Timed Automata with non-Instantaneous Actions

Roberto Barbuti[1], Nicoletta De Francesco[2] and Luca Tesei[1]

[1] Dipartimento di Informatica
Università di Pisa
Corso Italia, 40
56125 Pisa - Italy

[2] Dipartimento di Ingegneria dell'Informazione
Università di Pisa
Via Diotisalvi, 2
56126 Pisa - Italy

Keywords: real-time systems, timed automata, timed languages.

Abstract. In this paper we propose a model, *timed automata with non-instantaneous actions*, which allows representing in a suitable way real-time systems. Timed automata with non-instantaneous actions extend the timed automata model by dropping the assumption that actions are instantaneous, that is an action can take some time to be completed. Thus, for an action σ, there are two particular time instants, the *initiation of the action* and the *completion of the action*, which may occur at different times. We investigate the expressiveness of the new model, comparing it with classical timed automata. In particular, we study the set of timed languages which can be accepted by timed automata with non-instantaneous actions. We prove that timed automata with non-instantaneous actions are more expressive than timed automata and less expressive than timed automata with ϵ edges. Moreover we define the parallel composition of timed automata with non-instantaneous actions. We show how real systems can be suitably modeled by them, in particular we specify a system which was specified in [7] by timed automata. We point out how the specification by means of a parallel timed automata with non-instantaneous actions is, in some cases, more convenient to represent reality.

1 Introduction

Transition systems have been intensively used as a model for specifying and verifying "real-life systems". The representation of a system by means of a finite transition system allows reasoning easily about qualitative properties of it, such as "safety" and "liveness".

When real-time systems are considered, also quantitative timing properties must be taken into account, since the correctness of the whole system may depend on the magnitude of different delays. For these systems, the specification by means of classical transition systems is no longer satisfactory because time requirements cannot be described within this model. A natural way to solve this problem is to add, to the usual model, a suitable notion of time. Examples of this extension can be found in [11, 15].

Alur and Dill proposed the model of *timed automata* [6, 7]. Since their introduction, timed automata have been widely studied from different points of view [4, 5, 8, 9], in particular for their possible use in the verification of real-time systems [1–3, 10, 14, 16, 19]. Timed automata can be represented by finite graphs augmented with a finite set of (real-valued) clocks. An edge is labeled by a symbol which represents the action performed when the edge is taken. While actions are instantaneous, time can elapse in a state. An edge can reset to zero the value of a set of clocks, and, at any instant, the value of a clock is equal to the time elapsed since the last reset on it. Any edge can be equipped with a constraint on the value of clocks. An edge may be taken only if the current value of clocks satisfies the constraint of the edge.

In this paper we propose a model, *timed automata with non-instantaneous actions*, which allows representing in a suitable way real-time systems. Essentially, timed automata with non-instantaneous actions extend the timed automata model by dropping the assumption that actions are instantaneous, that is an action can take some time to be completed. To model this feature, every edge is equipped with two constraints, a *initiation constraint* and a *completion constraint*. An edge can be taken when its initiation constraint is satisfied by the current value of clocks, and it can be completed only when its completion constraint is satisfied. Analogously, every edge is associated with a set of clocks which are reset to zero when it is taken (*initiation reset*) and a set of clocks which are reset to zero when the action is completed (*completion reset*). Thus, for an action σ, there are two particular time instants, the *initiation of the action* and the *completion of the action*, which may occur at different times. Following the notation of [17] we use the notation $\sigma\uparrow$ and $\sigma\downarrow$ for these two events.

After the definition, we investigate the expressiveness of the new model, comparing it with classical timed automata. In particular, we study the set of timed languages which can be accepted by timed automata with non-instantaneous actions.

A timed language is a set of *timed words*. A timed word over an alphabet, is a pair $(\overline{\sigma}, \overline{t})$ where $\overline{\sigma}$ is an infinite string $(\sigma_0\sigma_1\ldots)$ on the alphabet, and $\overline{t}$ is an infinite sequence of non-decreasing time values $(t_0t_1\ldots)$ associated to the symbols in $\overline{\sigma}$. A timed word $(\overline{\sigma}, \overline{t})$ is accepted by a timed automaton if there exists a path in the automaton, labeled by $\overline{\sigma}$, such that any edge labeled by σ_i is taken at time t_i, and satisfying some acceptance condition.

