Preservation of timed properties during an incremental development by components

Jacques Julliand, Hassan Mountassir, Emilie Oudot

LIFC - Laboratoire d'Informatique de l'Université de Franche-Comté
16, route de Gray, 25030 Besançon Cedex, France
Ph:+33 (0)3 81 66 64 51, Fax:+33 (0)3 81 66 64 50
{julliand,mountass,oudot}@lifc.univ-fcomte.fr

Abstract. We are interested in the preservation of local properties of timed components during their integration in a timed system. Timed components are modeled as timed automata or timed automata with deadlines. Properties considered are all safety and liveness properties which can be expressed with the timed linear logic MITL (Metric Interval Linear Logic), as well as non-zenoness and deadlock-freedom. Integration of components is a kind of incremental development which consists in checking locally the properties of the components, before integrating them in the complete system, using some composition operator. Of course, established properties have to be preserved by this integration. Checking preservation can be achieved by means of the verification of timed τ-simulation relations. Composability, compatibility and compositionality of these relations w.r.t. composition operators are properties which allow to reduce the cost of this verification. We examine these properties when integration is achieved with two different timed composition operators: the classic operator usually taken for timed systems and which uses a CSP-like composition paradigm, and a non-blocking operator closer to the CCS paradigm.
Key-words. τ-simulations, component-based timed systems, integration of components, preservation of timed linear properties.

1 Introduction

Incremental development methods are a way to cope with the state space explosion problem of model-checking, which is increased in the case of timed systems due to the presence of timing constraints. In particular, for component-based systems, a way to develop incrementally is to use integration of components. This method is indicated for the verification of local properties of the components. It consists in checking the properties in isolation on the component before integrating it in its environment, with some parallel composition operator. Model-checking is there still applicable since the size of the components is generally small enough. Of course, this method is valid only if established properties of the component still hold after integration.

In Bellegarde et al. (2005), we defined two τ-simulation relations, adapted to timed systems, with preservation abilities: a timed τ-simulation preserving safety properties, and a divergence-sensitive and stability-respecting (DS) timed τ-simulation which preserves all properties which can be expressed in the linear timed logic MITL (Metric Interval Temporal Logic) (Alur et al., 1996), strong non-zenoness and deadlock-freedom.
Properties such as composability, compatibility and compositionality of the simulation relations are essential properties for integration of components. Consider components A, B, C and D. Composability is a major property since it expresses that a component A simulates its composition with another component. The direct consequence is that properties of A (which are preserved by the simulation) are automatically preserved by composition. It is thus clearly essential for integration of components, or for the reuse of a component. Given some composition operator $\|$, compatibility states that if A simulates B (and thus, properties of A also hold on B) then $A\|C$ simulates $B\|C$. During development, it is beneficial for instance in the case of the replacement in a system of the component A by the component B. Compositionality is a consequence of compatibility since it expresses that if A simulates B and C simulates D then, $A\|C$ simulates $B\|D$.
Therefore, in this paper, we study if the simulations we defined in Bellegarde et al. (2005) allow to benefit of these properties, in particular when integration is achieved with one of the two following operators: the classic parallel composition operator used for timed systems and a non-blocking operator defined in Bornot et al. (1997). The first operator uses a composition paradigm *a la CSP* (Hoare, 1985). The second one is closer to the paradigm of *CCS* (Milner, 1989) and uses a notion of priorities between actions to favour synchronizations. This analysis shows that the timed τ-simulation is well-adapted to both operators, since we benefit of the three properties without any assumptions. The DS timed τ-simulation is appropriate in the case of the non-blocking operator, on some conditions. This study of the properties of the simulations with respect to these composition operators is the contribution of the paper.
The structure of the paper is the following. In section 2, we recall some background on timed systems. We present timed automata which is the formalism we use to model timed systems and the two composition operators we consider for these automata. Section 3 recalls the simulation relations we defined for timed systems in Bellegarde et al. (2005), and their preservation abilities. Section 4 presents the contributions of this paper. We study whether the simulations have the composability, compatibility and compositionality properties w.r.t. the two composition operators. Finally, section 5 presents a synthesis of the results obtained, as well as the consequences in terms of preservation during integration, and plans some future works.

