Symbolic Simulation-Checking of Dense-Time Automata ⋆

Farn Wang

Dept. of Electrical Engineering, National Taiwan University
1, Sec. 4, Roosevelt Rd., Taipei, Taiwan 106, ROC;
+886-2-33663602; FAX:+886-2-23671909;
farn@cc.ee.ntu.edu.tw; http://cc.ee.ntu.edu.tw/˜farn

Library **RED** 7.0 is available at http://cc.ee.ntu.edu.tw/˜farn/red

Abstract. A *description automaton is simulated by* a *specification automaton* if every externally observable transition by the description can also be matched by the specification. In previous work, the simulation-checking of *timed automata (TA)* has been realized with theorem-proving, or region graph construction, or linear-hybrid system analysis. In this work, we establish a new formulation of simulation from a description TA to a specification TA. The new formulation allows us to develop a simulation-checking algorithm, in greatest fixpoint style, with zones. However, the algorithm may need many iterations to remove some state-pairs from the fixpoint images. We then present a technique to construct an under-approximation of the set of state-pairs to be removed in a fixpoint iteration. The technique does not sacrifice the precision of our algorithm and could enhance the performance of simulation-checking. Finally, we have endeavored to implement the algorithms and report its performance in the experiment.

Keywords: simulation, implementation, refinement, equivalence, bisimulation, bisimilarity, dense-time, real-time, embedded, model-checking, timed automaton, automata verification, events

1 Introduction

In the last two decades, we have witnessed significant progress in both theory and applications of the formal verification of dense-time systems. Especially, the technology of dense-time system *model-checking* [1] has been well-received by the academia, realized with many tools [7, 16, 19], and used for the verification of several industrial projects [5]. With model-checking, we represent the *description* as an *automaton* and the *specification* as a temporal logic formula and want to check whether the description satisfies the specification. However, in many applications, engineers may instead envision their *specifications* as automata (state-transition systems). There are two approaches for checking description automata against specification ones. The first is the *language inclusion problem* which checks if all runs of a description are also those of the specification. It was proved in [3] that when both the description and specification are represented as *timed*

⋆ The work is partially supported by NSC, Taiwan, ROC under grants NSC 94-2213-E-002-085 and NSC 94-2213-E-002-092. A complete version of the manuscript with all lemma proofs has been archived by ACM Computing Research Repository (CoRR) with PaperID: cs.LO/0610085 on Oct. 14, 2006.

automata (TA) [3], the language inclusion problem is undecidable. The second is called the *simulation-checking problem* that checks if every externally observable transition that can be made by the description at an instant can also be matched by the specification at the same instant. It was proved that the simulation-checking problem of TAs is in EXPTIME [15]. However, to our knowledge, so far, there has not been an efficient tool that algorithmically checks the simulation between two TAs. In contrast, the model-checking problem of TAs can be efficiently solved with *zones*[1] [7, 12, 16, 19]. In this work, we present a new formulation of simulation between two TAs which enables us to design a simulation-checking algorithm with zones.

For convenience, we denote the description automaton as A_1 and the specification as A_2. Given a state μ of A_1, a state ν of A_2, and a $\delta \in \mathbb{R}^{\geq 0}$ (the set of non-negative reals), if A_1 can make a transition at δ time units from μ and A_2 cannot match the transition at δ time units from ν, then δ is called a *refuting time* from μ to ν. Intuitively, a *simulation* is a relation between a state μ of A_1 and a state ν of A_2 such that there is no refuting time from μ to ν. Traditionally, people construct a simulation by starting from an initial image of the simulation and using the *greatest fixpoint* algorithm to iteratively remove state-pairs with a refuting time from the image. The iterations stop when we can no longer remove any state-pairs from the image. The final image is a simulation. We let $C_{(A_1,A_2)}$ be the biggest timing constant used in either A_1 or A_2. Our observation is that in the traditional formulation of simulation between TAs [4, 13, 15], in one iteration, we allow for the removal of state-pairs with any refuting time. But zones do not support the precise recording of any state-pair with unbounded refuting times. With our new formulation, in each iteration, a state-pair is removed if there is a refuting time in $[0, C_{(A_1,A_2)}]$ for the pair. The removal of state-pairs with the traditional formulation can then be achieved with a sequence of removals in successive iterations with our new formulation. Based on this new formulation, we have developed our algorithm I for the symbolic simulation-checking with zones.

Algorithm I may suffer from low performance since we may need many greatest fixpoint iterations to remove some state-pairs with only large refuting times $> C_{(A_1,A_2)}$. We thus present algorithm II which uses an under-approximation technique to conservatively remove state-pairs with refuting times $> C_{(A_1,A_2)}$ in each iteration. The good thing of algorithm II is that it is not approximate but may sometimes run faster.

We have also implemented our algorithm and techniques with TCTL model-checker **RED**, version 7. In our experiment, we used two classes of benchmarks. The first class is from the literature and previous experiments. For this class, algorithms I and II exhibited comparable performance. This could imply that our new formulation actually does not sacrifice verification performance against projects in practice. The second class is constructed specifically to investigate when algorithm II may prevail. From our experiment data, we can see that algorithm II may run exponentially more efficiently than algorithm I.

[1] A *zone* is a conjunction of atomic propositions and constraints like either $x - y \leq c$ or $x - y < c$ where x, y are either zero or clocks and c is an integer constant.

We have the following presentation plan. Section 2 is for related work. Section 3 presents the definition of TAs. Section 4 presents the traditional formulation of simulation between two TAs. Section 5 establishes our new formulation for simulation between TAs and presents algorithm I. Section 6 presents algorithm II. Section 7 reports our implementation and experiment. Section 8 is the conclusion. Due to page-limit, we have moved all proofs to the appendices.

2 Related work

In the literature, people say "*is simulated by*," "*implements*," and "*refines*" interchangeably. "*Equivalence*" means "*bisimulation*." Cerans showed that the bisimulation-checking problem of timed processes is decidable [8]. Taşıran et al showed that the simulation-checking problem of TAs is in EXPTIME [15]. They also proposed an algorithm to check whether a location homomorphism between a description TA and a specification TA preserves timed behaviors. However, there is no general strategy to efficiently construct such homomorphisms. In comparison, our approach is purely algorithmic and automatic.

Henzinger et al presented an algorithm that computes the time-abstract simulation that does not preserve timed properties [11].

Nakata also discussed how to do symbolic bisimulation checking with integer-time labeled transition systems [14]. Beyer has implemented a refinement-checking algorithm for TAs with integer-time semantics [6]. In comparison, our algorithm is for dense-time semantics.

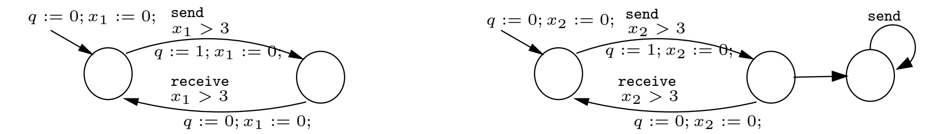
Lin and Wang presented a sound proof system for the equivalence of TAs with dense-time semantics [13]. Usually, the proofs may need human guidance.

Cleaveland and Steffen proposed to convert the equivalence-checking problem to the model-checking problem [9]. Aceto et al discussed how to construct such a modal logic formula that completely characterizes a TA [4]. However, the formula they constructed is not for TAs with timed invariance constraints. In comparison, our algorithms can handle TAs with timed invariance constraints. Moreover, their characteristic formulas fall in the realm of *linear-hybrid automata (LHA)*. In general, the model-checking problem of LHAs is undecidable [2] and is usually handled with high-complexity manipulation of convex polyhedra [2, 17].

3 Timed automata with events

Let $\mathbb{N}$ be the set of non-negative integers and $\mathbb{R}^{\geq 0}$ the set of non-negative reals. Let $\mathbb{I}_{\mathbb{N}}^{\mathbb{N}}$ be the set of all intervals with both lower-bound and upper-bound in $\mathbb{R}^{\geq 0}$. Given a set P of atomic propositions and a set X of clocks, we use $\mathbb{B}(P, X)$ as the set of all Boolean combinations of atoms of the forms p and $x \sim c$, where $p \in P$, $x \in X \cup \{0\}$, '$\sim$' is one of $\leq, <, =, >, \geq$, and c is an integer constant. An element in $\mathbb{B}(P, X)$ is called a *state predicate*.

A *valuation* of a set Y *(domain)* is a mapping from Y to a *codomain*. A *partial valuation* may be undefined for some elements in the domain. If a valuation ν is undefined for y, we denote



(a) A_1 with $H_1 : (q = 0 \wedge x_1 < 5) \vee (q = 1 \wedge x_1 < 10)$ (b) A_2 with $H_2 : (q = 0 \wedge x_2 < 5) \vee (q = 1 \wedge x_2 < 5)$

Fig. 1. Two example TAs

$\nu(y) = \circledcirc$. When it is not said specifically, a valuation means a *total valuation* that assigns a value to every element in the domain. Given two (partial) valuations ν and ν' on domain Y, $\nu\nu'$ is the composition of ν and ν' defined in the following way: for all $y \in Y$, if $\nu'(y)$ is defined, $\nu\nu'(y) = \nu'(y)$; otherwise $\nu\nu'(y) = \nu(y)$. Given a variable x and a value v in the domain of x, we use $[x \leftarrow v]$ as the partial mapping that maps x to v and everything else to $\circledcirc$. We use the following extension of TA [3] for the modeling language in this work.

Definition 1. **Timed automaton with events (TA)** A *timed automaton with events (TA)* A is a tuple $\langle \Sigma, X, G, L, I, H, E, \epsilon, \tau, \pi \rangle$ with the following restrictions. Σ is a finite set of event names. X is a finite set of clocks. G is a finite set of global atomic propositions. G and Σ represent the sets of external observables of a TA. L is a finite set of local atomic propositions such that $G \cap L = \emptyset$. $I \in \mathbb{B}(G \cup L, X)$ is the initial condition. $H \in \mathbb{B}(G \cup L, X)$ is the invariance condition. E is a finite set of transition rules. $\epsilon : E \mapsto 2^{\Sigma}$ labels each rule with a set of events. $\tau : E \mapsto \mathbb{B}(G \cup L, X)$ defines the triggering condition of each rule execution. For each $e \in E$, $\pi(e)$ is a partial valuation from X to $\{0\}$ and from $G \cup L$ to $\{\mathit{true}, \mathit{false}\}$ that defines the assignments to clocks and proposition variables of each rule execution. If $\pi(e)(y)$ is undefined, it means variable y stays unchanged in the transition. For convenience, we assume that for every TA, there is a null transition $\perp$ such that $\epsilon(\perp) = \emptyset$, $\tau(\perp) = \mathit{true}$, and $\pi(\perp)$ is undefined on everything. ■

Example 1. We have the transition diagrams of two example TAs in figure 1. They share events `send` and `receive` and global proposition q. They respectively have local clocks x_1 and x_2. A_1 has two transitions while A_2 has four. We stack the events, triggering conditions, and the assignments made at each transition. The initial conditions are labeled by the arcs without a source. ■

Definition 2. **States of a TA** A *state* of a TA A is a total valuation from X to $\mathbb{R}^{\geq 0}$ and $G \cup L$ to $\{\mathit{true}, \mathit{false}\}$. Let V_A denote the set of states of A. For any state ν and $\delta \in \mathbb{R}^{\geq 0}$, $\nu + \delta$ is a valuation identical to ν except that for every $x \in X$, $(\nu + \delta)(x) = \nu(x) + \delta$. ■

A valuation ν *satisfies* a state-predicate η, in symbols $\nu \models \eta$, if the state-predicate evaluates to true when all its variables are interpreted according to ν. Given two states ν, ν' and a transition e of a TA A, we say A *transits with* e from ν to ν', in symbols $\nu \xrightarrow{e} \nu'$, if $\nu \models \tau(e)$, $\nu\pi(e) = \nu'$, and $\nu' \models H$. Based on the above-presented notions, we now define linear computations of TAs.

Definition 3. **Runs** Given a TA $A = \langle \Sigma, X, G, L, I, H, E, \epsilon, \tau, \pi \rangle$, a *run* of A is an infinite sequence of state-time pairs $(\nu_0, t_0)(\nu_1, t_1) \ldots (\nu_k, t_k) \ldots\ldots$ with the following restrictions.

