2012 13th International Workshop on Microprocessor Test and Verification

Guaranteeing termination of fully symbolic timed forward model checking

Georges Morbé
University of Freiburg,
Department of Computer Science
morbe@informatik.uni-freiburg.com

Christoph Scholl
University of Freiburg,
Department of Computer Science
scholl@informatik.uni-freiburg.com

***Abstract*—In this paper we present a normalization technique to guarantee termination of fully symbolic forward model checking for timed automata. Whereas for semi-symbolic model checkers based on convex clock zones there exist methods in the literature to solve this problem, our normalization algorithm can be applied to fully symbolic model checkers representing arbitrary symbolic (convex and non-convex) state sets. Our method is based on a projection of region-equivalent clock valuations to the same area within their equivalence class. In a first approach we present a normalization algorithm for diagonal-free timed automata. Then we generalize the approach to timed automata with diagonal constraints. We show that our normalization technique enables termination of the fixed point iteration in a prototype forward model checker using fully symbolic state set representations.**

I. Introduction

The application area of real-time systems grows with an enormous speed and along with that grows their complexity as well as the damage caused by their failure. For these reasons verification of such systems becomes more and more important. Timed automata (TAs) [1], [2] turned out to be an adequate formalism for modelling and verifying real-time systems. TAs generalize finite automata by adding real-valued clock variables. All clock variables evolve over time with the same rate and they can be reset during discrete steps which in turn happen in zero-time. Verifying safety properties of TAs can be reduced to the computation of all states from which unsafe states can be reached and checking whether some initial states are included in this set of states (backward model checking). Equivalently the problem can be reduced to the computation of all states reachable from the initial states and checking whether one of the reached states is unsafe (forward model checking).

Alur and Dill have shown that the model checking problem on TAs is decidable [1], [2]. They introduced equivalence classes for clock valuations, which are called *clock regions*. A clock region combines infinitely many region-equivalent clock valuations. This concept allows to reduce the model checking problem on TAs to a reachability problem on a region graph with finitely many clock regions. Based on their work a lot of model checking approaches considering reachability for TAs have been developed. They can be classified into *semi-symbolic* and *fully symbolic* approaches. Semi-symbolic approaches represent discrete locations of TAs explicitly whereas sets of clock valuations are represented symbolically e.g. by *unions of clock zones*. Clock zones are convex state sets which result from an intersection of clock constraints of the form $x_i \sim d$ and $x_i - x_j \sim d$ where $d \in \mathbb{Z}$, $\sim \in \{<, \leq, \geq, >\}$ and x_i, x_j are clock variables. UPPAAL [3], [4], the probably most prominent semi-symbolic approach, represents clock zones by so-called difference bound matrices (DBMs) and provides efficient methods for manipulating DBMs. These techniques are well-suited when the sizes of the discrete state space and the numbers of different clock regions per location remain moderate. CDDs [5] make the attempt to represent unions of clock zones more compactly. CDDs are BDD-like data structures where nodes are labelled by clock differences $x_i - x_j$. CRDs [6] are a variant of CDDs where outgoing edges of nodes are labelled by upper bounds for clock differences instead of disjoint intervals of rational numbers. CRDs were combined with BDDs (leading to CRD+BDDs) to provide a *fully* symbolic representation of the state space in the tool RED [6]. Another fully symbolic representation has been given by difference decision diagrams (DDDs) [7] which are basically BDD representations where the decision variables are boolean abstractions of clock constraints $x_i - x_j \sim d$. Computing all states reachable by evolution of time amounts to the existential quantification of a real-valued variable. Both for CRD+BDDs and DDDs this quantification is performed based on the classical Fourier–Motzkin technique which requires enumerating all paths in the diagram. As in DDDs, Seshia and Bryant [8] consider BDD representations using boolean abstractions of clock constraints, however they reduce real-valued quantifier elimination to adding so–called transitivity constraints followed by a series of quantifications for boolean variables. Another fully symbolic representation is given by Clock Matrix Diagrams (CMDs) [9]. CMDs basically correspond to CRD+BDDs where sequences of edges representing convex constraints are collapsed into single edges labelled by DBMs and boolean variables are restricted to the lowest levels in the variable orders. In [10] finite state machines with time (FSMTs) have been introduced. An FSMT is a formal model to represent real-time systems using transition functions and reset functions which is especially suited for fully symbolic verification algorithms. The FSMT model checking algorithm from [10] uses LinAIGs (‘And-Inverter-Graphs with linear constraints’) [11]–[13] as symbolic data structure, which provide a fully symbolic representation both for the continuous part (i.e. the clock values) and the discrete part (i.e. the state variables). For the quantification of real-valued variables, LinAIGs make use of the Weispfenning–Loos test point method [14] which is especially suitable for LinAIG representations.

In this paper we consider the problem that the basic forward reachability analysis on TAs is not guaranteed to terminate. Due to the lack of global upper bounds for clock variables, the clock values can rise ad infinitum during time evolution.

This can have the effect that during forward traversal the state set is infinitely often enlarged such that a fixed point will never be reached. [1] As already mentioned, on the other hand Alur and Dill [1], [2] proved that model checking of timed automata can be reduced to a finite problem by introducing a finite number of clock regions, i.e. equivalence classes of clock valuations wrt. reachability. Clock regions have the following property: Let S be a state set containing a clock valuation u, and let u be in the clock region R_u. If we change S by (1) including another $v_1 \in R_u$ ($v_1 \neq u$) into S, by (2) removing some $v_2 \in R_u$ ($v_2 \neq u$) from S or by (3) replacing u by some $v_3 \in R_u$ ($v_3 \neq u$), then the locations of the TA which are reachable from S do not change. Thus, we have the potential to enhance the basic forward reachability analysis by normalization techniques using operations (1) – (3) in order to reach the goal of termination guarantees for forward model checking.