2 Preliminary definitions

In this section, we recall some background on the models we consider for timed systems, i.e., timed automata and timed automata with deadlines. We also present the two operators that we consider in this work for their composition.

2.1 Timed automata

Timed automata were introduced in Alur and Dill (1994). They are finite automata with real-valued variables, called clocks, which model the time elapsing.

Clock valuations and clock constraints. Let X be a set of clocks. A clock valuation over X is a function $v : X \to \mathbb{R}^+$ mapping to each clock in X a value in $\mathbb{R}^+$. Let v be a valuation over X and $t \in \mathbb{R}^+$, the valuation $v + t$ is obtained by adding t to the value of each clock. Given $Y \subseteq X$, the reset in v of the clocks in Y, written $[Y := 0]v$ is the valuation obtained from v by setting to zero all clocks in Y, and leaving the values of other clocks ($\in X \backslash Y$) unchanged. The set $\mathcal{C}_{df}(X)$ of diagonal-free clock constraints over X is defined as follows:

$$g ::= x \sim c \mid g \wedge g \mid true \text{ where } x \in X, c \in \mathbb{N}, \text{ and } \sim \in \{<, \leq, =, \geq, >\}.$$

A valuation v over X satisfies a constraint $x \sim c$, written $v \in x \sim c$, if $v(x) \sim c$.

Syntax. A timed automaton (TA) is tuple $A = \langle Q, q_0, \Sigma, X, T, \texttt{Invar} \rangle$, where Q is a finite set of locations, q_0 is the initial location, Σ is an alphabet of names of actions and X is a finite set of clocks. `Invar` is a function mapping a clock constraint to each location, called its invariant. Each edge of a TA is a tuple $e = (q, g, a, r, q')$ where q and q' are respectively its source and target location, g is its guard, a is its label and r is a set of clocks to be reset by the edge.

Semantics. The semantics of a TA A is an infinite graph which states are pairs (q, v) where q is a location of A and v is a valuation over X s.t. $v \in \texttt{Invar}(q)$. Transitions are either discrete transitions or time transitions. Consider a state (q, v). Given an edge $e = (q, g, a, r, q')$ of A, $(q, v) \xrightarrow{a} (q', [r := 0]v)$ is a discrete transition if $v \in g$. Given $t \in \mathbb{R}^+$, time transitions have the form $(q, v) \xrightarrow{t} (q, v + t)$. Given a state $s = (q, v)$, $s + t$ denotes the pair $(q, v + t)$. A run of a TA is a path of its semantic graph. A run is said non-zeno if time can diverge along the run. A TA is said strongly non-zeno if all its runs are non-zeno.

A variant: timed automata with deadlines. Timed automata with deadlines (TAD) are a variant of TA introduced in Sifakis and Yovine (1996). The main difference lies in the fact that time-progress conditions are not given as invariants in locations, but are associated as deadlines with the edges. The deadline represents the moment when time can not progress any more before taking the edge. Formally, the syntax of TAD is the same as the one of TA, with no invariants. Edges are tuples (q, g, d, a, r, q') where q, g, a, r and q' are defined as for TA and d is a clock constraint representing the deadline.

2.2 Timed parallel composition operators

We consider two composition operators which take into account the timing constraints of the components. The first one, which is the classic operator for TA, uses a composition paradigm close to the one of CSP. The second, which we call non-blocking parallel composition operator, is closer to the paradigm of CCS and uses a notion of priorities between actions.

Classic parallel composition operator. This composition operator, written $\|$, operates between TA with disjoint sets of clocks. Intuitively, it is defined as a synchronized product, where actions with the same label synchronize, other actions interleave and time elapses synchronously between the components. Formally, consider two TA $A = \langle Q_A, q_{0_A}, \Sigma_A, X_A, T_A, \texttt{Invar}_A \rangle$ and $B = \langle Q_B, q_{0_B}, \Sigma_B, X_B, T_B, \texttt{Invar}_B \rangle$, such that $X_A \cap X_B = \emptyset$. The classic parallel composition of A and B, written $A \| B$, results in a TA which set of clocks is $X_A \cup X_B$ and which labels are in $\Sigma_A \cup \Sigma_B$. The set Q of locations is a subset of $Q_A \times Q_B$. The initial location is the pair (q_{0_A}, q_{0_B}). The invariant of a location (q_A, q_B) is $\texttt{Invar}(q_A) \wedge \texttt{Invar}(q_B)$. Edges in one TA which label is not the label of any edge in the other TA is an interleaving edge