- $t_0 t_1 \ldots t_k \ldots\ldots$ is a monotonically non-decreasing real-number sequence.
- For all $k \geq 0$, for all $\delta \in [0, t_{k+1} - t_k]$, $\nu_k + \delta \models H$.
- For all $k \geq 0$, there is an $e \in E$ such that $\nu_k + t_{k+1} - t_k \xrightarrow{e} \nu_{k+1}$.

A *run segment* is a finite prefix of a run. Given a transition set $E' \subseteq E$, an E'-segment is a run segment $(\nu_0, t_0)(\nu_1, t_1) \ldots (\nu_n, t_n)$ such that for each $k \in [0, n)$, there is an $e \in E'$ with $\nu_k + t_{k+1} - t_k \xrightarrow{e} \nu_{k+1}$. ■

4 Simulation

Suppose we are given two TAs A_1 and A_2 with $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$. Intuitively, A_1 *is simulated by* A_2 if A_2 can always direct its runs so that no difference in external behaviors of A_1 and A_2 will ever be observed. We need the following notations for the convenience of discussion. Two transitions $e_1 \in E_1$ and $e_2 \in E_2$ are *compatible* if $\epsilon_1(e_1) = \epsilon_2(e_2)$ and $\forall p \in G(\pi_1(p) = \pi_2(p))$. Given an $e_1 \in E_1$, we let $E_2^{(e_1)} = \{e_2 \mid e_2 \in E_2; e_2 \text{ is compatible with } e_1.\}$.

An $E_2^{(\perp)}$-segment $(\nu_0, t_0) \ldots (\nu_k, t_k)$ of A_2 is called a *pre-matching segment* for μ, e_1, δ, and binary relation $Q \subseteq V_{A_1} \times V_{A_2}$ if and only if the following conditions are true.

- $t_k - t_0 = \delta$ and $(\mu + t_k - t_0, \nu) \in Q$.
- For every $h \in [0, k)$ and $t' \in [0, t_{h+1} - t_h]$, $(\mu + t_h - t_0 + t', \nu_h + t') \in Q$.

Definition 4. **<u>Simulation</u>** Suppose we are given two TAs A_1 and A_2 such that $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$, $L_1 \cap L_2 = \emptyset$, and $X_1 \cap X_2 = \emptyset$. A *simulation* from A_1 to A_2 is a binary relation $Q \subseteq V_{A_1} \times V_{A_2}$ with the following restrictions on all its element (μ_0, ν_0).

- μ_0 and ν_0 agree on valuation of variables in G.
- For every $\delta \in \mathbb{R}^{\geq 0}$ and transition e_1 of E_1, if $\mu_0 + \delta \models \tau_1(e_1)$, $(\mu_0 + \delta)\pi_1(e_1) \models H_1$, and $\forall \hat{\delta} \in [0, \delta](\mu_0 + \hat{\delta} \models H_1)$ are all true, then there are an $e_2 \in E_2^{(e_1)}$ and a pre-matching segment $(\nu_0, t_0) \ldots (\nu_n, t_n)$ of A_2, for μ, e_1, δ, and Q such that $\nu_n \models \tau_2(e_2)$, $\nu_n \pi_2(e_2) \models H_2$, and $((\mu_0 + t_n - t_0)\pi_1(e_1), \nu_n \pi_2(e_2)) \in Q$.

Given a simulation Q from A_1 to A_2, we denote $A_1 \preceq_Q A_2$ if for every state $\mu \models I_1$, there is a $\nu \models I_2$ with $(\mu, \nu) \in Q$. If $\exists Q(A_1 \preceq_Q A_2)$, we say A_1 *is simulated by* A_2, in symbols $A_1 \preceq A_2$. ■

Note that we require a pair (μ, ν) in a simulation must agree on the valuations of variables in G. Thus for all $g \in G$, $\mu(g) = \nu(g) = \mu\nu(g)$. With this observation, from now on, we represent (μ, ν) as valuation $\mu\nu$. With this representation, we can conveniently reason with constraints on pairs in a simulation.

Also note that simulation is defined with universal restrictions on all state-pairs in the simulation. Thus an empty simulation is still a valid simulation. There can be many simulations between two TAs. In general, we are interested at the *maximal* simulation[2] which properly contains other simulations. From now on, we assume that the simulations in discussion are always maximal.

[2] We assume that for every state-pair $\mu\nu$ in the maximal simulation, $\mu \models H_1$ and $\nu \models H_2$.

Example 2. In example 1, A_1 is not simulated by A_2 since A_1 can make a transition after 5 time units in state $q = 1$ while A_2 cannot. ■

In the following, we discuss the traditional formulation of simulation between TAs. Given an $e_1 \in E_1$ and a $\mu \in V_{A_1}$, we let $play^{\delta}_{e_1}(\mu)$ denote the condition that A_1 can do transition e_1 at time δ from state μ. Formally speaking,

$$play^{\delta}_{e_1}(\mu) \stackrel{\text{def}}{=} \mu + \delta \models \tau_1(e_1) \wedge (\mu + \delta)\pi_1(e_1) \models H_1 \wedge \neg\exists 0 \leq \delta' \leq \delta(\mu + \delta' \models \neg H_1)$$

Given a $\mu \in V_{A_1}$, a ν, a $\nu' \in V_{A_2}$, a $\delta \in \mathbb{R}^{\geq 0}$, and a $Q \subseteq V_{A_1} \times V_{A_2}$, we let $pre\text{-}match^{\delta}_2(\mu, \nu, \nu', Q)$ denote the condition that there exists a pre-matching segment $(\nu_0, t_0) \ldots (\nu_k, t_k)$ such that $\nu_0 = \nu$, $\nu_k = \nu'$, and $t_k - t_0 = \delta$. Given an $e_1 \in E_1$, a state-pair $\mu\nu \in V_{A_1} \times V_{A_2}$, and a state-pair subset $Q \subseteq V_{A_1} \times V_{A_2}$, we let $match^{\delta}_{e_1}(\mu, \nu, Q)$ denote the condition that A_2 can match e_1 at time δ from state ν with respect to Q. Note that $match^{\delta}_{e_1}(\mu, \nu, Q)$ is a predicate on state-pairs and values of δ. Formally speaking,

$$match^{\delta}_{e_1}(\mu, \nu, Q) \stackrel{\text{def}}{=} \exists \nu' \in V_{A_2} \left(\begin{array}{c} pre\text{-}match^{\delta}_2(\mu, \nu, \nu', Q) \\ \wedge\ \exists e_2 \in E_2^{(e_1)} \left(\begin{array}{c} \nu' \models \tau_2(e_2) \wedge \nu'\pi_2(e_2) \models H_2 \\ \wedge\ ((\mu + \delta)\pi_1(e_1))(\nu'\pi_2(e_2)) \in Q \end{array} \right) \end{array} \right)$$

Note that if $play^{\delta}_{e_1}(\mu) \wedge \neg match^{\delta}_{e_1}(\mu, \nu, Q)$ is true, then δ is a *refuting time* from μ to ν in Q. Given a $D \in \mathbb{I}^{\mathbb{N}}_{\mathbb{N}}$, we let $not\_sim^{D}_{e_1}(\mu, \nu, Q) \stackrel{\text{def}}{=} \exists \delta \in D(play^{\delta}_{e_1}(\mu) \wedge \neg match^{\delta}_{e_1}(\mu, \nu, Q))$. With the notations in the above, a simulation can be concisely reformulated with the following lemma.

Lemma 1. *Suppose we are given A_1 and A_2 such that $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$, $L_1 \cap L_2 = \emptyset$, and $X_1 \cap X_2 = \emptyset$. Q is a simulation from A_1 to A_2 if and only if $Q = \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\text{not\_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu, \nu, Q))\}$.* ■

Lemma 1 implies that the we can implement a simulation-checking procedure in the traditional greatest fixpoint style. That is, we first let $Q := V_{A_1} \times V_{A_2}$; and then iteratively remove set $\{\mu\nu \mid \exists e_1 \in E_1(not\_sim^{\mathbb{R}^{\geq 0}}_{e_1}(\mu, \nu, Q))\}$ from Q until a fixpoint is reached. Each iteration reduce Q to a smaller size. When a fixpoint is reached, Q is a (maximal) simulation.

5 Algorithm I for simulation checking

We plan to present the section as follows. Subsection 5.1 first explains a new formulation of simulation between TAs. The remainder subsections focus on the implementation of the new formulation with zones. Subsection 5.2 discusses how to use zones to characterize state-pairs and δ values. Subsection 5.3 explains the basic procedures that we need. Subsection 5.4 presents a predicate that characterizes those states μ satisfying $play^{\delta}_{e_1}(\mu)$ with $\delta \in [0, C_{(A_1,A_2)}]$. Subsection 5.5 presents a predicate that characterizes those state-pairs $\mu\nu$ satisfying $match^{\delta}_{e_1}(\mu, \nu, Q)$ with $\delta \in [0, C_{(A_1,A_2)}]$. Subsection 5.6 combines the results in subsections 5.4 and 5.5 to construct a predicate for our new formulation and present our first algorithm for simulation-checking.

5.1 A new formulation for simulation

Suppose we are given an $e_1 \in E_1$, a $\delta \in \mathbb{R}^{\geq 0}$, a run $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$ of A_2, a $k \geq 0$, and a $t \in [0, t_{k+1} - t_k]$. We say $t_k - t_0 + t$ is a *disrupting time* in ρ for μ, e_1, δ, and Q if and only if $t_k - t_0 + t < \delta$ and $(\mu + t_k + t - t_0)(\nu_k + t) \notin Q$. A run with a disrupting time for μ, e_1, δ, and Q is called a *disrupted run* for μ, e_1, δ, and Q. We define $minT^{\delta}_{e_1}(\mu, \rho, Q)$ as the smallest disrupting time in ρ for μ, e_1, δ, and Q. We let $maxminT^{\delta}_{e_1}(\mu, \nu, Q)$ be the maximum value of all such $minT^{\delta}_{e_1}(\mu, \rho, Q)$ with disrupted runs ρ starting from ν. That is,

$$maxminT^{\delta}_{e_1}(\mu, \nu, Q) = \max \left\{ minT^{\delta}_{e_1}(\mu, \rho, Q) \,\middle|\, \begin{array}{l} \rho \text{ is a disrupted run from } \nu \text{ of } A_2 \\ \text{for } \mu, e_1, \delta, \text{ and } Q \end{array} \right\}.$$

If there is neither a disrupting time $< \delta$ in ρ nor a state at δ time units from the starting state of ρ to execute a transition matching e_1, then we call ρ a *futile run* for μ, e_1, δ, and Q. It is easy to see that if given a $\mu\nu \in Q$ such that $play^{\delta}_{e_1}(\mu) \wedge \neg match^{\delta}_{e_1}(\mu, \nu, Q)$ is true, then we know all runs of A_2 starting from ν must be either disrupted or futile for μ, e_1, δ, and Q. Here we have the following central lemma of the work. We only give a brief and intuitive explanation why the lemma is true. A formal proof can be found in appendix B.

Lemma 2. *Suppose we are given a $Q \subseteq V_{A_1} \times V_{A_2}$, an $e_1 \in E_1$, and a $\delta \in \mathbb{R}^{\geq 0}$. If there is a $\mu\nu \in Q$ such that $\mathrm{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathrm{match}^{\delta}_{e_1}(\mu, \nu, Q)$ is true, then for each $t \in [0, \delta]$, there is a $\bar{\mu}\bar{\nu} \in Q$ such that either $\mathrm{play}^{t}_{e_1}(\bar{\mu}) \wedge \neg \mathrm{match}^{t}_{e_1}(\bar{\mu}, \bar{\nu}, Q)$ or $\mathrm{play}^{t}_{\perp}(\bar{\mu}) \wedge \neg \mathrm{match}^{t}_{\perp}(\bar{\mu}, \bar{\nu}, Q)$ is true.*

Proof : There are two cases to analyze. In the first case, we assume that there is a futile run $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$ for μ, e_1, δ, and Q. Then there is a $k \geq 0$ and a $t' \in [0, t_{k+1} - t_k]$ such that $\delta - (t_k + t' - t_0) = t$ in ρ. Since all runs from ν are either disrupted or futile for μ, e_1, δ, and Q, we can prove that all runs from $\nu_k + t'$ are also either disrupted or futile for μ, e_1, t, and Q. Thus $\mu + \delta - t$ is a candidate for $\bar{\mu}$ while $\nu_k + t'$ is for $\bar{\nu}$. Thus this case is proven.