For semi-symbolic model checking based on convex clock zones such a normalization technique has been introduced which relies on operation (1): If some $u \in R_u$ is included in a clock zone, the clock zone is transformed into a larger one including the complete set R_u. Using this normalization the fixed point iteration is guaranteed to terminate. [15], [16] presents an appropriate normalization algorithm for UPPAAL based on efficient DBM operations just removing or replacing clock constraints.

Adequate normalization algorithms providing termination guarantees for fully symbolic forward model checking did not exist so far (apart from simple approaches which transform non-convex state sets into (potentially large) unions of convex clock zones and normalize each convex clock zone by the method mentioned above). The goal of this paper is to provide a fully symbolic normalization algorithm which handles whole state sets at once. These possibly non-convex state sets are represented by symbolic formulas with discrete variables (encoding discrete locations) and continuous clock variables. In contrast to the algorithms for semi-symbolic model checking, where convex clock zones are enlarged, our normalization algorithm uses operation (3) mentioned above. It projects clock valuations onto 'representative clock valuations' from the same clock region and thus enables a fully symbolic forward model checker to reach a fixed point after a finite number of steps.

At first, we confine our consideration to TAs without diagonal constraints of the form $x_i - x_j \sim d$ and present a solution for this special case. Then we generalize to approach to TAs with such diagonal constraints appearing in guards. As done in [15], [16] for convex zones, we split all state sets along diagonal constraints in a preprocessing step. After the preprocessing step, we process the split state sets by a projection algorithm which is modified wrt. the presence of diagonal constraints.

The paper is organized as follows. In Sect. II we give a brief review of timed automata and we will define region-equivalence which our normalization technique is based on. In Sect. III we introduce our normalization technique for fully symbolic state sets in TAs without diagonal constraints. The problem with such diagonal constraints and the modification of our normalization algorithm to deal with it is given in Sect. IV. After giving some experimental results in Sect. V we conclude the paper in Sect. VI.

II. Preliminaries

Real-time systems are often represented as timed automata (TAs) [1], [2]. TAs use clock variables $X := \{x_1, \ldots, x_n\}$ to represent time. The set $\mathcal{C}(X)$ is the set of atomic clock constraints, containing simple boundary constraints of the form $(x_i \sim d)$ and diagonal constraints of the form $(x_i - x_j \sim d)$ with $d \in \mathbb{Z}$ and $\sim \in \{<, \leq, \geq, >\}$. Let $\mathcal{C}^{df}(X) \subseteq \mathcal{C}(X)$ contain only boundary constraints. Let $\mathcal{C}_c(X)$ be the set of conjunctions over clock constraints and similarly $\mathcal{C}_c^{df}(X)$ be the set of conjunctions over boundary constraints. An element $c \in \mathcal{C}_c(X)$ is called *clock zone* and describes a subset of $\mathbb{R}^n$, namely the set of all valuations of variables in X which evaluate c to true.

In general, transitions in TAs are labelled with guards and resets of clocks (see Fig. 2 for an example). Guards are restricted to conjunctions of clock constraints. A transition can only be taken, if its guard is satisfied. Resets are assignments to clock variables of the form $x := 0$. Invariants in TAs are conjunctions of clock constraints assigned to locations. A TA may stay in a location as long as the location invariant is not violated. Timed automata are formally defined as follows:

Definition 1 (Timed Automaton): A timed automaton (TA) is a tuple $\langle L, l_0, X, E, Inv\rangle$ where L is a finite set of locations, $l_0 \in L$ is an initial location, $X = \{x_1, \ldots, x_n\}$ is a finite set of real-valued clock variables, $E \subseteq L \times C_c(X) \times 2^X \times L$ is a set of transitions and the function $Inv : L \to \mathcal{C}_c^{df}(X)$ assigns a conjunction of boundary constraints as invariant to each location.

Definition 2 (Diagonal-free Timed Automaton): In a diagonal-free timed automaton the set of transitions $E \subseteq L \times C_c^{df}(X) \times 2^X \times L$ is restricted to transitions labelled with guards without diagonal constraints.

Definition 3 (Semantics of Timed Automata): Let $T = \langle L, l_0, X, E, Inv\rangle$ be a timed automaton. A state of T is a combination of a location and a valuation of the clock variables.

- There is a continuous transition ($s \to^c s'$) from state $s = (l, x_1^v, \ldots, x_n^v)$ to state $s' = (l, x_1^w, \ldots, x_n^w)$ iff $(x_1^v, \ldots, x_n^v)$ and $(x_1^w, \ldots, x_n^w)$ satisfy $Inv(l)$, and there is $t \in \mathbb{R}_0^+$ with $\forall 1 \leq j \leq n : x_j^w = x_j^v + t$.
- There is a discrete transition ($s \to^d s'$) from state $s = (l, x_1^v, \ldots, x_n^v)$ to state $s' = (l', x_1^w, \ldots, x_n^w)$ iff $(x_1^v, \ldots, x_n^v)$ satisfies $Inv(l)$, $(x_1^w, \ldots, x_n^w)$ satisfies $Inv(l')$, and $\exists e = (l, g_e, r_e, l') \in E$ with $(x_1^v, \ldots, x_n^v)$ satisfies the guard g_e, $x_i^w = 0$ for $x_i \in r_e$, $x_i^w = x_i^v$ for $x_i \notin r_e$.
- $\to = \to^d \cup \to^c$ is the transition relation of T. A trajectory of T is a finite or infinite sequence of states $(s^j)_{j \geq 0}$ with $s^0 = (l_0, 0, \ldots, 0)$ and $s^{j-1} \to s^j$ for each $j > 0$. A state is reachable, if there is a trajectory ending in that state.