in the composition and is still defined with the same guard, label and reset. Synchronized edges are obtained by the following rule:

$$\frac{(q_A, q_B) \in Q,\ (q_A, g_A, a, r_A, q'_A) \in T_A\ ,\ (q_B, g_B, a, r_B, q'_B) \in T_B}{((q_A, q_B), g_A \wedge g_B, a, r_A \cup r_B, (q'_A, q'_B)) \in T}$$

The main drawback of this operator is that deadlocks are generally introduced during composition. For this reason, other operators have been defined, such as the following one.

Non-blocking parallel composition operator. This operator was introduced in Bornot et al. (1997) to operate between TAD with disjoint alphabets and sets of clocks. It is defined as a product in which all actions interleave and time elapses synchronously between components. Some actions synchronize, according to a synchronization function $\mid\!: \Sigma_A \times \Sigma_B \rightarrow \Sigma_{sync} \cup \{\perp\}$, where Σ_{sync} is an alphabet disjoint from Σ_A and Σ_B. The function maps to each pair of labels (a, b) the label in Σ_{sync} of the action resulting of the synchronization of two actions a and b, or the special symbol $\perp$ if the two actions do not synchronize. Since all actions interleave, priorities are used to favour synchronized actions rather than interleaving. Different synchronization modes can be used. The AND mode is the classic one. The MIN mode corresponds to a synchronization with interruption, i.e., the first enabled action causes the synchronization even if the other one is not yet enabled. Finally, the MAX mode is a synchronization with waiting, i.e., the first enabled action waits for the other to be enabled for synchronization to occur.

3 Timed τ-simulations to preserve properties

In Bellegarde et al. (2005), we defined two kinds of simulation relations for timed automata. The first one, called timed τ-simulation, preserves safety properties. The second one, called divergence-sensitive and stability-respecting (DS) timed τ-simulation, preserves all properties expressed with the timed linear logic MITL (Metric Interval Temporal Logic), strong non-zenoness and deadlock-freedom.
Consider two TA A and B with respective alphabets Σ_A and Σ_B, s.t. $\Sigma_A \subseteq \Sigma_B$. In B, actions in $\Sigma_B \backslash \Sigma_A$ are called non-observable and renamed by τ. Other actions, in Σ_A, are called observable. In the sequel, we focus directly on the definition of the DS timed τ-simulation $\mathcal{S}_{ds}$. The definition of the timed τ-simulation, written $\mathcal{S}$, can be obtained by removing the clauses *divergence-sensitivity* and *stability-respect*. The predicate `free` (Tripakis, 1998), used in Definition 1, is defined as follows. Given a location q, $\texttt{free}(q)$ is the set of all valuations (of states with q as discrete part) from which a discrete transition can be taken after some delay.

Definition 1 (Divergence-sensitive and stability-respecting (DS) timed τ-simulation $\mathcal{S}_{ds}$) *Let* $A = \langle \mathrm{Q}_A, \mathrm{q}_{0_A}, \Sigma_A, \mathrm{X}_A, \mathrm{T}_A, \mathtt{Invar}_A \rangle$ *and* $B = \langle \mathrm{Q}_B, \mathrm{q}_{0_B}, \Sigma_A \cup \{\tau\}, \mathrm{X}_B, \mathrm{T}_B, \mathtt{Invar}_B \rangle$ *be two TA s.t.* $X_A \subseteq X_B$. *We call* S_A *and* S_B *the respective set of states of* A *and* B. *The DS timed* τ*-simulation* $\mathcal{S}_{ds}$ *is the greatest binary relation included in* $S_B \times S_A$. *Consider* $s_A = (q_A, v_A)$ *in* S_A *and* $s_B = (q_B, v_B)$ *in* S_B. *We say that* $s_B \mathcal{S}_{ds} s_A$ *if:*

1. *Strict simulation:* $s_B \xrightarrow{a} s'_B \wedge a \in \Sigma_A \ \Rightarrow \exists s'_A \cdot (s_A \xrightarrow{a} s'_A \ \wedge \ s'_B \ \mathcal{S}_{ds}\ s'_A)$.
2. *Delays equality:* $s_B \xrightarrow{t} s_B + t \ \Rightarrow s_A \xrightarrow{t} s_A + t \ \wedge \ s_B + t \, \mathcal{S}_{ds}\ s_A + t$.