When dealing with non-instantaneous actions, the previous definition must be modified. In fact there is no longer the time in which an edge is instantaneously traversed, but the time of its initiation and of its completion. For this reason we define the notions of *initiation acceptance* and *completion acceptance*. In the first notion a symbol σ is considered as occurring at the time of its initiation while in the second one it is considered as occurring at the time of its completion. We prove that the class of *initiation accepted languages* and the one of *completion accepted languages* are different and both strictly include the class of languages accepted by timed automata. Initiation and completion acceptance are both included in the notion of *selected acceptance*, in which the symbols of the alphabet are partitioned into two subset: the subset of symbols the occurrence of which is considered at initiation time, and the subset of the ones the occurrence of which is considered at completion time. We show that the class of *selected accepted languages* strictly includes the previous two classes and it is strictly included in the class of languages accepted by timed automata with non-observable actions.

We prove that timed automata with non-instantaneous actions are more expressive than timed automata and less expressive than timed automata with ϵ edges.

Afterwards, we define the parallel composition of timed automata with non-instantaneous actions, and we show how they are a formalism to conveniently represent reality, when parallel non-instantaneous actions are involved. We show how a classical example of use of timed automata as a specification language [7], can be more suitably modeled by a parallel timed automaton with non-instantaneous actions.

2 Timed automata

We recall the definition of timed automata [7]. In the following, $\mathcal{R}$ is the set of real numbers and $\mathcal{R}^+$ the set of non-negative real numbers. A *clock* takes values from $\mathcal{R}^+$. Given a set $\mathcal{X}$ of clocks, a *clock valuation* over $\mathcal{X}$ is a function assigning a non-negative real number to every clock. The set of valuations of $\mathcal{X}$, denoted $\mathcal{V}_\mathcal{X}$, is the set of total function from $\mathcal{X}$ to $\mathcal{R}^+$. Given $\nu \in \mathcal{V}_\mathcal{X}$ and $\delta \in \mathcal{R}^+$, with $\nu + \delta$ we denote the valuation that maps each clock $x \in \mathcal{X}$ into $\nu(x) + \delta$.

Given a set $\mathcal{X}$ of clocks, a *reset* γ is a subset of $\mathcal{X}$. The set of all resets of $\mathcal{X}$ is denoted by $\Gamma_\mathcal{X}$. Given a valuation $\nu \in \mathcal{V}_\mathcal{X}$ and a reset γ, with $\nu \backslash \gamma$ we denote the valuation

$$\nu\backslash\gamma(x) = \begin{cases} 0 & \text{if } x \in \gamma \\ \nu(x) & \text{if } x \notin \gamma \end{cases}$$

Given a set $\mathcal{X}$ of clocks, the set $\Psi_\mathcal{X}$ of *clock constraints* over $\mathcal{X}$ are defined by the following grammar:

$$\psi ::= true \mid false \mid \psi \wedge \psi \mid \psi \vee \psi \mid \neg\psi \mid x\#t \mid x - y\#t$$

where $x, y \in \mathcal{X}$, $t \in \mathcal{R}^+$, and $\#$ is a binary operator in $\{<, >, \leq, \geq, =\}$. Clock constraints are evaluated over clock valuations. The satisfaction by a valuation $\nu \in \mathcal{V}_\mathcal{X}$ of the clock constraint $\psi \in \Psi_\mathcal{X}$, denoted $\nu \models \psi$, is defined as follows:

$\nu \models true$ and $\nu \not\models false$

$\nu \models \psi_1 \wedge \psi_2$ iff $\nu \models \psi_1 \wedge \nu \models \psi_2$
$\nu \models \psi_1 \vee \psi_2$ iff $\nu \models \psi_1 \vee \nu \models \psi_2$
$\nu \models \neg\psi$ iff $\nu \not\models \psi$
$\nu \models x\#t$ iff $\nu(x)\#t$
$\nu \models x - y\#t$ iff $\nu(x) - \nu(y)\#t$

Definition 1 (Timed automaton). *A timed automaton T is a tuple $(Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$, where: Q is a finite set of states, Σ is a finite alphabet of actions, $\mathcal{E}$ is a finite set of edges, $B \subseteq Q$ is the set of initial states, $R \subseteq Q$ is the set of repeated states, $\mathcal{X}$ is a finite set of clocks. Each edge $e \in \mathcal{E}$ is a tuple in $Q \times \Psi_{\mathcal{X}} \times \Gamma_{\mathcal{X}} \times \Sigma \times Q$.*

If $e = (q, \psi, \gamma, \sigma, q')$ is an edge, q is the source*, q' is the* target*, ψ is the* constraint*, σ is the* label*, γ is the* reset*.*