The normalization technique we present in this paper is based on region-equivalence [1], [2]. We start our consideration with region-equivalence for diagonal-free TAs.

The idea of region-equivalence is as follows [17]:

If two states, which correspond to the same location of the timed automaton T, agree on the integral parts of all clock values and also on the ordering of the fractional parts of all

[1]Note that the situation is different in backward model checking: Here the (backward) time evolution is bounded, since clock values are always non-negative.

clocks, then the states will behave in the same manner. The integral parts of the clock values determine whether a clock constraint in the invariant of the location or in the guard of a transition is satisfied or not. The ordering of the fractional parts of the clock values determines which clock will change its integral part first. This is because clock constraints can involve only integers, and all clocks increase at the same rate. (Consider e.g. a TA T with two clock variables x_1 and x_2. Let l be a location of T and e an outgoing transition from l to l' with guard $g_e = x_1 \geq 5 \wedge x_2 \geq 4$. Then it is easy to see that if $(l, 3.2, 1.1)$ eventually satisfies the guard, then so will $(l, 3.9, 1.3)$.)

Moreover, whenever a clock value exceeds a certain upper limit, then the exact value of the clock is not important: If the clock x_i is never compared to a constant greater than $c^{df}(x_i)$ in the TA T, then the exact value of the clock will have no effect on the satisfiability of the clock constraints within T and thus no effect on the computation of T once it exceeds $c^{df}(x_i)$. With the knowledge that the value of x_i is greater than $c^{df}(x_i)$ the satisfiability of each boundary constraint containing x_i can be decided. (Consider e.g. a TA T with clock variable x_i which is never compared to a constant greater than $c^{df}(x_i) = 5$ in T. For the behavior of T it is irrelevant whether the value of x_i is 6 or 600. With both clock valuations the same constraints in T are satisfied or not.)

This motivates the definition of a clock ceiling function for every clock $x_i \in X$:

Definition 4 (Diagonal-free clock ceiling function): Let T be a diagonal-free timed automaton and X a set of clock variables. A clock ceiling function $c^{df} : X \rightarrow \mathbb{N}$ assigns a natural number as ceiling to each clock variable. $c^{df}(x_i) := max\,\{k_{x_i} | x_i \sim k_{x_i}$ is a subformula of a clock constraint in $T\,\}$. The natural number $c^{df}(x_i)$ is then the clock ceiling of the variable $x_i \in X$.

Based on the clock ceiling function region-equivalence of clock valuations allowing a finite representation for the infinite state space can be defined.

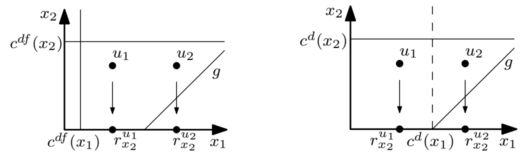
Definition 5 (Region-equivalence): For any $s \in \mathbb{R}$, $fr(s)$ denotes the fractional part of s and $\lfloor s \rfloor$ denotes the integral part of s such that $s = \lfloor s \rfloor + fr(s)$. Two clock valuations $u = (x_1^u, \ldots, x_n^u)$ and $v = (x_1^v, \ldots, x_n^v)$ are region-equivalent $u \equiv^{df} v$ in a TA, iff

1) for each clock variable x_i, either $\lfloor x_i^u \rfloor = \lfloor x_i^v \rfloor$ or both x_i^u and x_i^v are greater than $c^{df}(x_i)$, and
2) for each clock x_i, x_j if $x_i^u \leq c^{df}(x_i)$ and $x_i^v \leq c^{df}(x_i)$ then
   - $fr(x_i^u) = 0$ iff $fr(x_i^v) = 0$ and
   - $fr(x_i^u) \leq fr(x_j^u)$ iff $fr(x_i^v) \leq fr(x_j^v)$

The equivalence classes of $\equiv^{df}$ are called *clock regions*. The following Theorem 1 is shown in [1], [2] by Alur and Dill.

Theorem 1: Consider two states $s_1 = (l, u)$ and $s_2 = (l, v)$ in a diagonal-free TA with the region-equivalent clock valuations $u = (x_1^u, \ldots, x_n^u)$, $v = (x_1^v, \ldots, x_n^v)$. Then for every successor $s_1' = (l', u')$ of s_1 there exists a successor $s_2' = (l', v')$ of s_2 with $u' \equiv^{df} v'$ and vice versa.

From Theorem 1 it follows that two region-equivalent states in a diagonal-free TA can reach exactly the same locations. Let us consider w.l.o.g. only TAs where the safety property is encoded into the TA, i.e., the verification problem consists in the question whether an explicit error location is reachable. Then the verification result does not change, if we replace clock valuations with region-equivalent variants or add region-equivalent variants during reachability analysis.