In the second case, we assume that all runs of A_2 from ν are disrupted for μ, e_1, δ, and Q. If $t > maxminT^{\delta}_{e_1}(\mu, \nu, Q)$, according to the defintion of simulation, $play^{t}_{\perp}(\mu) \wedge \neg match^{t}_{\perp}(\mu, \nu, Q)$ is also true. Then the lemma is proven with $\mu = \bar{\mu}$ and $\nu = \bar{\nu}$. If $t \leq minT^{\delta}_{e_1}(\mu, \rho, Q) = maxminT^{\delta}_{e_1}(\mu, \nu, Q)$, we let $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$ of A_2 be a disrupted run for μ, e_1, δ, and Q such that $\nu_0 = \nu$ and $minT^{\delta}_{e_1}(\mu, \rho, Q) = maxminT^{\delta}_{e_1}(\mu, \nu, Q)$. Let $\nu_k + t'$ with $t' \in [0, t_{k+1} - t_k]$ be the state in ρ that causes the disrupting. Since time is continuous, we can find an $h \in [0, k]$ and a $t'' \in [0, t_{h+1} - t_h]$ such that $t_h - t_0 + t'' + t \geq t_k - t_0 + t'$. Then with an argument similar to the one in the first case, we can prove the case with $\bar{\mu} = \mu + t_h - t_0 + t''$ and $\bar{\nu} = \nu_h + t''$. ■

Lemma 2 implies the following lemma for a new formulation of simulation.

Lemma 3. *Suppose we are given A_1 and A_2 such that $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$, $L_1 \cap L_2 = \emptyset$, and $X_1 \cap X_2 = \emptyset$. Q is a simulation from A_1 to A_2 if and only if $Q = \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \neg \exists e_1 \in E_1 \exists \delta \in [0, C_{(A_1, A_2)}](\mathrm{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathrm{match}^{\delta}_{e_1}(\mu, \nu, Q))\}$.* ■

5.2 Representation with zones

Given a set P of atomic propositions, a set X of clocks, and a non-negative integer constant c, a *zone-predicate* ζ of X, P, and c is constructed inductively with the following rules.

$$\zeta ::= p \mid x - y \sim d \mid \neg\zeta_1 \mid \zeta_1 \wedge \zeta_2$$

Here $p \in P$, $x, y \in X \cup \{0\}$, '$\sim$' $\in \{\leq, <, =, \neq, >, \geq\}$, and d is an integer in $[-c, c]$. Let $Z_c(P, X)$ be the set of zone-predicates that can be constructed of P, X, and c. A zone-predicate is *conjunctive* if the only Boolean operator it can have is $\wedge$. A state-space that can be characterized with a conjunctive zone-predicate is called a *zone*. A *region* of P, X, and c is a smallest zone that cannot properly contain other zones characterizable with $Z_c(P, X)$. A valuation satisfies a zone-predicate if the zone-predicate is evaluated true when all its variables are interpreted according to the valuation.

Given two state-pairs $\mu\nu$ and $\mu'\nu'$ in $V_{A_1} \times V_{A_2}$, we denote $\mu\nu \equiv \mu'\nu'$ if and only if $\mu\nu$ and $\mu'\nu'$ are in the same region in $Z_{C_{(A_1,A_2)}}(G \cup L_1 \cup L_2, X_1 \cup X_2)$. A fundamental theorem about simulation-checking with zone-technology is theorem 10 in [15]. Here we restate the theorem.

Theorem 1. *Given the maximal simulation Q from A_1 to A_2 and two state-pairs $\mu\nu \equiv \mu'\nu'$ in $V_{A_1} \times V_{A_2}$, $\mu\nu \in Q$ if and only if $\mu'\nu' \in Q$.* ■

Lemma 3 and theorem 1 motivate us to construct a representation for $not\_sim_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu, \nu, Q)\}$ with zones. Suppose we are given A_1 and A_2 such that $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$, $L_1 \cap L_2 = \emptyset$, and $X_1 \cap X_2 = \emptyset$. In the formulation of $not\_sim_{e_1}^{D}(\mu, \nu, Q)$, we use a variable δ to measure the value of refuting times. One difficulty to use zones to represent this variable δ is that its values decrement with time progress while the values of clocks increment. The increment rates of clocks are uniform and exactly the same as the decrement rate of δ. To overcome this difficulty, we introduce an auxiliary clock 'Ω^δ' for the description of constraints on δ. The value of clock Ω^δ always equals $C_{(A_1,A_2)} - \delta$. Its values are in $(-\infty, C_{(A_1,A_2)}]$ with the same increment rate as other clocks. A valuation (state-pair or state) with $\Omega^\delta = C_{(A_1,A_2)}$ can be at the moment that a transition is made by A_1 and a matching transition from A_2 is expected. In this section, we only consider those valuations γ with $\gamma(\Omega^\delta) \in [0, C_{(A_1,A_2)}]$. Clock Ω^δ is neither used nor initialized in A_1 and A_2. Given two valuations μ and ν, if $\mu(\Omega^\delta) = \nu(\Omega^\delta)$, then we know they have the same timing distance to the moment when A_1 makes transition e_1 and A_2 is expected to match. We plan to use zones in $Z_{C_{(A_1,A_2)}}(G \cup L_1 \cup L_2, X_1 \cup X_2 \cup \{\Omega^\delta\})$ to construct a representation of $not\_sim_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu, \nu, Q)$.

Example 3. Given $G = \{a\}, X_1 = \{x_1\}, L_1 = \{b_1\}, X_2 = \{x_2\}, L_2 = \{b_2\}$, we may have the following zone-predicate for a fixpoint image.

$$\mathcal{Q} \equiv \begin{array}{l} (a \wedge b_1 \wedge \neg b_2 \wedge 0 \leq x_1 \wedge 3 < x_2 \leq 5 \wedge x_2 - x_1 \leq 5 \wedge \Omega^\delta \leq 5) \\ \vee (\neg a \wedge 2 \leq x_1 < 9 \wedge 1 < x_2 \wedge x_1 - x_2 < 8 \wedge -\Omega^\delta \leq -1 \wedge \Omega^\delta \leq 2) \end{array}$$ ■

5.3 Basic procedures

In the following, we explain procedures that manipulate zone-predicates to represent state-pairs with the restrictions of $\Omega^\delta \in [0, C_{(A_1,A_2)}]$. One fundamental procedure is *Fourier-Motzkin elimination* [10]. Suppose we have a formula F of variables in set Y. In this work, Fourier-Motzkin elimination constructs $\exists y(F)$ as a formula without $y \in Y$. For zone-predicates, efficient implementation of Fourier-Motzkin elimination has been discussed in [12, 16, 17] and can be found in appendix A. For convenience, we use $\mathrm{FM\_elm}(\eta, X')$ to denote an implementation of the Fourier-Motzkin elimination that eliminates clock variables in X' from η.

Example 4. For the $\mathcal{Q}$ in example 3, $\exists a(\mathcal{Q}) \equiv (b_1 \wedge \neg b_2 \wedge 0 \leq x_1 \wedge 3 < x_2 \leq 5 \wedge x_2 - x_1 \leq 5) \vee (2 \leq x_1 < 9 \wedge 1 < x_2 \wedge x_1 - x_2 < 8)$. And $\exists x_1(\mathcal{Q}) \equiv (a \wedge b_1 \wedge \neg b_2 \wedge 3 < x_2 \leq 5) \vee (\neg a \wedge 1 < x_2 < 6)$. ■

Given two zone-predicates ζ, ζ', we need a basic procedure $\texttt{Tbck}(\zeta', \zeta)$ that computes the precondition of ζ through a time-progress through states that satisfy ζ'. According to [12], we have $\texttt{Tbck}(\zeta_1, \zeta_2) \stackrel{\text{def}}{=} \exists t\, (t \geq 0 \wedge \zeta_2 + t \wedge \neg\exists t'((\neg\zeta_1) + t' \wedge 0 \leq t' \wedge t' \leq t))$.

Given a partial assignment Π of $G \cup L_1 \cup L_2 \cup X_1 \cup X_2$, we let $\eta\Pi$ be the precondition to η before the assignment. Suppose Π is defined on $y_1, \ldots, y_n$.

$$\eta\Pi \stackrel{\text{def}}{=} (\textstyle\bigwedge_{x \text{ is a clock defined in } \Pi.}\, x \geq 0) \wedge \exists y_1 \ldots \exists y_n(\eta \wedge \textstyle\bigwedge_{1 \leq i \leq n} y_i = \Pi(y_i))$$

Given $e_1 \in E_1$ and $e_2 \in E_2^{(e_1)}$, the weakest precondition to η through discrete transition pair (e_1, e_2) can be represented as $\texttt{Xbck}_{(e_1,e_2)}(\eta) \stackrel{\text{def}}{=} \tau_1(e_1) \wedge \tau_2(e_2) \wedge (\eta(\pi_1(e_1)\pi_2(e_2)))$.

5.4 Construction of formula $\texttt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}$

The restriction of a predicate ζ to $\delta \in [0, C_{(A_1,A_2)}]$ is simply the conjunction of ζ and $-\Omega^\delta \leq 0 \wedge \Omega^\delta \leq C_{(A_1,A_2)}$. For convenience, we let

$$\texttt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]} \stackrel{\text{def}}{=} \begin{pmatrix} -\Omega^\delta \leq 0 \wedge \Omega^\delta \leq C_{(A_1,A_2)} \\ \wedge\ \texttt{Tbck}\left(H_1, \texttt{Xbck}_{(e_1,\perp)}(H_1) \wedge \Omega^\delta = C_{(A_1,A_2)}\right) \end{pmatrix}$$

Here $\texttt{Tbck}\left(H_1, \texttt{Xbck}_{(e_1,\perp)}(H_1) \wedge \Omega^\delta = C_{(A_1,A_2)}\right)$ is the standard representation of the weakest timed precondition with transition $(e_1, \perp)$ [12,16]. The weakest characterization of μ with $\mathit{play}_{e_1}^\delta(\mu)$ can be derived backward from H_1 with the following lemma.

Lemma 4. *Given a state $\mu \in V_{A_1}$ and a real number $t \in [0, C_{(A_1,A_2)}]$,* $\mathrm{play}_{e_1}^t(\mu)$ *if and only if* $\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \texttt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}$. ■

5.5 Construction of formula $\texttt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(Q)$

With procedures $\texttt{Xbck}_{(e_1,e_2)}()$, $\texttt{Tbck}()$, and symbolic characterizations for ν' and Q, we now construct the symbolic characterization of $\textit{pre-match}_2^t(\mu, \nu, \nu', Q)$, denoted $\texttt{Rbck}_{E_2^{(\perp)}}^{[0,C_{(A_1,A_2)}]}(\eta_1, \eta_2)$.

Intuitively, this is similar to the backward reachability [12,16] within $C_{(A_1,A_2)}$ time units. Again, we use clock Ω^δ to measure the length of the reachability run-segments. Computationally, $\mathtt{Rbck}^{[0,C_{(A_1,A_2)}]}_{E_2^{(\perp)}}(\eta_1,\eta_2)$ is the least fixpoint solution of equation:

$$Y = \left(\Omega^\delta = C_{(A_1,A_2)} \wedge \eta_2 \vee \mathtt{Tbck}\left(\left(\begin{array}{c}\Omega^\delta \leq C_{(A_1,A_2)}\\ \wedge\, 0 \leq \Omega^\delta \wedge \eta_1\end{array}\right), \bigvee_{e_2\in E_2^{(\perp)}} \mathtt{Xbck}_{(\perp,e_2)}(Y)\right)\right)$$

In other words, $\mathtt{Rbck}^{[0,C_{(A_1,A_2)}]}_{E_2^{(\perp)}}(\eta_1,\eta_2)$ is equal to the following formula.

$$\mathbf{lfp}Y.\left(\Omega^\delta = C_{(A_1,A_2)} \wedge \eta_2 \vee \mathtt{Tbck}\left(\left(\begin{array}{c}\Omega^\delta \leq C_{(A_1,A_2)}\\ \wedge 0 \leq \Omega^\delta \wedge \eta_1\end{array}\right), \bigvee_{e_2\in E_2^{(\perp)}} \mathtt{Xbck}_{(\perp,e_2)}(Y)\right)\right)$$

Following the standard construction of backward reachability, we have the following lemma.