The semantics of a timed automaton T is an infinite transition system $\mathcal{S}(T) = (S, \rightarrow)$, where S is a set of states and $\rightarrow$ is the transition relation. The states S of $\mathcal{S}(T)$ are pairs (q, ν), where $q \in Q$ is a state of T, and ν is a valuation. An initial state of $\mathcal{S}(T)$ is a state (q, ν), where $q \in B$ is an initial state of T and ν is the valuation which assigns 0 to every clock in $\mathcal{X}$. At any state q, given a valuation ν, T can stay idle or it can perform an action labeling an outgoing edge e. If T stays idle, a transition is possible to a state of $\mathcal{S}(T)$ where the state of T is the same, but the valuation has been modified according to the elapsed time. If T moves along an outgoing edge $e = (q, \psi, \gamma, \sigma, q')$, this corresponds to a transition, labeled by σ, of $\mathcal{S}(T)$ from the state (q, ν) to the state $q', \nu\backslash\gamma$. This transition is possible only if the current clock valuation respects the constraint ψ of e. The rules to derive the transitions of $\mathcal{S}(T)$ are the following:

1. $\dfrac{\delta \in \mathcal{R}^+}{(q, \nu) \xrightarrow{\delta} (q, \nu + \delta)}$ 2. $\dfrac{(q, \psi, \gamma, \sigma, q') \in \mathcal{E}, \nu \models \psi}{(q, \nu) \xrightarrow{\sigma} (q', \nu\backslash\gamma)}$

Rule 1. represents the case in which T stays idle in a state and the time passes, while Rule 2. corresponds to the occurrence of an action.

Definition 2 (run, action sequence). *Given a timed automaton $T = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$, a* run *of the automaton is an infinite sequence of states and transitions of $\mathcal{S}(T)$*

$s_0 \xrightarrow{l_0} s_1 \xrightarrow{l_1} \ldots$

where

- *$s_0 = (q, \nu)$ where $q \in B$ and $\nu(x) = 0$ for every $x \in \mathcal{X}$*
- *a state $q \in R$ exists such that q occurs infinitely often in the pairs of the sequence $\{s_i\}$*

Note that, given a run $s_0 \xrightarrow{l_0} s_1 \xrightarrow{l_1} \ldots$, for each i, $l_i \in (\Sigma \cup \mathcal{R}^+)$. Let r be a run.

- *The* time sequence $\overline{t_j}$ *of the time elapsed from state* s_0 *to state* s_j *in* r *is defined as follows:*

$$t_0 = 0$$
$$t_{i+1} = t_i + \begin{cases} 0 \ \textit{if}\ l_i \in \Sigma \\ l_i \ \textit{otherwise} \end{cases}$$

- *The* event sequence *of the events occurring during* r*, including the elapsed times, is defined as follows:*
  $(l_0, t_0)(l_1, t_1)\ldots$
- *The* action sequence *of* r *is the projection of the event sequence of* r *on the pairs* $\{(l,t)|l \in \Sigma\}$

Definition 3 (timed word, timed language). *Let* Σ *be an alphabet. A* timed word *over* Σ *is a pair* $(\overline{\sigma}, \overline{t})$ *where* $\overline{\sigma} = \sigma_0\sigma_1\ldots$ *is an infinite sequence such that* $\sigma_i \in \Sigma$ *for all* i*, and* $\overline{t} = t_0t_1\ldots$ *is an infinite sequence such that* $t_i \in \mathcal{R}^+$ *and* $t_i \leq t_i + 1$*, for all* i*.*

A timed language *over* Σ *is a subset of the set of all timed words over* Σ*.*

Definition 4 (acceptance). *Given a timed automaton* $T = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$*, a timed word* w *over* Σ *is* accepted *by* T *if a run* r *of* T *exists such that* $w = v$*, where* v *is the action sequence of* r*. The set of timed words accepted by* T *is called the* accepted language *of* T *and is denoted by* $\mathcal{L}(T)$*.*

Note that we use the Büchi acceptance condition for the runs [7].

Automata with ϵ edges are defined in the same way, but with a special action, the non-observable action ϵ, belonging to Σ. Thus some transitions of the semantic automaton $\mathcal{S}(T)$ can be labeled by ϵ, and they are called ϵ transitions. When defining the accepted language for an automaton with ϵ edges, we must consider the action sequences as the the projection of the event sequences on the pairs $\{(l,t)|l \in \Sigma - \{\epsilon\}\}$.