3. *τ-transitions stuttering:* $s_B \xrightarrow{\tau} s'_B \Rightarrow s'_B \mathcal{S}_{ds} s_A$.
4. *Divergence-sensitivity:* B *does not contain any non-zeno* τ*-cycles.*
5. *Stability-respect:* $v_B \notin \texttt{free}(q_B) \Rightarrow v_A \notin \texttt{free}(q_A)$.

Given two TA A and B, and their respective initial state s_{0_A} and s_{0_B}, we say that A simulates B w.r.t. $\mathcal{S}_{ds}$ written $B \preceq_{\mathcal{S}_{ds}} A$ if $s_{0_B} \mathcal{S}_{ds} s_{0_A}$.

4 Properties of timed τ-simulations w.r.t. timed parallel composition

It seems interesting to avoid a systematic verification of the relations to ensure preservation during an incremental development. For this purpose, composability, compatibility and compositionality of the relations w.r.t. the composition operators used for integration of components are essential properties. Thus, in this section, we study these three properties for the timed τ-simulation and the DS one w.r.t. the two operators presented in section 2.2. In the sequel, we use the following notations. Given a TA A, we note S_A its set of states and Σ_A its alphabet. A state of A is simply written s_A or s'_A, which respectively represent the pairs (q_A, v_A) and (q'_A, v'_A). The initial state of A is written s_{0_A}.

4.1 Classic parallel composition

We first examine the properties with the timed τ-simulation.

Proposition 1 (Composability) *Let A and B be TA. We have: $A\|B \preceq_{\mathcal{S}} A$.*

PROOF. By construction of $A\|B$, its initial state is the pair (s_{0_A}, s_{0_B}). To prove that $A\|B \preceq_{\mathcal{S}} A$, it is enough to prove that $(s_{0_A}, s_{0_B}) \mathcal{S} s_{0_A}$. By definition, $\preceq_{\mathcal{S}}$ is the greatest relation included in $S_{A\|B} \times S_A$ which satisfies clauses 1 to 3 of Definition 1. Thus, each relation $R \subseteq S_{A\|B} \times S_A$ which satisfies these clauses is included in $\preceq_{\mathcal{S}}$. Consider a relation $R \subseteq S_{A\|B} \times S_A$ such that $\forall (s_A, s_B) \in S_{A\|B}$, $(s_A, s_B) R\, s'_A$ if $s_A = s'_A$. Consider $((s_A, s_B), s_A) \in R$.

1. *Strict simulation*: let $(s_A, s_B) \xrightarrow{a} (s'_A, s'_B)$ in $A\|B$ such that $a \in \Sigma_A$. By construction of $A\|B$, a transition $s_A \xrightarrow{a} s'_A$ exists in A. By definition of R, $(s'_A, s'_B) R\, s'_A$ and R satisfies the *strict simulation*.
2. *Delays equality*: same arguments than those for strict simulation can be used.
3. *τ-transitions stuttering*: consider a transition $(s_A, s_B) \xrightarrow{\tau} (s'_A, s'_B)$ in $A\|B$. Recall that τ-transitions represent non-observable actions initially labelled in $\Sigma_B \backslash \Sigma_A$. By construction of $A\|B$, $s'_A = s_A$. Thus, $(s_A, s'_B) R\, s_A$ and R satisfies *τ-transitions stuttering*. □

Proposition 2 (Compatibility) *Let A, B and C be TA. If $A \preceq_{\mathcal{S}} B$ then $A\|C \preceq_{\mathcal{S}} B\|C$.*

PROOF. The structure of the proof is similar to the previous one. A complete version can be found in Julliand et al. (2007). □

Proposition 3 (Compositionality) *Let A, B, C and D be TA. If $A \preceq_{\mathcal{S}} B$ and $C \preceq_{\mathcal{S}} D$ then $A\|C \preceq_{\mathcal{S}} B\|D$.*