Lemma 5. *Given a zone-predicate $\mathcal{Q}$ for a set $Q \subseteq V_{A_1} \times V_{A_2}$, a state-pair $\mu\nu \in Q$, a zone-predicate $f_{\nu'}$ for states $\nu' \in V_{A_2}$, and a $t \in [0, C_{(A_1,A_2)}]$,* pre-match$^t_2(\mu,\nu,\nu',Q)$ *is true if and only if $\mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Rbck}^{[0,C_{(A_1,A_2)}]}_{E_2^{(\perp)}}(\mathcal{Q}, f_{\nu'})$.* ■

Specifically, the quantified ν' is for the precondition of all transitions in E_2 that match e_1. Given a zone-predciate $\mathcal{Q}$, the zone-predicate for ν', i.e. $f_{\nu'}$, is $\bigvee_{e_2\in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$. Note that to make sure that A_1 and A_2 observe the same behavior with e_1 and e_2 respectively, we construct the precondition of both e_1 and e_2 out of $\mathcal{Q}$. Now with the formulation of $f_{\nu'_2}$, we let

$$\mathtt{Match}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q}) \stackrel{\text{def}}{=} \mathtt{Rbck}^{[0,C_{(A_1,A_2)}]}_{E_2^{(\perp)}}\left(\mathcal{Q}, \bigvee_{e_2\in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})\right).$$

We can establish the following lemma.

Lemma 6. *Given a zone-predicate $\mathcal{Q}$ for a $Q \subseteq V_{A_1} \times V_{A_2}$, a $\mu\nu \in Q$, and a $t \in [0, C_{(A_1,A_2)}]$,* match$^t_{e_1}(\mu,\nu,Q)$ *is true if and only if $\mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Match}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q})$.* ■

5.6 Algorithm I

Given a zone-predicate $\mathcal{Q}$ for a set $Q \subseteq V_{A_1} \times V_{A_2}$, we have the following shorthand.

$$\mathtt{Not\_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q}) \stackrel{\text{def}}{=} \text{FM\_elm}\left(\mathtt{Play}^{[0,C_{(A_1,A_2)}]}_{e_1} \wedge \neg\mathtt{Match}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q}), \{\Omega^\delta\}\right).$$

With lemmas 4 and 6, we can establish the following lemma.

Lemma 7. *Given a zone-predicate $\mathcal{Q}$ for a $Q \subseteq V_{A_1} \times V_{A_2}$, an $e_1 \in E_1$, and a $\mu\nu \in S$,* not_sim$^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q)$ *is true if and only if $\mu\nu \models \mathtt{Not\_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q})$.* ■

With lemma 7 and theorem 1, we are now ready to present our algorithm for simulation checking. The algorithm is a greatest-fixpoint one. We start from $\mathcal{Q} = H_1 \wedge H_2$. Then we iteratively delete state pairs in $\mathtt{Not\_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q})$ for each $e_1 \in E_1$ until a fixpoint is reached.

$\mathrm{Sim\_Check}^{[0,C_{(A_1,A_2)}]}(A_1,A_2)$ /* $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i\rangle$, $1 \leq i \leq 2$ */ {
let $\mathcal{Q} := H_1 \wedge H_2$; $\mathcal{Q}' := \mathit{false}$;
while ($\mathcal{Q} \neq \mathcal{Q}'$), do {
$\mathcal{Q}' := \mathcal{Q}$;
for each $e_1 \in E_1$,
$\mathcal{Q} := \mathcal{Q} \wedge \neg \mathtt{Not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}(Q)$; .... (A)
if ($I_1 \neq \mathrm{FM\_elm}(I_1 \wedge I_2 \wedge \mathcal{Q}, L_2 \cup X_2)$) .... (B)
print "A_1 does not implement A_2." and return *false*.
}
print "A_1 implements A_2." and return *true*.
}

In statement (B), we check whether all initial states of A_1 is paired with some initial states of A_2 in $\mathcal{Q}$. The statement employs a technique called *early decision of greatest fixpoint (EDGF)* [18] and could significantly reduce the computation time of greatest fixpoint evaluation when no simulation exists. The following lemma establishes the correctness of our algorithm.

Lemma 8. *Sim_Check*$^{[0,C_{(A_1,A_2)}]}(A_1,A_2)$ *terminates. It returns true iff* $A_1 \preceq A_2$. ■

6 Algorithm II with under-approximation of $\boldsymbol{not\_sim_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)}$

We plan to modify Algorithm I in the following direction. Assume that we have a set $Q \subseteq V_{A_1} \times V_{A_2}$ at an iteration of the greatest fixpoint loop in procedure $\mathrm{Sim\_Check}^{[0,C_{(A_1,A_2)}]}(A_1,A_2)$. If this is not the last iteration, some state-pairs will be eliminated from $\mathcal{Q}$ in this iteration. To make sure that the result greatest fixpoint is maximal, it is curcial that no state-pairs in the maximal greatest fixpont are eliminated from $\mathcal{Q}$. This implies that if we only eliminate an under-approximation of $\mathit{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)$ in each iteration, then the correctness of the greatest fixpoint calculation is maintained. Moreover, it is possible that the elimination of an under-approximation of $\mathit{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)$ could lead to faster greatest fixpoint calculation.

In this work, we propose the following under-approximation of $\mathit{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$.

$$\begin{aligned}
&\mathit{uapprox_not_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)\\
&\stackrel{\text{def}}{=} \mu\nu \in Q \wedge \forall \delta \in (C_{(A_1,A_2)},\infty)(\mathit{play}_{e_1}^{\delta}(\mu) \wedge \neg \mathit{match}_{e_1}^{\delta}(\mu,\nu,Q))\\
&\equiv \mu\nu \in Q \wedge (\forall \delta \in (C_{(A_1,A_2)},\infty)(\mathit{play}_{e_1}^{\delta}(\mu))) \wedge \neg(\exists \delta \in (C_{(A_1,A_2)},\infty)(\mathit{match}_{e_1}^{\delta}(\mu,\nu,Q)))
\end{aligned}$$

Lemma 9. *Given a set* $Q \subseteq V_{A_1} \times V_{A_2}$*, a transition* $e_1 \in E_1$*, and a state-pair* $\mu\nu \in Q$*, if* $\mathrm{uapprox\_not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$ *is true, then* $\mathrm{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$ *is true.* ■

In the following, we present techniques for the construction of characterizations for the components in this under-approximation.

6.1 $\mathtt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)$ to characterize $\forall\delta \in (C_{(A_1,A_2)},\infty)(play_{e_1}^{\delta}(\mu))$

The following lemma helps us to characterize $play_{e_1}^{\delta}(\mu)$ with zone-predicates.

Lemma 10. *Given a state $\mu \in V_{A_1}$ and a $t \in (C_{(A_1,A_2)},\infty)$, $\mathrm{play}_{e_1}^{t}(\mu)$ is true if and only if for every $t' \in (C_{(A_1,A_2)},\infty)$, $\mathrm{play}_{e_1}^{t'}(\mu)$ is true.*

Proof : Note that zone-predicates are with inequalities like $x - x' \sim d$ where $|d| \leq C_{(A_1,A_2)}$. Thus for any $t, t' \in (C_{(A_1,A_2)},\infty)$, $\mu + t \models x - x' \sim d$ if and only if $\mu + t' \models x - x' \sim d$. ■

Since $(C_{(A_1,A_2)},\infty)$ is non-empty, lemma 10 implies that for any $\mu \in V_{A_1}$, the truth values of $\forall\delta \in (C_{(A_1,A_2)},\infty)(play_{e_1}^{\delta}(\mu))$ and $\exists\delta \in (C_{(A_1,A_2)},\infty)(play_{e_1}^{\delta}(\mu))$ are the same. For convenience, we let $\mathtt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)}$ denote the following zone-predicate.

$$\mathtt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)} \stackrel{\text{def}}{=} \text{FM\_elm}\left(\left(\begin{array}{c} \Omega^{\delta} < 0 \\ \wedge\ \mathtt{Tbck}\left(H_1, \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \Omega^{\delta} = C_{(A_1,A_2)}\right) \end{array}\right), \{\Omega^{\delta}\}\right)$$

Then we have the following lemma.

Lemma 11. *Given a state $\mu \in V_{A_1}$, $\forall\delta \in (C_{(A_1,A_2)},\infty)\left(\mathrm{play}_{e_1}^{\delta}(\mu)\right)$ is true if and only if $\mu \models \mathtt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)}$.* ■

6.2 $\mathtt{OMatch}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)$ to characterize $\exists\delta \in (C_{(A_1,A_2)},\infty)(match_{e_1}^{\delta}(\mu,\nu,Q))$

There are several observations that we can make for the evaluation of $\exists\delta \in (C_{(A_1,A_2)},\infty)(match_{e_1}^{\delta}(\mu,\nu,Q))$. First, for a state-pair $\mu\nu$ such that $t > C_{(A_1,A_2)}$ and $match_{e_1}^{t}(\mu,\nu,Q)$ is true, then for all state-pairs $\mu'\nu'$ that can forwardly reach $\mu\nu$ through a pre-matching segment for μ', e_1, t', and Q, we know $t' + t \geq t > C_{(A_1,A_2)}$ also. Also according to the discussion in the previous section, for those state-pairs $\mu\nu$ with $\delta \leq C_{(A_1,A_2)}$ and $match_{e_1}^{\delta}(\mu,\nu,Q)$ true for some Q, we can use zone-predicates to characterize the relation between $\mu\nu$, δ, and Q for the calculation of backward reachability. We use the following two zone-predicate variables to record the information we need for the evaluation of $\exists\delta \in (C_{(A_1,A_2)},\infty)(match_{e_1}^{\delta}(\mu,\nu,Q))$.

- Π : for the recording of those state-pairs $\mu\nu$ and time $t \leq C_{(A_1,A_2)}$ such that there is a pre-matching segment of t time units from $\mu\nu[\Omega^{\delta} \leftarrow C_{(A_1,A_2)} - t]$ to some $\bar{\mu}\bar{\nu}[\Omega^{\delta} \leftarrow C_{(A_1,A_2)}]$ in $\Omega^{\delta} = C_{(A_1,A_2)} \wedge \bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$.
- Δ : for the recording of those state-pairs $\mu\nu$ which are backwardly reachable from a state-pair in $\Omega^{\delta} = C_{(A_1,A_2)} \wedge \bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$ through a finite run segment of length $> C_{(A_1,A_2)}$.

The precondition of state-pairs in Δ will also be characterized in Δ eventually. The precondition of state-pairs in Π may end up in either Π or Δ. Moreover, the precondition calculation of Δ does not involve variable δ and clock Ω^{δ} in this approximation. With these observations, we present the following algorithm that works on Δ and Π to calculate $\exists\delta \in (C_{(A_1,A_2)},\infty)(match_{e_1}^{\delta}(\mu,\nu,Q))$.