Let us denote by $\mathcal{T}$ the set of timed automata and by $\mathcal{T}_\epsilon$ the set of timed automata with ϵ edges. We denote by $\mathcal{L}(\mathcal{T})$ and $\mathcal{L}(\mathcal{T}_\epsilon)$ the classes of timed languages accepted by automata in $\mathcal{T}$ and in $\mathcal{T}_\epsilon$, respectively. It is well-known that ϵ edges add power to timed automata. In fact, in [12] it is proved $\mathcal{L}(\mathcal{T}) \subset \mathcal{L}(\mathcal{T}_\epsilon)$.

3 Timed automata with non-instantaneous actions

Let us introduce a new class of timed automata, where the actions are considered non-instantaneous.

Definition 5 (Timed automaton with non-instantaneous actions). *A timed automaton with non-instantaneous actions* N *is a tuple* $(Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$*, where:* Q *is a finite set of states,* Σ *is a finite alphabet of actions,* $\mathcal{E}$ *is a finite set of edges,* $B \subseteq Q$ *is the set of initial states,* $R \subseteq Q$ *is the set of repeated states,* $\mathcal{X}$ *is a finite set of clocks.*

Each edge $e \in \mathcal{E}$ is a tuple in $Q \times \Psi_{\mathcal{X}} \times \Gamma_{\mathcal{X}} \times \Sigma \times \Psi_{\mathcal{X}} \times \Gamma_{\mathcal{X}} \times Q$. If $e = (q, \psi^i, \gamma^i, \sigma, \psi^c, \gamma^c, q')$ is an edge, q is the source*, q' is the* target*, ψ^i and ψ^c are the* initiation constraint *and the* completion constraint*, respectively, σ is the* label*, γ^i and γ^c are the* initiation reset *and the* completion reset*, respectively.*

The set of all timed automata with non-instantaneous actions will be denoted by $\mathcal{N}$.

The semantics of a timed automaton with non-instantaneous actions, $N = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$, is an infinite transition system $\mathcal{S}(N) = (S, \rightarrow)$, where S is a set of states and $\rightarrow$ is the transition relation. The states S of $\mathcal{S}(N)$ are of two kinds:

- a pair (q, ν), where $q \in Q$ is a state of N and $\nu \in \mathcal{V}_{\mathcal{X}}$ is a valuation;
- a pair $(\overrightarrow{\sigma}_{(\psi,\gamma,q)}, \nu)$ where $\sigma \in \Sigma$, $\psi \in \Psi_{\mathcal{X}}$, $\gamma \in \Gamma_{\mathcal{X}}$, $q \in Q$ and $\nu \in \mathcal{V}_{\mathcal{X}}$. These states represent the execution of action σ after its initiation. They record which constraint must hold, which clocks must be reset and which state must be reached at the completion of σ.

The rules to derive the transitions of $\mathcal{S}(N)$ are the following:

$$1.\ \frac{\delta \in \mathcal{R}^+}{(q, \nu) \xrightarrow{\delta} (q, \nu + \delta)} \qquad 2.\ \frac{(q, \psi^i, \gamma^i, \sigma, \psi^c, \gamma^c, q') \in \mathcal{E}, \nu \models \psi^i}{(q, \nu) \xrightarrow{\sigma\uparrow} (\overrightarrow{\sigma}_{(\psi^c,\gamma^c,q')}, \nu \backslash \gamma^i)}$$

$$3.\ \frac{\delta \in \mathcal{R}^+}{(\overrightarrow{\sigma}_{(\psi,\gamma,q)}, \nu) \xrightarrow{\delta} (\overrightarrow{\sigma}_{(\psi,\gamma,q)}, \nu + \delta)} \qquad 4.\ \frac{\nu \models \psi}{(\overrightarrow{\sigma}_{(\psi,\gamma,q)}, \nu) \xrightarrow{\sigma\downarrow} (q, \nu \backslash \gamma)}$$

Rule 1. is analogous to the one for timed automata. If N moves along an outgoing edge $e = (q, \psi^i, \gamma^i, \sigma, \psi^c, \gamma^c, q')$, this corresponds to a transition of $\mathcal{S}(N)$ from the state (q, ν) to the state $(\overrightarrow{\sigma}_{(\psi^c,\gamma^c,q')}, \nu \backslash \gamma^i)$ in which σ is initiated (Rule 2.). This transition is possible only if the current clock valuation satisfies the initiation constraint ψ^i of e, and is labeled by $\sigma \uparrow$, the initiation of σ. In this state the valuation of the clocks is modified according to the initiation reset γ^i of e. Moreover, the state records the completion constraint and the completion reset of e, to be considered for the completion of σ. In this state, some time can elapse: in this case $\mathcal{S}(N)$ reaches a state, where σ continues to be executed, but the valuation of clocks is modified according to the elapsed time (Rule 3.). When the execution of σ terminates, N reaches a new state (the target of e) with a new valuation of the clocks, given by the completion reset of e (Rule 4.). This transition is labeled by $\sigma \downarrow$, i.e. the completion of σ. This transition is possible only if the current clock valuation satisfies the completion constraint of e.