(a) Error after $reset$ (b) Refined clock ceiling

Fig. 1. $reset$ may cause error

When we consider TAs with diagonal constraints, Def. 5 may cause problems, as can be seen in Fig. 1(a). Here u_i with $i \in \{1, 2\}$ are clock valuations, $r_{x_2}^{u_i}$ are the respective successors after a reset of the clock x_2 and g is a diagonal constraint in the TA. u_1 and u_2 are region-equivalent according to Def. 5.

However, after a reset of x_2 the successors $r_{x_2}^{u_1}$ and $r_{x_2}^{u_2}$ are no longer equivalent as $r_{x_2}^{u_1}$ and $r_{x_2}^{u_2}$ are on different sides of g. If there is a diagonal constraint in a guard of a TA, then Theorem 1 no longer holds. To define a notion of region-equivalence which is suited for TAs with diagonal constraints the clock ceiling function (Def. 4) and the definition of region-equivalence (Def. 5) must be adapted. The clock ceiling function must not only consider boundary constraints on clock variables but also diagonal constraints appearing in the guards of a TA [16].

Definition 6 (Refined clock ceiling function): Let T be a TA with diagonal constraints and X a set of clock variables. The clock ceiling function $c^d : X \rightarrow \mathbb{N}$ is defined as $c^d(x_i) := max\,\{k_{x_i} | x \sim k_{x_i}$ or $x - y \sim k_{x_i}$ is a subformula of a clock constraint in $T\,\}$. The natural number $c^d(x_i)$ is then the clock ceiling of the variable x_i.

The definition of region-equivalence in TAs with diagonal constraints uses the refined clock ceiling function of Def. 6 and has to ensure that two region-equivalent clock valuations are also equivalent with regard to the diagonal constraints in the TA [16].

Definition 7 (Refined region-equivalence): Let T be a TA with diagonal constraints and $\mathcal{C}_T^d(X)$ the set of diagonal constraints in T. Two clock valuations $u = (x_1^u, \ldots, x_n^u)$ and $v = (x_1^v, \ldots, x_n^v)$ are region-equivalent $u \equiv^d v$ in T, iff $u \equiv^{df} v$ with usage of the ceiling function of Def. 6 and $\forall g \in \mathcal{C}_T^d(X)$ it holds $u \in g \Leftrightarrow v \in g$.[2]

As the number of clock regions defined by $\equiv^{df}$ is finite and there are only finitely many diagonal constraints in a TA, the number of clock regions defined by $\equiv^d$ is also finite. Using Def. 7 we see that $r_{x_2}^{u_1}$ and $r_{x_2}^{u_2}$ in Fig. 1(b) are not region-equivalent. In contrast to Def. 5, we can also see that u_1 and u_2 are identified as non-equivalent due to the adaptation of the clock ceiling function.

III. Normalization for diagonal-free timed automata

In this section we will see how an arbitrary (possibly non-convex) state set in a diagonal-free TA can be normalized to guarantee termination of a fully symbolic forward model checking algorithm. Such a state set is given by a formula representing an arbitrary boolean combination of clock constraints and boolean variables (boolean variables encode locations in

[2]for a clock valuation u and a clock constraint g we write $u \in g$ if the valuation u satisfies the constraint g.

the TA). The state sets represented by such a symbolic formula can have an arbitrary shape.

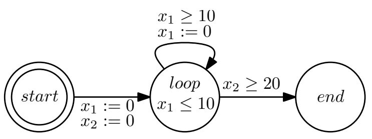


Fig. 2. Diagonal-free timed automaton

To illustrate the problem for which we need normalization, we consider the TA from [15] shown in Fig. 2. During forward traversal of the TA the location $loop$ is reached with a clock zone $z_1 = (x_1 \leq 10) \wedge (x_2 \leq 10) \wedge (x_1 = x_2)$. After taking the transition $loop \rightarrow loop$ a state $(loop, z_2)$ with $z_2 = (x_1 \leq 10) \wedge (10 \leq x_2 \leq 20) \wedge (x_2 - x_1 = 10)$ is reached. A second loop will add $(loop, z_3)$ to the state set with $z_3 = (x_1 \leq 10) \wedge (20 \leq x_2 \leq 30) \wedge (x_2 - x_1 = 20)$. The loop can be taken infinitely often enlarging the state set such that a fixed point will never be reached. This is caused by the fact that during forward traversal there are no global bounds preventing clock variables to grow ad infinitum. Normalization is used to identify different but region-equivalent states as equivalent and allows the forward traversal to reach a fixed point.

Based on region-equivalence (Def. 5) we will define a global upper bound for each clock variable. All valuation above these global constraints are region-equivalent.

Definition 8 (Global constraints): Let T be a TA and x_i a clock variable in T. The global constraint of x_i is a boundary constraint which determines the relevant area of x_i in T.

$$GC(x_i) := (x_i \leq \tilde{c}(x_i)) \tag{1}$$

with $\tilde{c}(x_i) = (c^{df}(x_i) + \epsilon)$, $\epsilon \in \mathbb{R}^+$.

In a diagonal-free TA with n clock variables $a = \bigwedge_{i=1}^{n}(x_i \leq c^{df}(x_i))$ determines the area above which the exact value of clock variables is irrelevant according to region-equivalence. Using the global constraints we define an area $\tilde{a} = \bigwedge_{i=1}^{n} GC(x_i)$ which completely covers a. Our normalization technique is based on a projection of all clock valuations above $\tilde{a}$ onto the borders of $\tilde{a}$; all clock valuations within $\tilde{a}$ remain untouched.