$\texttt{OMatch}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$ {
$\Pi := \Omega^\delta = C_{(A_1,A_2)} \wedge \bigvee_{e_2 \in E_2^{(e_1)}} \texttt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$;
$\Delta := \mathit{false}$; $\Pi' := \mathit{false}$; $\Delta' := \mathit{false}$; .... (C)
while $\Pi \neq \Pi' \vee \Delta \neq \Delta'$, do {
$\Delta' := \Delta$; $\Pi' := \Pi$;
$\Delta := \Delta \vee \texttt{Tbck}(\Delta, \texttt{Xbck}_{(e_1,\perp)}(\Delta))$; .... (D)
$\Gamma := \texttt{Tbck}(\mathcal{Q}, \texttt{Xbck}_{(e_1,\perp)}(\Pi))$; $\Pi := \Pi \vee (\Omega^\delta \geq 0 \wedge \Gamma)$; .... (E)
$\Delta := \Delta \vee \text{FM\_elm}(\Omega^\delta < 0 \wedge \Gamma, \{\Omega^\delta\})$; .... (F)
}
return Δ; .... (G)
}

Note that the only changes that we can make to Π and Δ are at statements (D), (E), and (F). Every time when we change Π, we make sure that the δ values in Π are in $[0, C_{(A_1,A_2)}]$ through conjunction with constraint $\Omega^\delta \geq 0$. Every time when we change Δ, we make sure that the δ values in Δ are in $(C_{(A_1,A_2)}, \infty)$. This is guaranteed in both statements (D) and (F). Statement (D) calculates the timed weakest precondition from all state-pairs already in Δ. Since we are using backward analysis, we find that statement (D) can only add predicates with δ values in $(C_{(A_1,A_2)}, \infty)$. Statement (F) calculates the timed weakest precondition from all state-pairs already in Π. It guarantees that all δ values added to Δ are in $(C_{(A_1,A_2)}, \infty)$ through conjunction with constraint $\Omega^\delta < 0$.

Lemma 12. *At statement (G) of procedure* $\texttt{OMatch}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$*, for all* $\mu\nu \in V_{A_1} \times V_{A_2}$*,* $\mu\nu \models \Delta$ *if and only if* $\mu\nu \models \exists\delta \in (C_{(A_1,A_2)}, \infty)(\text{match}_{e_1}^{\delta}(\mu, \nu, Q))$*.* ∎

6.3 Algorithm II with a characterization of $\mathit{uapprox\_not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu, \nu, Q)$

Given a zone-predicate $\mathcal{Q}$, we define

$$\texttt{UNot\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q}) \stackrel{\text{def}}{=} \texttt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)} \wedge \neg\texttt{OMatch}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$$

Lemma 13. *Given a zone-predicate* $\mathcal{Q}$ *for a set* $Q \subseteq V_{A_1} \times V_{A_2}$*, for any state-pair* $\mu\nu \in V_{A_1} \times V_{A_2}$*, if* $\mu\nu \models \texttt{UNot\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$*, then* $\text{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu, \nu, Q)$ *is true.* ∎

Lemma 13 implies that we may use $\texttt{UNot\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$ to speed up the fixpoint calculation without hurting the precision of the algorithm. With the lemma, we can present our algorithm II, Sim_Check(A_1, A_2), for simulation-checking. It is identical to $\text{Sim\_Check}^{[0,C_{(A_1,A_2)}]}(A_1, A_2)$ except that we substitute the following statement (A') for statement (A).

$\mathcal{Q} := \mathcal{Q} \wedge \left(\neg\texttt{Not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}(Q)\right) \wedge \left(\neg\texttt{UNot\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)\right)$; .... (A')

Due to page-limit, we do not present the whole Algorithm II here. A comment on performance enhancement is that in procedure $\texttt{OMatch}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mathcal{Q})$, at statement (E), the final value of

Π is exactly $\mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q})$. Thus we can share the evaluation of Π while evaluating $\mathtt{Not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}(Q)$ and $\mathtt{UNot\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(Q)$ in statement (A').

6.4 Complexity of our algorithm

The complexity of our algorithm relies on the implementation of the basic manipulation procedures of zones. Like in [12], we argue that we can implement the zones as sets of regions [1]. In such an implementation, basic operations like subsumption, intersection, union, complement, time progression, and variable quantification can all be done in EXPTIME.

Lemma 14. *A proper implementation of Sim_Check*(A_1, A_2) *can run in EXPTIME.* ■

7 Experiments

We have implemented the techniques discussed in this manuscript in **RED** 7.0, a model-checker for TAs and parametric safety analysis for LHAs based on CRD and HRD-technology [16, 17]. The TAs to **RED** are described as a set of communicating TAs, each representing a process. The state-spaces are explored in a symbolic on-the-fly style. We have experimented with the following three parameterized benchmarks with various numbers of processes from the literature and previous experiments for model-checking. A_1 and A_2 differ in only one process.

1. *Fischer's timed mutual exclusion algorithm* [16]: The algorithm relies on a global lock and a local clock per process to control access to the critical section. Two timing constants used are 10 and 19.
2. *CSMA/CD* [19]: This is the Ethernet bus arbitration protocol with collision-and-retry. The timing constants used are 26, 52, and 808.
3. *Timed consumer/producer*: There is a buffer, some producers, and some consumers. The producers periodically write data to the buffer if it is empty. The consumers periodically wipe out data, if any, in the buffer. The timing constants used are 5, 10, 15, and 20.

For each benchmark, two versions are used, one with a simulation relation and one without. The performance data is reported in table 1. The CPU time used, the total memory consumption for the data-structures in state-space representations, and the total number of fixpoint iterations are reported. As can be seen, the performance of algorithms I and II are roughly the same. Especially, algorithm II did not use fewer iterations to calculate the fixpoints. This could implies that our new formulation of simulation between TAs might actually not hurt the performance in practice.

We have also specifically designed the following fourth benchmark to investigate in what situations, the speed-up technique of algorithm II may prevail.

4. *Periodical samplers*: Two sensors periodically send in sampling signals with periods p_1 and p_2 respectively. When the sampling signals from the two processes happen at the same instant, the operating system needs to do some extra work and may disrupt the service.

Table 1. Performance data of scalability w.r.t. various strategies

			Algorithm I			Algorithm II		
benchmarks	versions	m	time	memory	iter'n	time	memory	iter'n
Fischer's mutual exclusion (m processes)	Simulation exists.	4	31.6s	347k	8	31.3s	320k	8
		5	98.3s	731k	8	92.3s	664k	8
		6	304s	1518k	8	281s	1319k	8
	No simulation exists.	4	9.87s	251k	3	11.7s	250k	3
		5	31.5s	475k	3	28.0s	475k	3
		6	86.0s	957k	3	86.7s	955k	3
CSMA/CD (1 bus+ m senders)	Simulation exists.	1	0.099s	41k	2	0.098s	41k	2
		2	0.80s	177k	2	0.80s	177k	2
		3	122s	3509k	7	125s	3503k	7
	No simulation exists.	1	0.103s	41k	2	0.085s	41k	2
		2	1.50s	202k	2	2.03s	203k	2
		3	21.7s	2089k	2	25.7s	2089k	2
Consumer & producer (1 buffer +1 producer +m consumers)	Simulation exists.	3	0.26s	57k	2	0.30s	57k	2
		4	0.79s	65k	2	0.43s	65k	2
		5	1.21s	76k	2	0.53s	75k	2
	No simulation exists.	3	0.65s	70k	4	0.99s	70k	4
		4	0.93s	77k	4	1.35s	775k	4
		5	1.17s	83k	4	1.16s	83k	4
Periodical sampler (1 process with periods p_1 and p_2)	Simulation exists.	11/13	22.0s	4670k	14	5.86s	2451k	3
		13/17	56.9s	5989k	16	12.7s	3192k	3
		17/19	125s	17846k	20	22.4s	8444k	3
	No simulation exists.	11/13	21.1s	4664k	12	5.43s	2452k	1
		13/17	55.2s	5978k	14	13.0s	3194k	1
		17/19	125s	17842k	18	20.5s	8448k	1

data collected on a Pentium 4 1.7GHz with 380MB memory running LINUX;
s: seconds; k: kilobytes of memory in data-structure; iter'n: the number of iterations

The benchmark was designed so that the initial state-pairs can only be rejected through a prematching segment of $p_1 \cdot p_2$ time units. As can be seen from the table, for the version with a simulation, algorithm II always uses only 3 iterations to reach the fixpoints. On the other hand, the numbers of iterations of algorithm I roughly grow with the values of the periods. The same pattern can be observed in the data for the version without a simulation. As a result, algorithm I runs exponentially slower than algorithm II.

8 Concluding remarks

In this work, we discuss how to implement a symbolic simulation-checker for TAs with zones. Our implementation and experiment shows the promise that our algorithm could be useful in practice in the future. With a straightforward adaptation, our techniques can also be used to check bisimulation (equivalence) between TAs.

References

1. R. Alur, C. Courcoubetis, D.L. Dill. Model Checking for Real-Time Systems, IEEE LICS, 1990.
2. R. Alur, C.Courcoubetis, T.A. Henzinger, P.-H. Ho. Hybrid Automata: an Algorithmic Approach to the Specification and Verification of Hybrid Systems. in Proceedings of Workshop on Theory of Hybrid Systems, LNCS 736, Springer-Verlag, 1993.
3. R. Alur, D.L. Dill. A Theory of Timed Automata. Theoretical Computer Science, 126:183-235, 1994.
4. L. Aceto, A. Ingólfsdóttir, M.L. Pedersen, J. Poulsen. Characteristic Formulae for Timed Automata. Theoretical Informatics and Applications, 2000, Vol. 34, No.6, pp.565-584. EDP Sciences.
5. J. Bengtsson, W.O.D. Griffioen, K.J. Kristoffersen, K.G. Larsen, F. Larsson, P. Pettersson, Wang Y. Verification of an Audio Protocol with Bus Collision Using UPPAAL. CAV 1996, LNCS 1119, Springer-Verlag.
6. D. Beyer. Efficient Reachability Analysis and Refinement Checking of Timed Automata Using BDDs. CHARME 2000, LNCS 2144, Springer-Verlag.
7. J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, Wang Yi. UPPAAL - a Tool Suite for Automatic Verification of Real-Time Systems. Hybrid Control System Symposium, 1996, LNCS, Springer-Verlag.
8. K. Cerans. Decidability of bisimulation equivalence for parallel timer processes. CAV'92, LNCS 663, Springer-Verlag.
9. R. Cleaveland, B. Steffen. Computing behavioral relations, logically. ICALP'91, Madrid, July 1991, LNCS 510, Springer-Verlag, pp.127-138.
10. J.B. Fourier. (reported in:) Analyse des travaux de l'Académie Royale des Sciences pendant l'année 1824, Partie Mathématique, 1827.
11. M.R. Henzinger, T.A. Henzinger, P.W. Kopke. Computing simulations on finite and infinite graphs. IEEE FOCS'95.
12. T.A. Henzinger, X. Nicollin, J. Sifakis, S. Yovine. Symbolic Model Checking for Real-Time Systems, IEEE LICS 1992.
13. H. Lin, Wang Y. Axiomatising timed automaton. Acta Informatica, Vol.38, Nr.4, 2002, pp.277-305.
14. A. Nakata. Symbolic Bisimulation Checking and Decomposition of Real-Time Service Specifications. Ph.D. Dissertation, Faculty of the Engineering Science, Osaka University, 1997.
15. S. Taşıran, R. Alur, R.P. Kurshan, R.K. Brayton. Verifying abstractions of timed systems. CONCUR'1996, LNCS 1119, Springer-Verlag.
16. F. Wang. Efficient Verification of Timed Automata with BDD-like Data-Structures, STTT (Software Tools for Technology Transfer), Vol. 6, Nr. 1, June 2004, Springer-Verlag; special issue for the 4th VMCAI, Jan. 2003, LNCS 2575, Springer-Verlag.
17. F. Wang. Symbolic Parametric Safety Analysis of Linear Hybrid Systems with BDD-like Data-Structures. IEEE Transactions on Software Engineering, Volume 31, Issue 1 (January 2005), pp. 38-51, IEEE Computer Society. A preliminary version is in proceedings of 16th CAV, 2004, LNCS 3114, Springer-Verlag.
18. F. Wang, G.-D. Huang, F. Yu. TCTL Inevitability Analysis of Dense-Time Systems: From Theory to Engineering. IEEE Transactions on Software Engineering, Vol. 32, Nr. 7, July 2006, IEEE Computer Society.
19. S. Yovine. Kronos: A Verification Tool for Real-Time Systems. International Journal of Software Tools for Technology Transfer, Vol. 1, Nr. 1/2, October 1997.

APPENDICES

A Fourier-Motzkin elimination for the special case

There are two cases to discuss. The first case is for calculating $\exists p_1(f(p_1, \ldots, p_m, x_1, \ldots, x_n))$ where p_1 is an atomic proposition. According to Shannon expansion, we have

$$\exists p_1(f(p_1, \ldots, p_m, x_1, \ldots, x_n)) \equiv f(\mathit{false}, \ldots, p_m, x_1, \ldots, x_n) \vee f(\mathit{true}, \ldots, p_m, x_1, \ldots, x_n)$$

The second case, for the calculation of $\exists x_1(f(p_1, \ldots, p_m, x_1, \ldots, x_n))$ where x_1 is a clock, can be handled with the following steps.