As for timed automata, the language accepted by a timed automaton with non-instantaneous actions $N = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$ is a timed language over Σ. The difference is that we can choose, for each action $\sigma \in \Sigma$, if we want to consider the time of its initiation or the time of its completion. If, for some action σ, we choose to consider its initiation, σ is considered as occurring when $\sigma\uparrow$ occurs, and $\sigma\downarrow$ is ignored, while, if we choose to consider its completion, σ is considered as occurring when $\sigma\downarrow$ occurs, and $\sigma\uparrow$ is ignored.

Given $A \subseteq \Sigma$, let us denote by $A\uparrow = \{\sigma\uparrow | \sigma \in A\}$ and $A\downarrow = \{\sigma\downarrow | \sigma \in A\}$ the set of initiations and completions of the actions in A, respectively.

Definition 6 (run, selected sequence, initiation sequence, completion sequence). *Given a timed automaton with non-instantaneous actions $N = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$, the runs of N are defined in the same way as for timed automata. Note that, given a run $s_0 \xrightarrow{l_0} s_1 \xrightarrow{l_1} \ldots$, for each i, $l_i \in (\Sigma\uparrow \cup \Sigma\downarrow \cup \mathcal{R}^+)$. Moreover, the time sequence and the event sequence of a run r are defined as for timed automata.*

Given a run r and a partition, (I, C), of Σ, that is $I \cup C = \Sigma$ and $I \cap C = \emptyset$:

- *The (I, C)* selected action sequence *of r is the projection of the event sequence of r on the pairs $\{(l, t) | l \in I\uparrow \cup C\downarrow\}$.*
- *The* initiation sequence *of the initiations of the actions in r is the $(\Sigma, \emptyset)$ selected action sequence of r, i.e. the projection of the event sequence of r on the pairs $\{(l, t) | l \in \Sigma\uparrow\}$.*
- *The* completion sequence *of the completions of the actions in r is the $(\emptyset, \Sigma)$ selected action sequence of r, i.e. the projection of the event sequence of r on the pairs $\{(l, t) | l \in \Sigma\downarrow\}$.*

Example 1. Consider the automaton N_1 in Figure 1.

- Let r be the run $q_0 \xrightarrow{0.3} q_0 \xrightarrow{0.7} q_0 \xrightarrow{a\uparrow} \overrightarrow{a}_{(true,\{\},q_0)} \xrightarrow{0.2} \overrightarrow{a}_{(true,\{\},q_0)} \xrightarrow{a\downarrow} q_0 \xrightarrow{0.1} q_0 \xrightarrow{b\uparrow} \overrightarrow{b}_{(x=1,\{x\},q_0)} \xrightarrow{0.7} \overrightarrow{b}_{(x=1,\{x\},q_0)} \xrightarrow{b\downarrow} q_0 \xrightarrow{1} q_0 \xrightarrow{a\uparrow} \overrightarrow{a}_{(true,\{\},q_0)} \xrightarrow{a\downarrow} q_0 \ldots$
- $(a\uparrow, 1)(b\uparrow, 1.3)(a\uparrow, 3)\ldots$ is the initiation sequence of r.
- $(a\downarrow, 1.2)(b\downarrow, 2)(a\downarrow, 3)\ldots$ is the completion sequence of r.
- $(a\uparrow, 1)(b\downarrow, 2)(a\uparrow, 3)\ldots$ is the $(\{a\}, \{b\})$ selected sequence of r.