PROOF. Immediate with Proposition 2. Since $A \preceq_{\mathcal{S}} B$, then $A\|C \preceq_{\mathcal{S}} B\|C$. Since $C \preceq_{\mathcal{S}} D$, then $B\|C \preceq_{\mathcal{S}} B\|D$. By transitivity of the relation $\preceq_{\mathcal{S}}$, we have $A\|C \preceq_{\mathcal{S}} B\|D$. □

The timed τ-simulation allows to benefit of the three properties for free. This is not the case for the DS timed τ-simulation. Indeed, the operator $\|$ is known to introduce deadlocks during composition, which prevents the clause stability-respect of the DS timed τ-simulation from being established. However, this simulation allows to get the properties when using the non-blocking parallel composition operator, on some simple conditions.

4.2 Non-blocking parallel composition

First note that the non-blocking parallel composition operates between TAD. Thus, in this section, we extend the notations $\preceq_{\mathcal{S}}$ and $\preceq_{\mathcal{S}_{ds}}$, initially defined for TA, to TAD. This extension does not matter since the simulations are defined at a semantic level and that the semantics of TAD is given by an infinite graph of the same kind than for TA.
Consider two TAD A and A', and suppose that A' is obtained from A by integration of components using the non-blocking parallel composition operator, i.e., $A' = A|B$ for some automaton B. For a TAD A to simulates a TAD A', we imposed in the definition of the simulation that $\Sigma_A \subseteq \Sigma_{A'}$. Moreover, we suppose that observable actions in A' (and, in particular, synchronized actions) have the same label than in A. Therefore, without loss of generality, we consider here that the synchronization function of the operator $|$ is defined by $\shortmid\colon \Sigma_A \times \Sigma_B \to \Sigma_A \cup \{\bot\}$ such that, given $a \in \Sigma_A$ and $b \in \Sigma_B$, $a \shortmid b = a$ if the two actions synchronize. In other words, the label of the synchronized action is the same than the one of the action of A which takes part in the synchronization.
We focus directly on the DS timed τ-simulation, and on the most used synchronization mode of the operator, the AND one. First, the following result is necessary for composability. Complete proofs for the following propositions can be found in Julliand et al. (2007).

Proposition 4 (Non τ-divergence preservation) *Let A and B be TAD. Actions in $\Sigma_B \backslash \Sigma_A$ are renamed by τ. If B does not contain any non-zeno τ-cycles, then $A|B$ does not contain any non-zeno τ-cycles.*

Proposition 5 (Composability) *Let A and B be TAD. Actions in $\Sigma_B \backslash \Sigma_A$ are renamed by τ in B. If B does not contain any non-zeno τ-cycles, we have: $A|B \preceq_{\mathcal{S}_{ds}} A$.*

PROOF. This proposition can be proved using the same method than for proposition 1. The proof for stability-respect is immediate by definition of $|$ and the fact that this operator does not introduce deadlocks, due to a total interleaving of all the actions. Divergence-sensitivity is ensured since τ-transitions of $A|C$ are labelled with actions in $\Sigma_B \backslash \Sigma_A$ and since $A \preceq_{\mathcal{S}_{ds}} B$. □

Proposition 6 (Compatibility) *Let A, B and C be TAD. If $A \preceq_{\mathcal{S}_{ds}} B$ then $A|C \preceq_{\mathcal{S}_{ds}} B|C$.*

PROOF. Similar arguments than in the proof of proposition 2 apply for clauses 1 to 3. The proof for divergence-sensitivity and stability-respect is immediate as in proposition 5. □

	Classic parallel composition	Non-blocking parallel composition	
		AND / MIN	MAX
Properties preserved during integration of components	safety	MITL, deadlock-freedom, strong non-zenoness (hyp. div. 1 and 2)	none

TAB. 1 – *Synthesis on the preservation of properties during integration of components*

Proposition 7 (Compositionality) *Let A, B, C and D be TA. The internal actions of $A|C$ (i.e. in $(\Sigma_A \cup \Sigma_C)\backslash(\Sigma_B \cup \Sigma_D)$) are renamed by τ. If $A \preceq_{\mathcal{S}_{ds}} B$, $C \preceq_{\mathcal{S}_{ds}} D$ and $A|C$ does not contain any non-zeno τ-cycles then $A|C \preceq_{\mathcal{S}_{ds}} B|D$.*

