNCS*, pages 587–600. Springer, 2007.

[4] S. Bogomolov, D. Magazzeni, S. Minopoli, and M. Wehrle. Pddl+ planning with hybrid automata: Foundations of translating must behavior. In *Proceedings International Conference on Automated Planning and Scheduling, ICAPS*, volume 2015-January, pages 42–46. AAAI Press, 2015.

[5] J. T. Buck, S. Ha, E. A. Lee, and D. G. Messerschmitt. *Ptolemy: A framework for simulating and prototyping heterogeneous systems*. Ablex Publishing Corporation, 1994.

[6] M. Fränzle. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In *Computer Science Logic*, pages 126–139. Springer, 1999.

[7] G. Frehse. Reachability of hybrid systems in space-time. In *EMSOFT'15*, 2015.

[8] G. Frehse, C. L. Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV 11: Proc. of 23rd Conf. on Computer Aided Verification*, pages 379–395, 2011.

[9] G. Frehse and A. Paice. Optimal control of a gas compressor field. In *MTNS'00*, 2000.

[10] B. Gebremichael and F. Vaandrager. Specifying urgency in timed i/o automata. In *SEFM'05*, pages 64–74. IEEE Computer Society, 2005.

[11] W. Heemels, D. Lehmann, J. Lunze, and B. De Schutter. Introduction to hybrid systems. In *Handbook of Hybrid Systems Control – Theory, Tools, Applications*, pages 3–30. Cambridge University Press, Cambridge, UK, 2009.

[12] T. Henzinger. The theory of hybrid automata. In *11th IEEE Symp. Logic in Comp. Sci.*, pages 278–292, 1996.

[13] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: the next generation. In *Proc. IEEE Real-Time Systems Symposium (RTSS '95)*, page 56. IEEE Computer Society, 1995.

[14] P.-H. Ho. *Automatic Analysis of Hybrid Systems*. PhD thesis, Cornell University, Aug. 1995. Technical Report CSD-TR95-1536.

[15] S. Jha, B. Brady, and S. Seshia. Symbolic reachability analysis of lazy linear hybrid automata. In J.-F. Raskin and P. Thiagarajan, editors, *Formal Modeling and Analysis of Timed Systems*, volume 4763 of *Lecture Notes in Computer Science*, pages 241–256. Springer Berlin Heidelberg, 2007.

[16] MathWorks. Mathworks simulink: Simulation et model-based design, Mar. 2014. www.mathworks.fr/products/simulink.

[17] S. E. Mattsson, H. Elmqvist, and M. Otter. Physical system modeling with modelica. *Control Engineering Practice*, 6(4):501–510, 1998.

[18] S. Minopoli and G. Frehse. SL2SX tool and case study. www-verimag.imag.fr/~minopoli/SL2SXdemo.zip.

[19] S. Minopoli and G. Frehse. Non-convex invariants and urgency conditions on linear hybrid automata. In *12th International Conference on Formal Modeling and Analysis of Timed Systems*, 2014.

[20] S. Minopoli and G. Frehse. From simulation models to hybrid automata using urgency and relaxation. Technical Report TR-2015-10, Verimag, October 2015.

[21] S. Minopoli and G. Frehse. SL2SX translator: From simulink to spaceex models. In *HSCC'16*, 2016.

[22] N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105(1):21–68, 1999.

[23] L. V. Nguyen and T. T. Johnson. Dc-to-dc switched-mode power converters. In *1st Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. http://cps-vo.org/node/12113, 2014.

[24] X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. An approach to the description and analysis of hybrid systems. In *Hybrid Systems*, pages 149–178. Springer, 1993.

[25] M. W. Whalen, A. Murugesan, S. Rayadurgam, and M. P. E. Heimdahl. Structuring simulink models for verification and reuse. In *Proceedings of the 6th International Workshop on Modeling in Software Engineering*, MiSE 2014, pages 19–24, New York, NY, USA, 2014. ACM.

[26] M. Wulf, L. Doyen, and J.-F. Raskin. Almost asap semantics: From timed models to timed implementations. In *HSCC'04*, volume 2993 of *LNCS*, pages 296–310. Springer, 2004.