Let z be an arbitrary state set in a diagonal-free TA with n clock variables. The projection of z over a clock variable x_i is defined as

$$proj^{df}_{x_i}(z) := (z \wedge GC(x_i)) \vee \Big((x_i = \tilde{c}(x_i))\, \exists x_i \left(z \wedge \overline{GC(x_i)}\right)\Big) \tag{2}$$

It is easy to see that Lemma 1 holds:

Lemma 1: Let x_i be a clock variable, let z, z_1, and z_2 be state sets of a diagonal-free TA with $z = z_1 \vee z_2$, then

$$proj^{df}_{x_i}(z) = proj^{df}_{x_i}(z_1) \vee proj^{df}_{x_i}(z_2) \tag{3}$$

Lemma 1 gives us a means to simplify proofs by considering single states $(l, x_1^u, \ldots, x_n^u)$ to be projected instead of state sets z. Since the projection of Eqn. (2) does not change locations, we even omit l and consider only projections of clock valuations $u = (x_1^u, \ldots, x_n^u)$ (symbolically represented by formulas $\wedge_{i=1}^{n}(x_i = x_i^u)$) in our proofs.

In a TA with several clock variables a projection of a clock zone has to be done over all these variables. This can be done for all clocks separately with no concern of the order.

Lemma 2: Let x_i, x_j be two clock variables and z a state set, then

$$proj^{df}_{x_i}(\ proj^{df}_{x_j}(z)\ ) = proj^{df}_{x_j}(\ proj^{df}_{x_i}(z)\ ) \tag{4}$$

Proof: (Sketch) As mentioned above, we can consider only a single clock valuation u instead of the state set z in order to keep the proof simple. There are 4 cases to consider: (1) $u \in GC(x_i) \wedge GC(x_j)$, (2) $u \in GC(x_i) \wedge \overline{GC(x_j)}$, (3) $u \in \overline{GC(x_i)} \wedge GC(x_j)$ and (4) $u \in \overline{GC(x_i)} \wedge \overline{GC(x_j)}$. In every case Lemma 2 can be proven by inserting Eq. (2) into Eq. (4) followed by easy formula manipulations. ■

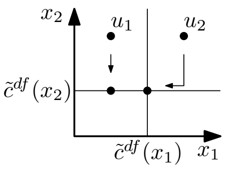


Fig. 3. Normalization

The diagonal-free normalization $norm^{df}(z)$ of a state set z is the consecutive projection over all clock variables in the TA.

Fig. 3 shows two examples of our normalization technique based on projection. Consider the clock valuation $u_1 = (x_1^{u_1}, x_2^{u_1})$. During projection only the value $x_2^{u_1}$ changes to $\tilde{c}(x_2)$, since $u_1 \in (GC(x_1) \wedge \overline{GC(x_2)})$. The result is a region-equivalent clock-valuation. Projecting $u_2 = (x_1^{u_2}, x_2^{u_2})$ will change the values of both variables resulting in $(\tilde{c}(x_1), \tilde{c}(x_2))$, since $u_2 \in (\overline{GC(x_1)} \wedge \overline{GC(x_2)})$. Normalization results in a region-equivalent clock valuation for u_2, too.

We have seen how a state set z in a diagonal-free TA can be projected to another state set $z' = norm^{df}(z)$. This projection is an adequate normalization technique for fully symbolic state sets in diagonal-free TA, if for all $u \in z$ it holds $u \equiv^{df} norm^{df}(u)$.

Theorem 2: Let T be a diagonal-free TA with $X = \{x_1, \ldots, x_n\}$, u a clock valuation, $x_i \in X$ an arbitrary clock variable and $v = proj^{df}_{x_i}(u)$ the projection of u over x_i. Then $u \equiv^{df} v$.

Proof: (Sketch) Let $u = (x_1^u, \ldots, x_n^u)$, $v = (x_1^v, \ldots, x_n^v)$. We distinguish 2 cases, (1) $u \in GC(x_i)$ and (2) $u \in \overline{GC(x_i)}$.

(1) It holds $v = u$ and then of course also $v \equiv^{df} u$.
(2) v is symbolically represented by $(x_i = \tilde{c}(x_i)) \exists x_i \wedge_{i=1}^{n} (x_i = x_i^u)$, i.e., $v = (x_1^u, \ldots, x_{i-1}^u, \tilde{c}(x_i), x_{i+1}^u, \ldots, x_n^u)$. As $u \in \overline{GC(x_i)}$ it follows $x_i^u > \tilde{c}(x_i) > c(x_i)$. Because of $x_i^v = \tilde{c}(x_i)$ it holds $x_i^v > c(x_i)$ as well. As the projection only affects the valuation of variable x_i, $v \equiv^{df} u$ according to Def. 5.

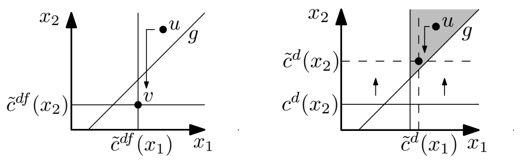
■

Since our normalization technique relies on a series of projections, which only replace clock valuations by region-equivalent ones according to Theorem 2, it follows that the normalization does not change the set of reachable locations in the TA. Moreover, normalization ensures that the number of steps needed for symbolic forward reachability analysis of TAs remains finite.