PROOF. Immediate with proposition 6. □

Remark 1 (MIN and MAX modes) *Recall that the MIN synchronization mode corresponds to a synchronization with interruption. As for the AND mode, this paradigm is taken into account by strengthening the guard of the synchronized action, which implies that the concerned clauses (strict simulation and delays equality) hold. Therefore, propositions 5 to 7 also hold when using the MIN mode. The MAX synchronization mode corresponds to a synchronization with waiting, which means that when one action in the synchronization is enabled, it waits for the other to be enabled to synchronize. It follows immediatly that the propositions do not hold. For instance, composability does not hold since synchronized actions (which are observable actions) can be taken later in $A|B$ than they were in A. Similar arguments can be given for compatibility and compositionality.*

Remark 2 (Timed τ-simulation and $|$) *We focused on the DS timed τ-simulation. The propositions also hold, without assumptions, in the case of timed τ-simulation, when using AND and MIN modes. Indeed, the DS timed τ-simulation is obtained from the timed τ-simulation by adding divergence-sensitivity and stability-respect. As the three properties hold for the DS timed τ-simulation (modulo assumptions for divergence-sensitivity in the case of composability and compositionality), they also hold for the timed τ-simulation.*

4.3 Synthesis

TAB. 1 gives an interpretation of these results in terms of properties preserved during an integration of components. The abbreviations *hyp. div. 1* and *hyp. div. 2* represent respectively the assumptions in propositions 5 and 7 for divergence-sensitivity.

5 Conclusion and Future works

In previous works, we defined τ-simulation relations for timed systems, with preservation abilities. Checking these relations is a way to guarantee the preservation of properties during incremental development of timed systems, in particular during integration of components. However, we wish to avoid the verification of the simulations, while still benefiting of their

preservation abilities. For this purpose, in this paper, we studied the properties of composability, compatibility and compositionality of the relations w.r.t. two composition operators for timed systems: the classic operator, and a non-blocking one with three different synchronization modes. It turns out that the properties hold for the timed τ-simulation with both operators (except when using the MAX synchronization mode with the non-blocking one). This means that the preservation of safety properties is ensured for free when using these operators for integration of components. The divergence-sensitive and stability-respecting timed τ-simulation has the properties only with the non-blocking operator (except with the MAX mode), on some conditions for divergence-sensitivity. Thus, MITL properties, deadlock-freedom and strong non-zenoness, are preserved (on the conditions expressed) during integration of components with this operator. This is not the case when using the classic operator. The reason is that this operator does not prevent from introducing deadlocks during composition, which makes the *stability-respecting* part of the simulation not guaranteed during integration of components.

References

Alur, R. and D. Dill (1994). A theory of timed automata. *Theoretical Computer Science 126*(2), 183–235.

Alur, R., T. Feder, and T. Henzinger (1996). The benefits of relaxing punctuality. *Journal of the ACM 43*, 116–146.

Bellegarde, F., J. Julliand, H. Mountassir, and E. Oudot (2005). On the contribution of a τ-simulation in the incremental modeling of timed systems. In *Proc. of FACS'05*, Volume 160 of *ENTCS*, Macao, Macao, pp. 97–111. Elsevier.

Bornot, S., J. Sifakis, and S. Tripakis (1997). Modeling Urgency in Timed Systems. In *COMPOS'97*, Volume 1536 of *LNCS*. Springer-Verlag.

Hoare, C. (1985). *Communicating Sequential Processes*. Prentice Hall.

Julliand, J., H. Mountassir, and E. Oudot (2007). Composability, compatibility, compositionality: automatic preservation of timed properties during incremental development. Research Report RR2007-01, LIFC, Laboratoire d'Informatique de l'Université de Franche-Comté.

Milner, R. (1989). *Communication and Concurrency*. Prentice Hall.

Sifakis, J. and S. Yovine (1996). Compositional Specification of Timed Systems. In *Proc. of STACS'96 - Invited Paper*, Volume 1046 of *LNCS*, pp. 347–359.

Tripakis, S. (1998). *The analysis of timed systems in practice*. Ph. D. thesis, Universite Joseph Fourier, Grenoble, France.