(1) Rewrite $f(p_1, \ldots, p_m, x_1, \ldots, x_n)$ in disjunctive normal form.
(2) For each disjunct, do the following three steps in sequence.
  (2.1) Classify the linear constraints in the following three types.
    TYPE I: $-x_1 + x_i \sim d$.
    TYPE II: $x_1 - x_j \sim' d'$.
    TYPE III: $x_i - x_j \sim d$ with $i \neq 1$ and $j \neq 1$.
  (2.2) For each TYPE I constraint $-x_1 + x_i \sim d$ and TYPE II constraint $x_1 - x_j \sim' d'$, conjunct constraint $x_i - x_j \sim'' d + d'$ to the disjunct. Here $\sim''=$ '$\leq$' if both $\sim=$ '$\leq$' and $\sim=$ '$\leq$'; and $\sim''=$ '$<$' otherwise.
  (2.3) Delete every constraint with clock x_1 in the disjunct.

B Proof for lemma 2

We need the following concepts for the proof of the lemma. Suppose we are given an $e_1 \in E_1$, a $\delta \in \mathbb{R}^{\geq 0}$, a run $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$ of A_2, a $k \geq 0$, and a $t \in [0, t_{k+1} - t_k]$. We say $t_k - t_0 + t$ is a *disrupting time* in ρ for μ, e_1, δ, and Q if and only if $t_k - t_0 + t < \delta$ and $(\mu + t_k + t - t_0)(\nu_k + t) \notin Q$. A run with a disrupting time for μ, e_1, δ, and Q is called a *disrupted run* for μ, e_1, δ, and Q.

Given a disrupted run ρ of A_2 for μ, e_1, δ, and Q, we define $\mathit{minT}^{\delta}_{e_1}(\mu, \rho, Q)$ as the smallest disrupting time in ρ for μ, e_1, δ, and Q. We let $\mathit{maxminT}^{\delta}_{e_1}(\mu, \nu, Q)$ be the maximum value of all such $\mathit{minT}^{\delta}_{e_1}(\mu, \rho, Q)$ with disrupted runs ρ starting from ν. That is,

$$\mathit{maxminT}^{\delta}_{e_1}(\mu, \nu, Q) = \max \left\{ \mathit{minT}^{\delta}_{e_1}(\mu, \rho, Q) \,\middle|\, \begin{array}{l} \rho \text{ is a disrupted run from } \nu \text{ of } A_2 \\ \text{for } \mu, e_1, \delta, \text{ and } Q \end{array} \right\}.$$

For convenience, given a run $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$, a $k \in \mathbb{N}$, and a $t \in [0, t_{k+1} - t_k]$, we can specify state $\nu_k + t$ in ρ as $\rho(k, t)$. A state $\rho(k, t)$ *precedes* another $\rho(k', t')$ in ρ if and only if either $k < k'$ or $k = k' \wedge t < t'$.

If there is no disrupting time $< \delta$ in ρ but there is neither a state at δ time units from ν to execute a transition matching e_1, then we call ρ a *futile run* for μ, e_1, δ, and Q. Formally speaking, a futile run $\rho = (\nu_0, t_0) \ldots (\nu_k, t_k) \ldots$ for μ, e_1, δ, and Q satisfies the following restrictions.

- $\nu_0 = \nu$.
- For every $k \in \mathbb{N}$ and $t \in [0, t_{k+1} - t_k]$, if $t_k + t - t_0 = \delta$, then there is a state $\rho(h, t')$ such that $t_h + t' - t_0 = \delta$, $\rho(k,t)$ does not precede $\rho(h,t')$, and either one of the following two conditions is true.
  - $(\mu + t_h - t_0 + t')(\nu_h + t') \notin Q$.
  - For all transitions e_2 in $E_2^{(e_1)}$, $\nu_h + t' \not\models \tau_2(e_2)$, $(\nu_h + t')\pi_2(e_2) \not\models H_2$, or $((\mu + t_h - t_0 + t')\pi_1(e_1))((\nu_h + t')\pi_2(e_2)) \notin Q$.

It is easy to see that if given a $\mu\nu \in Q$ such that $play_{e_1}^{\delta}(\mu) \wedge \neg match_{e_1}^{\delta}(\mu, \nu, Q)$ is true, then we know all runs of A_2 with states as a second components in Q must be either disrupted or futile for μ, e_1, δ, and Q.

Lemma 2. *Suppose we are given a $Q \subseteq V_{A_1} \times V_{A_2}$, a transition $e_1 \in E_1$, and a $\delta \in \mathbb{R}^{\geq 0}$. If there is a $\mu\nu \in Q$ such that $\mathrm{play}_{e_1}^{\delta}(\mu) \wedge \neg\mathrm{match}_{e_1}^{\delta}(\mu,\nu,Q)$ is true, then for each $t \in [0,\delta]$, there is a $\bar{\mu}\bar{\nu} \in Q$ such that either $\mathrm{play}_{e_1}^{t}(\bar{\mu}) \wedge \neg\mathrm{match}_{e_1}^{t}(\bar{\mu},\bar{\nu},Q)$ or $\mathrm{play}_{\perp}^{t}(\bar{\mu}) \wedge \neg\mathrm{match}_{\perp}^{t}(\bar{\mu},\bar{\nu},Q)$ is true.*

Proof : We assume that there is a $\mu\nu \in Q$ such that $play_{e_1}^{\delta}(\mu)$ is true and $match_{e_1}^{\delta}(\mu,\nu,Q)$ is false. There are two cases to analyze.

CASE I: In the first case, we assume that there is a futile run ρ. Then there is a $k \geq 0$ and a $t' \in [0, t_{k+1} - t_k]$ such that $\delta - (t_k + t' - t_0) = t$ in ρ. We argue that $\mu + \delta - t$ is a candidate for $\bar{\mu}$ while $\nu_k + t'$ is for $\bar{\nu}$. To prove $play_{e_1}^{t}(\mu + \delta - t)$, we have the following deduction steps.

$$\begin{aligned}
play_{e_1}^{\delta}(\mu) &\stackrel{\text{def}}{=} \mu + \delta \models \tau_1(e_1) \wedge (\mu+\delta)\pi_1(e_1) \models H_1 \wedge \neg\exists 0 \leq \delta' \leq \delta(\mu + \delta' \models \neg H_1)\\
&\Rightarrow (\mu + \delta - t) + t \models \tau_1(e_1) \wedge ((\mu+\delta-t)+t)\pi_1(e_1) \models H_1\\
&\quad \wedge \neg\exists 0 \leq \delta' \leq \delta(\mu + \delta' \models \neg H_1)\\
&\Rightarrow (\mu + \delta - t) + t \models \tau_1(e_1) \wedge ((\mu+\delta-t)+t)\pi_1(e_1) \models H_1\\
&\quad \wedge \neg\exists 0 \leq \delta' \leq t((\mu + \delta - t) + \delta' \models \neg H_1), \text{ since } t \leq \delta\\
&\Rightarrow play_{e_1}^{t}(\mu + \delta - t), \text{ definition}
\end{aligned}$$

The deduction steps end with the establishment of $play_{e_1}^{t}(\mu + \delta - t)$.

Now we need to prove the falsehood of $match_{e_1}^{t}(\mu + \delta - t, \nu_k + t_0 + \delta - t_k - t, Q)$. The existence of k and t' implies the existence of a pre-matching segment

$$(\nu_0, t_0)(\nu_1, t_1)\ldots(\nu_k, t_k)(\nu_k + t', t_k + t')$$

This further implies that $t' = t_0 + \delta - t_k - t$ and $pre\text{-}match_2^{\delta - t}(\mu, \nu, \nu_k + t_0 + \delta - t_k - t, Q)$ is true. In fact, in the definition of $match_{e_1}^{\delta}(\mu,\nu,\nu',Q)$, $pre\text{-}match_2^{\delta}(\mu,\nu,\nu',Q)$ is equivalent to $pre\text{-}match_2^{\delta-t}(\mu,\nu,\nu_k + t_0 + \delta - t_k - t, Q) \wedge pre\text{-}match_2^{t}(\mu + \delta - t, \nu_k + t_0 + \delta - t_k - t, \nu', Q)$. Thus we have the following deduction steps.

$$\begin{aligned}
\neg match_{e_1}^{\delta}(\mu,\nu,Q) &\stackrel{\text{def}}{=} \neg\exists \nu' \in V_{A_2} \left(\begin{array}{l} pre\text{-}match_2^{\delta}(\mu,\nu,\nu',Q) \\ \wedge\ \exists e_2 \in E_2^{(e_1)} \left(\begin{array}{l} \nu' \models \tau_2(e_2) \wedge \nu'\pi_2(e_2) \models H_2 \\ \wedge\ ((\mu+\delta)\pi_1(e_1), \nu'\pi_2(e_2)) \in Q \end{array} \right) \end{array} \right)\\
&\Leftrightarrow \neg\exists \nu' \in V_{A_2} \left(\begin{array}{l} pre\text{-}match_2^{\delta-t}(\mu,\nu,\nu_k + t_0 + \delta - t_k - t, Q) \\ \wedge\ pre\text{-}match_2^{t}(\mu + \delta - t, \nu_k + t_0 + \delta - t_k - t, \nu', Q) \\ \wedge\ \exists e_2 \in E_2^{(e_1)} \left(\begin{array}{l} \nu' \models \tau_2(e_2) \wedge \nu'\pi_2(e_2) \models H_2 \\ \wedge\ ((\mu+\delta)\pi_1(e_1), \nu'\pi_2(e_2)) \in Q \end{array} \right) \end{array} \right)
\end{aligned}$$

As argued in the above, the existence of k and t' implies that $pre\text{-}match_2^{\delta-t}(\mu,\nu,\nu_k + t_0 + \delta - t_k - t, Q)$. Thus we have

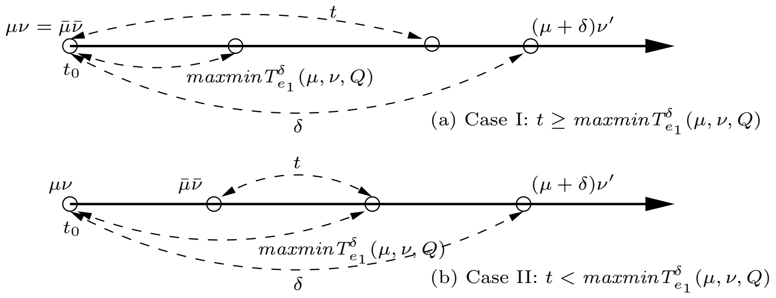


Fig. 2. Cases I & II in lemma 2

$$\neg match^{\delta}_{e_1}(\mu,\nu,Q) \Rightarrow \neg\exists\nu' \in V_{A_2}\left(\begin{array}{c} pre\text{-}match^{t}_{2}(\mu+\delta-t, \nu_k+t_0+\delta-t_k-t, \nu', Q) \\ \wedge\ \exists e_2 \in E_2^{(e_1)}\left(\begin{array}{c}\nu' \models \tau_2(e_2) \wedge \nu'\pi_2(e_2) \models H_2 \\ \wedge\ ((\mu+\delta)\pi_1(e_1), \nu'\pi_2(e_2)) \in Q\end{array}\right)\end{array}\right)$$
$$\equiv \neg match^{t}_{e_1}(\mu+\delta-t, \nu_k+t_0+\delta-t_k-t, Q)$$

Thus the lemma is also proven in this case.

CASE II: In the second case, we assume that there is no futile run for μ, e_1, δ, and Q. Then all runs of A_2 from ν are with a disrupting time $t < \delta$ for μ, e_1, δ, and Q. We let $\rho = (\nu_0, t_0)\ldots(\nu_k, t_k)\ldots$ of A_2 be a disrupted run for μ, e_1, δ, and Q such that $\nu_0 = \nu$ and $minT^{\delta}_{e_1}(\mu,\rho,Q) = maxminT^{\delta}_{e_1}(\mu,\nu,Q)$. There are two subcases to analyze.