Definition 7 (selected acceptance, initiation acceptance, completion acceptance). *Let $N = (Q, \Sigma, \mathcal{E}, B, R, \mathcal{X})$ be a timed automaton with non-instantaneous actions, and (I, C) a partition of Σ.*

- *A timed word w over Σ is (I, C)* selected accepted *by N if a run r of N exists such that, for all $\sigma \in I, \sigma' \in C$, $w = v[\sigma/\sigma\uparrow, \sigma'/\sigma'\downarrow]$, where v is the (I, C) selected sequence of r and $v[\sigma/\sigma\uparrow]$ denotes the sequence v in which every symbol $\sigma\uparrow \in I\uparrow$ is substituted by σ (analogously for $v[\sigma/\sigma\downarrow]$). The set of timed words (I, C) selected accepted by N is called the (I, C)* selected accepted language *of N and is denoted by $\mathcal{L}^s_{(I,C)}(N)$.*
- *A timed word w over Σ is* initiation accepted *by N iff it is $(\Sigma, \emptyset)$ selected accepted by N. We shall use $\mathcal{L}^i(N)$ for $\mathcal{L}^s_{(\Sigma,\emptyset)}(N)$, to denote the* initiation accepted language *of N.*

- *A timed word w over Σ is* completion accepted *by N iff it is $(\emptyset, \Sigma)$ selected accepted by N. We shall use $\mathcal{L}^c(N)$ for $\mathcal{L}^s_{(\emptyset,\Sigma)}(N)$, to denote the* completion accepted language *of N.*
- *The class of timed languages selected (for any (I, C)), initiation and completion accepted by automata in $\mathcal{N}$ is denoted by $\mathcal{L}^s(\mathcal{N})$, $\mathcal{L}^i(\mathcal{N})$, $\mathcal{L}^c(\mathcal{N})$, respectively.*

Example 2. Consider the automaton N_1 of Figure 1. The initiation accepted language L_1 is the set of all timed words $(\overline{\sigma}, \overline{t})$, $\overline{\sigma} = \sigma_0\sigma_1\sigma_2\ldots$, $\overline{t} = t_0t_1t_2\ldots$, such that each $t_i \in (i, i+1]$, for all natural numbers i, and either $\sigma_i = a$ and $t_i = i+1$, or $\sigma_i = b$ and $t_i \in (i, i+1)$.

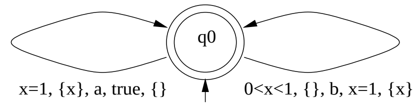

Fig. 1. Automaton N_1

Example 3. Consider the automaton N_2 of Figure 2. The completion accepted language L_2 is the set of all timed words $(\overline{\sigma}, \overline{t})$, $\overline{\sigma} = ccc\ldots$, $\overline{t} = t_0t_1t_2\ldots$, such that $t_i < t_{i+1}$, and there exists $\gamma > 0$ such that $0 < t_i < t_0 + 1 - \gamma$, for all i.

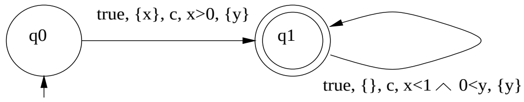

Fig. 2. Automaton N_2

Proposition 1. 1. $\mathcal{L}^i(\mathcal{N}) \not\subseteq \mathcal{L}^c(\mathcal{N})$, 2. $\mathcal{L}^c(\mathcal{N}) \not\subseteq \mathcal{L}^i(\mathcal{N})$.

Proposition 2. $\mathcal{L}^i(\mathcal{N}) \cup \mathcal{L}^c(\mathcal{N}) \subset \mathcal{L}^s(\mathcal{N})$.

We now give a theorem which states that the power of timed automata with non-instantaneous actions is greater than that of timed automata (without ϵ edges).

Theorem 1. 1. $\mathcal{L}(\mathcal{T}) \subset \mathcal{L}^i(\mathcal{N})$, 2. $\mathcal{L}(\mathcal{T}) \subset \mathcal{L}^c(\mathcal{N})$

Note that neither the language L_1, initiation accepted by the automaton N_1, nor the language L_2, completion accepted by the automaton N_2, can be accepted by a timed automaton without ϵ edges.

The following theorem states that timed automata with non-instantaneous actions are less expressive than timed automata with ϵ edges.

Theorem 2. $\mathcal{L}^s(\mathcal{N}) \subset \mathcal{L}(\mathcal{T}_\epsilon)$.

As a consequence, our model can be put at an intermediate level between timed automata and timed automata with ϵ edges. Automata with periodic clock constraints, defined in [13], also have an expressive power between timed automata and timed automata with ϵ edges. They contain constraints which are based on regularly repeated time intervals. However, their power is not comparable with the power of our model. In fact the aim of automata with periodic clock constraints is that of model periodic behaviors, while we model actions that have a duration.