IV. Normalization for Timed Automata with Diagonal Constraints

In Sec. III we have seen a new normalization technique for diagonal-free timed automata based on the projection of sets of states. But in a timed automaton with diagonal constraints the projection using Eq. (2) can lead to false results.

Consider the clock valuation u shown in Fig. 4(a). During normalization u is projected across the diagonal constraint g resulting in a clock valuation v which is not region-equivalent to u. This can of course lead to errors during forward model checking.



(a) Diagonal constraints (b) Usage of local constraints

Fig. 4. normalization with diagonal constraints

A similar observation can be made for a slightly modified version of an example from [16] shown in Fig. 5. In the location *loop* we have a similar situation as in the TA shown in Fig. 2 with a clock x_3 rising ad infinitum, preventing a forward model checking algorithm to reach a fixed point and terminate. Additionally this TA has diagonal constraints on the transition $loop \to end$. During forward traversal the location *loop* is reached with a clock zone $z = (x_1 > 2) \wedge (x_3 \leq 3) \wedge (x_2 \leq x_3 < x_1) \wedge (x_1 - x_2 > 2)$. From the two diagonal constraints $x_1 - x_3 < 1$ and $x_3 - x_2 < 1$ on the transition $loop \to end$ the condition $x_1 - x_2 < 2$ can be inferred, such that this transition is disabled for zone z. During a continuous step the values of x_1 and x_2 evolve but the difference $x_1 - x_2$ remains the same such that the transition $loop \to end$ remains disabled. But normalizing the clock zone z using the technique presented in Sec. III will do a projection across the diagonal constraint and enable the transition. The location *end* will become reachable by mistake.

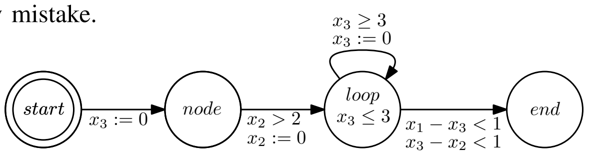


Fig. 5. Timed automaton with diagonal constraints

This problem also occurs during basic normalization for semi-symbolic model checkers and has been solved in [16] by an extended normalization algorithm which respects equivalence classes generated by difference constraints. In this section we will show how our normalization technique can be adapted to deal with diagonal constraints.

In a first step we consider diagonal constraints which go through a state set and divide it into two. Two clock valuations on different sides of a diagonal constraint are not region-equivalent according to Def. 7 and can not be projected to the same clock valuation. Let T be a TA with diagonal constraints, X the set of clock variables, $\mathcal{C}^d_T(X) = \{g_1, \dots g_m\}$ the set of diagonal constraints in T and z the state set which has to be normalized. Then for every diagonal constraint $g_i \in \mathcal{C}^d_T(X)$ with $z \wedge g_i \neq \emptyset$ and $z \wedge \overline{g_i} \neq \emptyset$, z is split along g_i, resulting in two state sets $z_1 = z \wedge g_i$ and $z_2 = z \wedge \overline{g_i}$. This splitting is done for all $g_i \in \mathcal{C}^d_T(X)$. The resulting state sets are no longer divided by any diagonal constraint and can be normalized separately. This preprocessing step is also done in the normalization algorithms for semi-symbolic model checkers [16].

As already shown by the example from Fig. 4(a), it does not help only to split state sets according to diagonal constraints and to project all clock values outside the area $\tilde{a} = \bigwedge_{i=1}^{n} GC(x_i)$ onto the border of $\tilde{a}$. Nevertheless, we will derive a similar projection method here, which will just use another area $\tilde{e} = \bigwedge_{i=1}^{n} LC(x_i)$ for projection. ($LC(x_i)$ are boundary constraints and are called 'local constraints'.). The question which area has to be used for projection is decided by the information which diagonal constraints $g_i \in \mathcal{C}^d_T(X)$ are satisfied in the state set z to be projected. After our preprocessing step we can assume that for all $g_i \in \mathcal{C}^d_T(X)$ either g_i is satisfied for all states in z or not satisfied for all states in z. W.l.o.g. we assume in the following that all $g_1, \dots, g_m$ are satisfied for the states in z.

In order to compute the local constraints, we consider the clock region $R = (\bigwedge_{i=1}^{m} g_i) \wedge \bigwedge_{i=1}^{n} (x_i > c^d(x_i))$ at first. For the time being, we only ensure that a certain subset of the states which lie both in R and z are projected to a region-equivalent state. Other states will be considered later in Theorem 3.

In principle, it would be sufficient to fix an *arbitrary clock valuation* $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$ inside the clock region $R = (\bigwedge_{i=1}^{m} g_i) \wedge \bigwedge_{i=1}^{n} (x_i > c^d(x_i))$ and project all clock valuations of states in $z \wedge R$ to $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$. By definition $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$ is region-equivalent to all clock valuations of states in $z \wedge R$. In order to determine $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$, we proceed as follows: We consider the clock region R (which is a special convex zone) and compute the closure of R [15]. Closure computation, which can be reduced to a shortest path problem, provides all constraints $x_i > b_i$ or $x_i \geq b_i$ which define the exact boundaries of R. Now we have to increase the values b_i by small values $\epsilon_i \in \mathbb{R}^+$ to ensure that $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$ with $\tilde{c}^d(x_i) = b_i + \epsilon_i$ lies inside R. [3]