CASE II(a): Assume $t > maxminT^{\delta}_{e_1}(\mu,\nu,Q)$. This subcase is illustrated in figure 2(a). Since $play^{\delta}_{e_1}(\mu)$ is true and $t \leq \delta$, we know that to make e_1 at δ time units from μ by A_1, we must first pass a state that is t time units from μ through pure time-progress. This implies that $play^{t}_{\perp}(\mu)$ is true. Moreover, according to the definition of $match^{\delta}_{e_1}(\mu,\nu,Q)$, in this case, for every $t \geq maxminT^{\delta}_{e_1}(\mu,\nu,Q)$, $match^{t}_{\perp}(\mu,\nu,Q)$ is also false. Then it is obvious that $play^{t}_{\perp}(\mu) \wedge \neg match^{t}_{\perp}(\mu,\nu,Q)$ is also true. According to the definition of simulation, we know that $play^{t}_{\perp}(\mu) \wedge \neg match^{t}_{\perp}(\mu,\nu,Q)$ is true and μ and ν are respectively candidates for $\bar{\mu}$ and $\bar{\nu}$. The lemma is proven in this subcase.

CASE II(b): If $t \leq minT^{\delta}_{e_1}(\mu,\rho,Q) = maxminT^{\delta}_{e_1}(\mu,\nu,Q)$, we assume that $\nu_k + t'$ with $t' \in [0, t_{k+1} - t_k]$ is the state in ρ that causes the disrupting. Since time is continuous, we can find an $h \in [0,k]$ and $t'' \in [0, t_{h+1} - t_h]$ such that $t_h - t_0 + t'' + t \geq t_k - t_0 + t'$. This subcase is illustrated in figure 2(b). The argument is similar to the one in CASE I. Then obviously, $\mu + t_h - t_0 + t''$ is a candidate to $\bar{\mu}$ while $\nu_h + t''$ is for $\bar{\nu}$. Thus the lemma is proven in this subcase. ∎

C Proof for lemma 3

Lemma 3. *Suppose we are given A_1 and A_2 such that $A_i = \langle \Sigma, X_i, G, L_i, I_i, H_i, E_i, \epsilon_i, \tau_i, \pi_i \rangle$, $1 \leq i \leq 2$, $L_1 \cap L_2 = \emptyset$, and $X_1 \cap X_2 = \emptyset$. Q is a simulation from A_1 to A_2 if and only if $Q = \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1 \exists\delta \in [0, C_{(A_1,A_2)}](\mathrm{play}^{\delta}_{e_1}(\mu) \wedge \neg\mathrm{match}^{\delta}_{e_1}(\mu,\nu,Q))\}$.*

Proof : We prove the lemma in two directions.
$\Rightarrow$: We first assume that Q is a simulation. Then we have the following deduction steps.

$$\begin{aligned}
&\neg\exists e_1 \in E_1(\mathit{not_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu,\nu,Q))\\
&\Longleftrightarrow \neg\exists e_1 \in E_1 \exists\delta \in \mathbb{R}^{\geq 0}(\mathit{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathit{match}^{\delta}_{e_1}(\mu,\nu,Q)); \text{ definition}\\
&\Longleftrightarrow \forall e_1 \in E_1 \forall\delta \in \mathbb{R}^{\geq 0}\neg(\mathit{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathit{match}^{\delta}_{e_1}(\mu,\nu,Q)); \text{ de Morgan's law for } \forall \text{ and } \exists\\
&\Longrightarrow \forall e_1 \in E_1 \forall\delta \in [0, C_{(A_1,A_2)}]\neg(\mathit{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathit{match}^{\delta}_{e_1}(\mu,\nu,Q)); \text{ since } [0, C_{(A_1,A_2)}] \subset \mathbb{R}^{\geq 0}\\
&\Longleftrightarrow \neg\exists e_1 \in E_1 \exists\delta \in [0, C_{(A_1,A_2)}](\mathit{play}^{\delta}_{e_1}(\mu) \wedge \neg \mathit{match}^{\delta}_{e_1}(\mu,\nu,Q)); \text{ de Morgan's law for } \forall \text{ and } \exists
\end{aligned}$$

Thus this direction is proven.
$\Leftarrow$: We assume that for all $\mu\nu \in Q$, $\exists e_1 \in E_1(\mathit{not\_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q))$ is false. This means that the following set is empty.

$$Y^{[0,C_{(A_1,A_2)}]} \stackrel{\text{def}}{=} \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \exists e_1 \in E_1(\mathit{not\_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q))\}.$$

According to lemma 2, this implies that the following set is empty.

$$Y^{\mathbb{R}^{\geq 0}} \stackrel{\text{def}}{=} \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \exists e_1 \in E_1(\mathit{not\_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu,\nu,Q))\}.$$

This allows for the following deduction steps.

$$\begin{aligned}
&\{\mu\nu \mid \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\mathit{not_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q))\}\\
&= \{\mu\nu \mid \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\mathit{not_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q))\} - Y^{\mathbb{R}^{\geq 0}}; \; Y^{\mathbb{R}^{\geq 0}} \text{ is empty.}\\
&= \left\{\mu\nu \,\middle|\, \begin{array}{l} \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\mathit{not_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q)),\\ \neg(\mu \models H_1 \wedge \nu \models H_2 \wedge \exists e_1 \in E_1(\mathit{not_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu,\nu,Q))) \end{array}\right\} \begin{array}{l}\text{; defintion of}\\ \text{; set subtraction}\end{array}\\
&= \left\{\mu\nu \,\middle|\, \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\mathit{not_sim}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mu,\nu,Q) \vee \mathit{not_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu,\nu,Q)) \right\}\\
&= \left\{\mu\nu \,\middle|\, \mu \models H_1, \nu \models H_2, \neg\exists e_1 \in E_1(\mathit{not_sim}^{\mathbb{R}^{\geq 0}}_{e_1}(\mu,\nu,Q)) \right\}; \text{ since } [0, C_{(A_1,A_2)}] \subseteq \mathbb{R}^{\geq 0}
\end{aligned}$$

The last line of the equality deduction is the definition of simulation in lemma 1. Thus the lemma is also proven in this direction. ■

D Proof for lemma 4

Lemma 4. *Given a state $\mu \in V_{A_1}$ and a real number $t \in [0, C_{(A_1,A_2)}]$, $\mathrm{play}^t_{e_1}(\mu)$ if and only if $\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Play}^{[0,C_{(A_1,A_2)}]}_{e_1}$.*
Proof : We first assume that $\mu \in \mathit{play}^t_{e_1}(\mu)$ with $t \in [0, C_{(A_1,A_2)}]$. We can make the following deduction steps from the definition of $\mathit{play}^t_{e_1}(\mu)$.

$$
\begin{aligned}
& play^t_{e_1}(\mu) \\
& \stackrel{\text{def}}{=} 0 \le t \le C_{(A_1,A_2)} \wedge \mu + t \models \tau_1(e_1) \wedge (\mu+t)\pi_1(e_1) \models H_1 \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \\
& \quad ;\text{ definition of } play^t_{e_1}() \\
& \Leftrightarrow 0 \le t \le C_{(A_1,A_2)} \wedge \mu + t \models \tau_1(e_1) \wedge \mu + t \models H_1\pi_1(e_1) \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \\
& \Leftrightarrow 0 \le t \le C_{(A_1,A_2)} \wedge \mu + t \models \tau_1(e_1) \wedge H_1\pi_1(e_1) \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \\
& \Leftrightarrow 0 \le t \le C_{(A_1,A_2)} \wedge \mu + t \models \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \\
& \Leftrightarrow \exists t \left(\begin{array}{l} \quad 0 \le t \le C_{(A_1,A_2)} \wedge [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge (\mu+t)[\Omega^\delta \leftarrow C_{(A_1,A_2)}] \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \\ \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \end{array} \right) \\
& \quad ;\text{ with the assumption that } \Omega^\delta = C_{(A_1,A_2)} - \delta \text{ and } 0 \le t \le C_{(A_1,A_2)}, \\
& \quad ;\text{ then } \Omega^\delta \ge 0 \equiv \delta \le C_{(A_1,A_2)} \text{ and } \Omega^\delta \le C_{(A_1,A_2)} \equiv \delta \ge 0. \\
& \Leftrightarrow \exists t \left(\begin{array}{l} \quad 0 \le t \wedge [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge (\mu+t)[\Omega^\delta \leftarrow C_{(A_1,A_2)}] \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \\ \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1) \end{array} \right) \\
& \quad ;\text{ since } [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \Omega^\delta \ge 0 \text{ implies } t \le C_{(A_1,A_2)} \\
& \Leftrightarrow \exists t \left(\begin{array}{l} \quad 0 \le t \wedge [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge (\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t) \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \\ \wedge \neg\exists 0 \le t' \le t(\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t' \models \neg H_1) \end{array} \right) \\
& \Leftrightarrow \exists t \left(\begin{array}{l} \quad 0 \le t \wedge \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge (\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t) \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \\ \wedge \neg\exists 0 \le t' \le t(\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t' \models \neg H_1) \end{array} \right) \\
& \Leftrightarrow \exists t \left(\begin{array}{l} \quad 0 \le t \wedge \hat{\mu} \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge (\hat{\mu}+t) \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \neg\exists 0 \le t' \le t(\hat{\mu} + t' \models \neg H_1) \end{array} \right) \\
& \quad ;\text{ renaming } [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \text{ as } \hat{\mu}. \\
& \Leftrightarrow \left(\begin{array}{l} \quad \hat{\mu} \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge \exists t \left(\, 0 \le t \wedge \hat{\mu} + t \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \neg\exists 0 \le t' \le t(\hat{\mu} + t' \models \neg H_1) \right) \end{array} \right) \\
& \quad ;\text{ since } \hat{\mu} \models \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \text{ is independent of } t. \\
& \Leftrightarrow \hat{\mu} \models \left(\begin{array}{l} \quad \Omega^\delta \ge 0 \wedge \Omega^\delta \le C_{(A_1,A_2)} \\ \wedge \mathtt{Tbck}(H_1, \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1)) \end{array} \right) ;\text{ definition of } \mathtt{Tbck}() \\
& \Leftrightarrow \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Play}^{[0,C_{(A_1,A_2)}]}_{e_1} ;\text{ definition of } \mathtt{Play}^{[0,C_{(A_1,A_2)}]}_{e_1}
\end{aligned}
$$

In the above deduction sequence, we explain each step with a statement beginning with a semicolon at the end of the step. The proof is completed with the deduction steps. ■

E Proof for lemma 6

Lemma 6. *Given zone-predicate $\mathcal{Q}$ for a set $Q \subseteq V_{A_1} \times V_{A_2}$, a state-pair $\mu\nu \in Q$, and a $t \in [0, C_{(A_1,A_2)}]$,* $\text{match}^t_{e_1}(\mu,\nu,Q)$ *is true if and only if* $\mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Match}^{[0,C_{(A_1,A_2)}]}_{e_1}(\mathcal{Q})$

Proof : Starting from the definition of $match^\delta_{e_1}()$, we have the following deduction sequence.

$$\exists \nu' \in V_{A_2} \left(\textit{pre-match}_2^t(\mu,\nu,\nu',Q) \wedge \exists e_2 \in E_2^{(e_1)} \left(\begin{array}{c} \nu' \models \tau_2(e_2) \wedge \nu'\pi_2(e_2) \models H_2 \\ \wedge\ ((\mu+\delta)\pi_1(e_1), \nu'\pi_2(e_2)) \in Q \end{array} \right) \right)$$

; defintion of $\textit{match}_{e_1}^t(\mu,\nu,Q)$

$$\Leftrightarrow \exists \nu' \in V_{A_2} \left(\textit{pre-match}_2^t(\mu,\nu,\nu',Q) \wedge \exists e_2 \in E_2^{(e_1)} \left(\nu' \models \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q}) \right) \right)$$

; definition of $\mathtt{Xbck}_{(e_1,e_2)}()$.

$$\Leftrightarrow \exists \nu' \in V_{A_2} \left(\textit{pre-match}_2^t(\mu,\nu,\nu',Q) \wedge \nu' \models \bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q}) \right) \text{; translation from } \exists \text{ to } \bigvee.$$

$$\Leftrightarrow \mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Rbck}_{E_2^{(\perp)}}^{[0,C_{(A_1,A_2)}]} \left(\mathcal{Q}, \bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q}) \right)$$