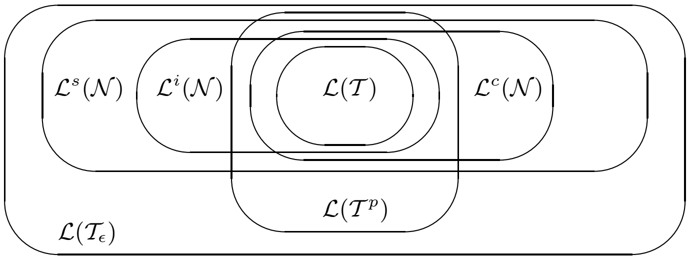

Fig. 3. Inclusion among language classes

The inclusion among language classes is given in Figure 3, where $\mathcal{L}(\mathcal{T}^p)$ is the class of languages accepted by timed automata with periodic clock constraints.

4 Parallel composition of timed automata with non-instantaneous actions

In this section we define the parallel composition of timed automata with non-instantaneous actions.

Definition 8 (Parallel composition). *Let $N_1, N_2, \ldots, N_n$ be timed automata with non-instantaneous actions, such that each N_i is defined by $(Q_i, \Sigma_i, \mathcal{E}_i, B_i, R_i, \mathcal{X}_i)$, for all $i \in [1, n]$.*

We denote by $(N_1 \parallel N_2 \parallel \ldots \parallel N_n)$ the parallel composition of $N_1, N_2, \ldots, N_n$.

We require that the states and the clock, for every pair of automata, are disjoint each other, that is for all $i, j \in [1, n]$, $i \neq j$, both $S_i \cap S_j = \emptyset$ and $\mathcal{X}_i \cap \mathcal{X}_j = \emptyset$.

Definition 9 (Synchronization function). *Let $(N_1 \parallel N_2 \parallel \ldots \parallel N_n)$ be a parallel composition of timed automata with non-instantaneous actions.*

We denote by $\mathcal{F}^{sync}_{(N_1 \parallel N_2 \parallel \ldots \parallel N_n)} : \Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_n \rightarrow \wp([1, n])$ (where $\wp$ is the powerset operator),
the function which, given an action in $\Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_n$, results in the set of indexes of the alphabets to which the action belong.

$\mathcal{F}^{sync}_{(N_1 \| N_2 \| \ldots \| N_n)}(\sigma) = \{k \in [1, n] \mid \sigma \in \Sigma_k\}$.

In the following, when clear from the context, we use $\mathcal{F}^{sync}$ in place of $\mathcal{F}^{sync}_{(N_1 \| N_2 \| \ldots \| N_n)}$. $\mathcal{F}^{sync}$ applied to an action $\sigma \in \Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_n$, results in the set of indexes of the automata which must synchronize on that action. This means that, if $\mathcal{F}^{sync}(\sigma) = A$, all the processes N_j, such that $j \in A$, must initiate and terminate the action σ at the same time. Of course, if A is the singleton $\{k\}$, the process N_k can proceed executing σ independently from other processes.

Let $(N_1 \parallel N_2 \parallel \ldots \parallel N_n)$ be a parallel composition of timed automata with non-instantaneous actions, its semantics is given by a transition system $\mathcal{S}_{\|}(N_1 \parallel N_2 \parallel \ldots \parallel N_n)$, the states of which are called *global configurations*. A global configuration $\langle \mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n \rangle$ is a tuple in which each $\mathcal{C}_i$ is a state of the transition system $\mathcal{S}(N_i)$.

The rules to derive the transitions of $\mathcal{S}_{\|}(N_1 \parallel N_2 \parallel \ldots \parallel N_n)$ are the following:

1. $$\frac{\delta \in \mathcal{R}^+}{\langle \mathcal{C}_1, \ldots, \mathcal{C}_n \rangle \xrightarrow{\delta} \langle \mathcal{C}'_1, \ldots, \mathcal{C}'_n \rangle}$$

   where for all $j \in [1, n]$, $\mathcal{C}'_j = \langle s, \nu + \delta \rangle$ if $\mathcal{C}_j = \langle s, \nu \rangle$, for any s

2. $$\frac{\mathcal{F}^{sync}(\sigma) = J, \quad \forall j \in J.\ \left(\mathcal{C}_j = (q_j, \nu_j),\ (q_j, \psi^i_j, \gamma^i_j, \sigma, \psi^c_j, \gamma^c_j, q'_j) \in \mathcal{E}_j,\ \nu_j \models \psi^i_j\right)}{\langle \mathcal{C}_1, \ldots, \mathcal{C}_n \rangle \xrightarrow{\sigma\uparrow} \langle \mathcal{C}'_1, \ldots, \mathcal{C}'_n \rangle}$$

   where for all $j \in [1, n]$, $\mathcal{C}'_j = \begin{cases} \mathcal{C}_j & \text{if } j \notin J \\ (\overrightarrow{\sigma}_{(\psi^c_j, \gamma^c_j, q'_j)}, \nu_j \backslash \gamma^i_j) & \text{if } j \in J \end{cases}$