We define $LC(x_i) := (x_i \leq \tilde{c}^d(x_i))$. As in Sect. III we obtain a projection (now applied with $\tilde{e} = \bigwedge_{i=1}^{n} LC(x_i)$ instead of $\tilde{a} = \bigwedge_{i=1}^{n} GC(x_i)$):

$$proj^d_{x_i}(z) := (z \wedge LC(x_i)) \vee \Big((x_i = \tilde{c}^d(x_i)) \, \exists x_i \left(z \wedge \overline{LC(x_i)} \right) \Big) \quad (5)$$

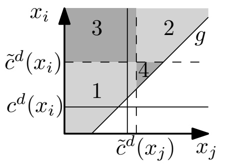


Fig. 6. Four cases

As in Sect. III the normalization $norm^d(z)$ of the state set z is a consecutive projection for all clock variables. Note that $norm^d(z)$ depends on the diagonal constraints satisfied in z (which we assumed to be $g_1, \dots g_m$ w.l.o.g.).

All clock valuations which are inside the subset $\tilde{R} = (\bigwedge_{i=1}^{m} g_i) \wedge \bigwedge_{i=1}^{n} \overline{LC(x_i)}$ of R are projected onto the region-equivalent valuation $(\tilde{c}^d(x_1), \dots, \tilde{c}^d(x_n))$.

As an example consider Fig. 4(b) where the grey shaded triangular area containing u represents the clock region R. The computed local constraints $LC(x_1)$ and $LC(x_2)$ are represented with the dashed lines and we see that the projection of u using Eq. (5) results in an region-equivalent clock valuation.

It remains to show that the projection defined by Eq. (5) also projects to region-equivalent clock valuations for clock valuations which are *outside* $\tilde{R}$ (but still satisfy $\bigwedge_{i=1}^{m} g_i$). Fortunately we are able to prove the following theorem:

Theorem 3: Let T be a TA with diagonal constraints and n clock variables $X := \{x_1, \dots, x_n\}$. Let z be an arbitrary state set satisfying $g_1, \dots, g_m$, let $norm^d$ be the normalization for the case that $g_1, \dots, g_m$ are satisfied. Let $u \in z$ be a clock

[3]The computation of ϵ_i is based on the fact that the diagonal constraints have a fixed slope. It may be performed efficiently using the computation of strongly connected components. Details are is omitted due to lack of space.

valuation, and $v := norm^d(u)$ the normalization of u. Then $u \equiv^d v$.

Proof: (Sketch) The proof is similar to the proof of Theorem 2. The only additional fact we have to prove is that v satisfies $g_1, \ldots, g_m$ as u does. Let $u = (x_1^u, \ldots, x_n^u)$, $v = (x_1^v, \ldots, x_n^v)$. Consider a fixed diagonal constraint $g_k = (x_j - x_i \prec d)$, $\prec \in \{<, \leq\}$. Then we have to distinguish four cases as shown in Fig. 6:

1) Consider the case that $u \in LC(x_j) \wedge LC(x_i)$. In this case the projection does not change x_j^u and x_i^u, i.e., v satisfies g_k as well.
2) Consider the case that $u \notin LC(x_j)$ and $u \notin LC(x_i)$. Then $x_j^v = \tilde{c}^d(x_j)$ and $x_i^v = \tilde{c}^d(x_i)$. Since $(\tilde{c}^d(x_1), \ldots, \tilde{c}^d(x_n))$ satisfies $g_1, \ldots, g_m$ by construction, it holds $\tilde{c}^d(x_j) - \tilde{c}^d(x_i) \prec d$, i.e., v satisfies g_k.
3) Consider the case that $u \in LC(x_j)$ and $u \notin LC(x_i)$. Then $x_j^v = x_j^u$ and $x_i^v = \tilde{c}^d(x_i)$. Since $\tilde{c}^d(x_j) - \tilde{c}^d(x_i) \prec d$ (see previous case) and $x_j^u \leq \tilde{c}^d(x_j)$ ($u \in LC(x_j)$), it follows $x_j^u - \tilde{c}^d(x_i) \prec d$, i.e., v satisfies g_k.
4) Consider the case that $u \notin LC(x_j)$ and $u \in LC(x_i)$. Then $x_j^v = \tilde{c}^d(x_j)$ and $x_i^v = x_i^u$. Since $x_j^u - x_i^u \prec d$ (u satisfies g_k) and $x_j^u > \tilde{c}^d(x_j)$ ($u \notin LC(x_j)$), it follows $\tilde{c}^d(x_j) - x_i^u \prec d$, i.e., v satisfies g_k.

■

With Theorem 3 we have shown that we can normalize an arbitrary state set z to a region-equivalent state set z'. As in the case of diagonal-free TAs, normalization ensures that the number of steps needed for symbolic forward reachability analysis remains finite.

V. Experiments

We have implemented a prototype version of our normalization technique and integrated it into the FSMT model checker [10] extended by a forward traversal. For first experimental results we used the TAs from Figs. 2 and 5.

At first we consider the TA shown in Fig. 2 and check whether it is possible to reach the location *end* with $x_1 = x_2$. It turns out that this state set is not reachable. Without normalization after 84 continuous steps and 83 discrete steps the computation reaches a timeout of 7200 seconds. After activating our normalization technique, executed after each continuous step, the model checking algorithm detects unreachability after 7 continuous steps, 6 discrete steps and 3,2 seconds. The example shows that normalization is needed during forward model checking and that our technique can be used for fully symbolic model checking.