; lemma 5 and projection over ν'

$$\Leftrightarrow \mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q}) \text{; definition of } \mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}()$$

The proof is completed with the deduction sequence. ■

F Proof for lemma 7

Lemma 7. *Given a zone-predicate $\mathcal{Q}$ for a set $Q \subseteq V_{A_1} \times V_{A_2}$, a transition $e_1 \in E_1$, and a state-pair $\mu\nu \in S$, $\mu\nu \in \text{not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}(Q)$ if and only if $\mu\nu \models \mathtt{Not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q})$.*

Proof : From the definition of $\textit{not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}()$, we can conduct the following deduction sequence.

$$\exists t \in [0, C_{(A_1,A_2)}](\textit{play}_{e_1}^t(\mu) \wedge \neg \textit{match}_{e_1}^t(\mu,\nu,Q)) \text{; definition of } \textit{not\_sim}_{e_1}^{[0,C_{(A_1,A_2)}]}()$$

$$\Leftrightarrow \exists t \in [0, C_{(A_1,A_2)}] \left(\begin{array}{c} \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu) \\ \wedge\ \neg(\mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}\mathcal{Q}) \end{array} \right) \text{; lemmas 4 and 6.}$$

$$\Leftrightarrow \exists t \in [0, C_{(A_1,A_2)}] \left(\begin{array}{c} \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \mathtt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu) \\ \wedge\ \mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \neg\mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q}) \end{array} \right)$$

$$\Leftrightarrow \exists t \in [0, C_{(A_1,A_2)}](\mu\nu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models \left(\mathtt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu) \wedge \neg\mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q}) \right)$$

; $\mu\nu$ does not change any valuations in μ.

$$\Leftrightarrow \mu\nu \models \text{FM\_elm}\left(\left(\mathtt{Play}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mu) \wedge \neg\mathtt{Match}_{e_1}^{[0,C_{(A_1,A_2)}]}(\mathcal{Q}) \right), \{\Omega^\delta\} \right)$$

; FM_elm() implements the existential quantification.

The proof is completed with deduction sequence. ■

G Proof for lemma 9

Lemma 9. *Given a set $Q \subseteq V_{A_1} \times V_{A_2}$, a transition $e_1 \in E_1$, and a state-pair $\mu\nu \in Q$, if $\text{uapprox\_not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$ is true, then $\text{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$ is true.*

Proof : The truth of $\textit{uapprox\_not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$, according to the definition, implies that $\mu\nu \in Q \wedge \forall\delta \in (C_{(A_1,A_2)},\infty)(\textit{play}_{e_1}^\delta(\mu) \wedge \neg\textit{match}_{e_1}^\delta(\mu,\nu,Q))$. Since $(C_{(A_1,A_2)},\infty)$ is nonempty, this implies that $\mu\nu \in Q \wedge \exists\delta \in (C_{(A_1,A_2)},\infty)(\textit{play}_{e_1}^\delta(\mu) \wedge \neg\textit{match}_{e_1}^\delta(\mu,\nu,Q))$ which is the definition of $\textit{not\_sim}_{e_1}^{(C_{(A_1,A_2)},\infty)}(\mu,\nu,Q)$. Thus the lemma is proven. ■

H Proof for lemma 10

Lemma 10. *Given a state* $\mu \in V_{A_1}$ *and a* $t \in (C_{(A_1,A_2)}, \infty)$, $\text{play}^{t}_{e_1}(\mu)$ *is true if and only if for every* $t' \in (C_{(A_1,A_2)}, \infty)$, $\text{play}^{t'}_{e_1}(\mu)$ *is true.*

Proof : $play^{t}_{e_1}(\mu)$ is derived from the triggering condition $\tau(e_1)$ and H_1. In $\tau(e_1)$ and H_1, the timing constraints are with timing constants whose magnitudes are no greater than $C_{(A_1,A_2)}$. Thus for every inequality $x - x' \sim c$ in H_1 and $\tau(e_1)$ and $t, t' \in (C_{(A_1,A_2)}, \infty)$, $\mu + t \models x - x' \sim c$ if and only if $\mu + t' \models x - x' \sim c$. Thus we know if A_1 can make e_1 at state $\mu + t$, then so can A_1 at $\mu + t'$ for all $t' \in (C_{(A_1,A_2)}, \infty)$. ■

I Proof for lemma 11

Lemma 11. *Given a state* $\mu \in V_{A_1}$, $\forall t \in (C_{(A_1,A_2)}, \infty) \left(\text{play}^{t}_{e_1}(\mu)\right)$ *if and only if* $\mu \models \mathtt{UPlay}_{e_1}^{(C_{(A_1,A_2)},\infty)}$.

Proof : We have the following deduction steps.

$\forall t \in (C_{(A_1,A_2)},\infty)\left(play^t_{e_1}(\mu)\right)$
$\Leftrightarrow \exists t\left(t > C_{(A_1,A_2)} - t \wedge play^t_{e_1}(\mu)\right)$, according to lemma 10
$\Leftrightarrow \exists t\left(t > C_{(A_1,A_2)} \wedge \mu + t \models \tau_1(e_1) \wedge (\mu+t)\pi_1(e_1) \models H_1 \wedge \neg\exists 0 \le t' \le t(\mu + t' \models \neg H_1)\right)$
; definition of $play^t_{e_1}(\mu)$

$$\Leftrightarrow \exists t \begin{pmatrix} [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models t > C_{(A_1,A_2)} \wedge [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t \models \Omega^\delta = C_{(A_1,A_2)} \\ \wedge\, \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t \models \tau_1(e_1) \wedge (\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t)\pi_1(e_1) \models H_1 \\ \wedge\, \neg\exists 0 \le t' \le t(\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t' \models \neg H_1) \end{pmatrix}$$

; rewriting with Ω^δ

$$\Leftrightarrow \exists t \begin{pmatrix} [\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models t > C_{(A_1,A_2)} \\ \wedge\, \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}(e_1,\perp)(H_1) \\ \wedge\, \neg\exists 0 \le t' \le t(\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t' \models \neg H_1) \end{pmatrix}$$

; the definition of $\mathtt{Xbck}_{(e_1,\perp)}()$

$$\Leftrightarrow \exists t \begin{pmatrix} \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] \models t > C_{(A_1,A_2)} \\ \wedge\, \mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}(e_1,\perp)(H_1) \\ \wedge\, \neg\exists 0 \le t' \le t(\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t] + t' \models \neg H_1) \end{pmatrix}$$

$$\Leftrightarrow \exists t \begin{pmatrix} \hat\mu \models t > C_{(A_1,A_2)} \\ \wedge\, \hat\mu + t \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}(e_1,\perp)(H_1) \wedge \neg\exists 0 \le t' \le t(\hat\mu + t' \models \neg H_1) \end{pmatrix}$$

; renaming $\mu[\Omega^\delta \leftarrow C_{(A_1,A_2)} - t]$ as $\hat\mu$.

$$\Leftrightarrow \begin{pmatrix} \hat\mu \models \Omega^\delta < 0 \\ \wedge\, \exists t\left(\hat\mu + t \models \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}(e_1,\perp)(H_1) \wedge \neg\exists 0 \le t' \le t(\hat\mu + t' \models \neg H_1)\right) \end{pmatrix}$$

; $\hat\mu \models \Omega^\delta < 0$ is independent of t.

$$\Leftrightarrow \begin{pmatrix} \hat\mu \models \Omega^\delta < 0 \\ \wedge\, \hat\mu \models \mathtt{Tbck}\left(H_1, \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \delta = 0\right) \end{pmatrix}$$

; the definition of $\mathtt{Tbck}()$

$$\Leftrightarrow \hat\mu \models \begin{pmatrix} \Omega^\delta < 0 \\ \wedge\, \mathtt{Tbck}\left(H_1, \Omega^\delta = C_{(A_1,A_2)} \wedge \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \delta = 0\right) \end{pmatrix}$$

; the definition of $\mathtt{Tbck}()$
$\Leftrightarrow \mu \models \exists\Omega^\delta\left(\Omega^\delta < 0 \wedge \mathtt{Tbck}\left(H_1, \mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \Omega^\delta = C_{(A_1,A_2)}\right)\right)$
$\Leftrightarrow \mu \models \text{FM\_elm}\left(\Omega^\delta < 0 \wedge \mathtt{Tbck}\left(H_1, \left(\mathtt{Xbck}_{(e_1,\perp)}(H_1) \wedge \Omega^\delta = C_{(A_1,A_2)}\right)\right), \{\Omega^\delta\}\right)$
; since $\text{FM\_elm}(\eta,\{x\})$ implements $\exists x(\eta)$ for all x
$\Leftrightarrow \mu \models \mathtt{UPlay}^{(C_{(A_1,A_2)},\infty)}_{e_1}$; definition of $\mathtt{UPlay}^{(C_{(A_1,A_2)},\infty)}_{e_1}()$

Thus the lemma is proven. ■

J Proof for lemma 12

Lemma 12. *At statement (G) of procedure* $\mathtt{OMatch}^{(C_{(A_1,A_2)},\infty)}_{e_1}(Q)$*, for all* $(\mu,\nu) \in V_{A_1} \times V_{A_2}$*,* $\mu\nu \models \Delta$ *if and only if* $\mu\nu \models \exists\delta \in (C_{(A_1,A_2)},\infty)(\mathrm{match}^\delta_{e_1}(\mu,\nu,Q))$*.*

Proof : We prove the lemma in two directions.

$\Longrightarrow$: First we assume that $\exists\delta \in (C_{(A_1,A_2)},\infty)(match^\delta_{e_1}(\mu,\nu,Q))$ is true. This means that there is a pre-matching segment from ν of A_2 for μ, e_1, δ, and Q to a state-pair in $\bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$. Assume that the pre-matching segment consists of k transitions in $E_2^{(\perp)}$. This means that after k iterations of timed weakest precondition calculation, $\mu\nu$ can be characterized with either Δ or Π. Since the pre-matching segment is longer than $C_{(A_1,A_2)}$, $\mu\nu$ will be characterized in Δ.

Statement (G) can be executed only when $\Pi = \Pi' \wedge \Delta = \Delta'$. That means that we no longer can add new state-pairs to Π and Δ. This in turn means that $\mu\nu$ must satisfies Δ.

$\Longleftarrow$: Now we assume that $\mu\nu \models \Delta$ after statement (G) is executed. According to the procedure, this can happen only when at some point, we execute statements (D) or (F) to insert $\mu\nu$ to Δ. This is possible only when we have calculated a finite series of timed weakest preconditions through statements (D), (E), or (F) from those state-pairs in $\bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$. This finite series of timed weakest precondition calculation can be mapped to a finite pre-matching segment from $\mu\nu$ to those states in $\bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$. Since all state-pairs satisfying Δ are with $\Omega^\delta < 0$, which implies $\delta \in (C_{(A_1,A_2)}, \infty)$, we know that the timing distance from $\mu\nu$ to those states in $\bigvee_{e_2 \in E_2^{(e_1)}} \mathtt{Xbck}_{(e_1,e_2)}(\mathcal{Q})$ must all be greater than $C_{(A_1,A_2)}$. This implies that $\mu\nu \models \exists\delta \in (C_{(A_1,A_2)}, \infty)(match^{\delta}_{e_1}(\mu, \nu, Q))$. Thus the lemma is proven. ■

K Proof for lemma 14

Lemma 14. *A proper implementation of Sim_Check(A_1, A_2) can run in EXPTIME.*

Proof : If we implement the representations of relations between state-pairs and Ω^δ as regions [1], then it is clear that in any of the steps, we need to handle at most exponential numbers of regions. The greatest fixpoint loop iteratively eliminates regions of state-pairs from the fixpoint images. Since the number of regions in $Z_{C_{(A_1,A_2)}}(G \cup L_1 \cup L_2, X_1 \cup X_2 \cup \{\Omega^\delta\})$ is at most exponential, the greatest fixpoint loop can repeat at most an exponential number of times. Thus it is clear that with region implementation, both algorithms I and II run in EXPTIME. ■