In the TA with diagonal constraints shown in Fig. 5, the transition $loop \rightarrow loop$ prevents our prototype FSMT model checking algorithm to reach a fixed point during forward traversal without normalization. The algorithm runs into a timeout of 7200 seconds after 67 continuous steps and 67 discrete steps. We have seen that the diagonal constraints on the transition $loop \rightarrow end$ can provoke an error when diagonal-free normalization is used. Using the normalization technique presented in Sec. III our model checking algorithm reaches the location *end* by mistake after 3 continuous steps, 3 discrete steps and 2,19 seconds. With the normalization technique for TAs with diagonal constraints the forward model checking algorithm reaches a fixed point after 5 continuous steps, 4 discrete steps and 7,9 seconds. This shows that using diagonal-free normalization in TAs with diagonal constraints can lead to false results and that our refined technique can be used for TAs with diagonal constraints.

VI. Conclusions

We have presented a normalization technique to guarantee termination of fully symbolic forward model checking for TAs. Our method is based on projection of region-equivalent clock zones with regard to maximum constants appearing in clock constraints. We have given two versions of our normalization method, one for diagonal-free timed automata and a second to handle diagonal constraints, which makes usage of a shortest path algorithm to compute a suitable area where a clock valuation can be projected to. We have proven that a clock valuation and its normalized counterpart are region-equivalent. In first experimental results we have shown that normalization is needed for a forward traversal to reach a fixed point and that our method is suited for fully symbolic model checking.

References

[1] R. Alur, "Timed automata," in *CAV*, ser. LNCS, N. Halbwachs and D. Peled, Eds., vol. 1633. Springer, 1999, pp. 8–22.

[2] R. Alur and D. L. Dill, "A theory of timed automata," *Theor. Comput. Sci.*, vol. 126, no. 2, pp. 183–235, 1994.

[3] K. G. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *STTT*, vol. 1, no. 1-2, pp. 134–152, 1997.

[4] G. Behrmann, A. David, and K. G. Larsen, "A tutorial on Uppaal," in *SFM*, ser. LNCS, M. Bernardo and F. Corradini, Eds., vol. 3185. Springer, 2004, pp. 200–236.

[5] K. G. Larsen, J. Pearson, C. Weise, and W. Yi, "Clock difference diagrams," *Nordic J. of Computing*, vol. 6, pp. 271–298, 1999.

[6] F. Wang, "Efficient verification of timed automata with BDD-like data structures," *Int. J. Softw. Tools Technol. Transf.*, vol. 6, pp. 77–97, 2004.

[7] J. Møller, J. Lichtenberg, H. Andersen, and H. Hulgaard, "Difference decision diagrams," in *CSL*, ser. LNCS, J. Flum and M. Rodriguez-Artalejo, Eds. Springer Berlin / Heidelberg, 1999, vol. 1683, pp. 826–826.

[8] S. A. Seshia and R. E. Bryant, "Unbounded, fully symbolic model checking of timed automata using boolean methods," in *CAV*, ser. LNCS, W. A. Hunt and F. Somenzi, Eds., vol. 2725. Springer, 2003, pp. 154–166.

[9] R. Ehlers, D. Fass, M. Gerke, and H.-J. Peter, "Fully symbolic timed model checking using constraint matrix diagrams," in *RTSS*, 2010, pp. 360 –371.

[10] G. Morbé, F. Pigorsch, and C. Scholl, "Fully symbolic model checking for timed automata," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 616–632.

[11] W. Damm, S. Disch, H. Hungar, S. Jacobs, J. Pang, F. Pigorsch, C. Scholl, U. Waldmann, and B. Wirtz, "Exact state set representations in the verification of linear hybrid systems with large discrete state space," in *ATVA*, ser. LNCS, K. S. Namjoshi, T. Yoneda, T. Higashino, and Y. Okamura, Eds., vol. 4762. Heidelberg: Springer, 2007, pp. 425–440.

[12] C. Scholl, S. Disch, F. Pigorsch, and S. Kupferschmid, "Computing optimized representations for non-convex polyhedra by detection and removal of redundant linear constraints," in *TACAS*, ser. LNCS, S. Kowalewski and A. Philippou, Eds. Springer Berlin / Heidelberg, 2009, vol. 5505, pp. 383–397.

[13] W. Damm, H. Dierks, S. Disch, W. Hagemann, F. Pigorsch, C. Scholl, U. Waldmann, and B. Wirtz, "Exact and fully symbolic verification of linear hybrid automata with large discrete state spaces." *Sci. Comput. Program.*, vol. 77, no. 10-11, pp. 1122–1150, 2012.

[14] R. Loos and V. Weispfenning, "Applying linear quantifier elimination," *Comput. J.*, vol. 36, no. 5, pp. 450–462, 1993.

[15] J. Bengtsson and W. Yi, "Timed automata: Semantics, algorithms and tools," in *In Lecture Notes on Concurrency and Petri Nets*, ser. Lecture Notes in Computer Science vol 3098, W. Reisig and G. Rozenberg, Eds. Springer–Verlag, 2004.

[16] ——, "On clock difference constraints and termination in reachability analysis of timed automata," in *Proc. of ICFEM'03*, ser. Lecture Notes in Computer Science, J. S. Dong and J. Woodcock, Eds., no. 2885. Springer–Verlag, 2003.

[17] E. Clarke, O. Grumberg, and D. Peled, *Model Checking*. MIT Press, 1999.