3. $$\frac{\mathcal{F}^{sync}(\sigma) = J, \quad \forall j \in J.\mathcal{C}_j = (\overrightarrow{\sigma}_{(\psi^c_j, \gamma^c_j, q_j)}, \nu_j),\ \nu_j \models \psi^c_j}{\langle \mathcal{C}_1, \ldots, \mathcal{C}_n \rangle \xrightarrow{\sigma\downarrow} \langle \mathcal{C}'_1, \ldots, \mathcal{C}'_n \rangle}$$

where for all $j \in [1, n]$, $\mathcal{C}'_j = \begin{cases} \mathcal{C}_j & \text{if } j \notin J \\ (q_j, \nu_j \backslash \gamma^c_j) & \text{if } j \in J \end{cases}$

Rule 1. represents the case in which all the automata stay idle while the time passes. Rule 2. describes the situation in which a set of automata initiate, at the same time, an action σ. This set is exactly the set of the automata which have σ in their alphabet. If $\mathcal{F}^{sync}(\sigma)$ is a singleton, only one automaton proceeds. Rule 3. describes the case in which a set of automata in parallel complete an action σ.

Let us remark that the notions of run and acceptance are analogous to the ones for timed automata with non-instantaneous transitions, the only difference being that at least a repeated state for each automaton must be infinitely repeated in a run.

5 An example of specification

We consider the example of an automatic controller which opens and closes a gate at a railroad crossing. This example was originally presented in [18], and it was used in [7] to show the specification and verification capabilities of timed automata. For simplicity we assume that one unit of time corresponds to a minute.

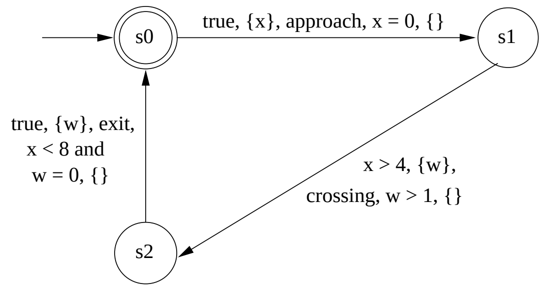

Fig. 4. Train

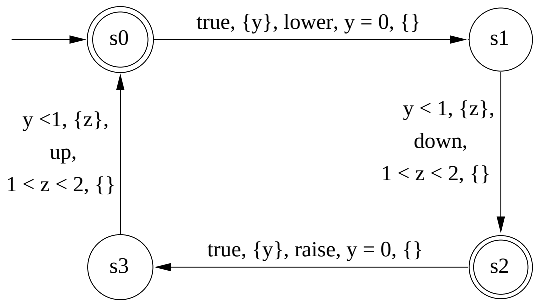

Fig. 5. Gate

The automaton modeling the train is shown in Figure 4. The automaton starts in state s_0. When approaching the railroad crossing, the train sends a (instantaneous) signal, *approach*, to the controller. Note that the signal *approach* belongs to both the alphabet of the train and of the controller, thus the two automata must synchronize on it. The train sends the signal *approach* at least four minutes before it enters the crossing. The train takes at least one minute to pass through the railroad crossing (action *crossing*). Note that, because this action is non-instantaneous, in [7] it was modeled by two actions, *in* and *out*, simulating the initiation and completion of it. After the signal *approach* is sent to the controller, the signal *exit* is sent within 8 minutes.

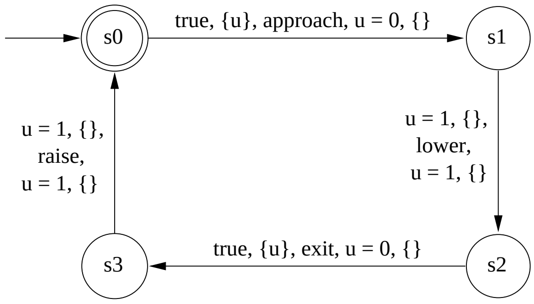

Fig. 6. Controller

The gate is modeled by the automaton in Figure 5. The synchronization with the controller is ensured by the instantaneous signals *lower* and *raise*. When the gate receives the signal *lower*, it starts to close the gate within one minute, and the closing action takes between one and two minutes. The gate responds to the signal *raise* by opening the gate with the same delays.

Finally, Figure 6 shows the controller. When the controller receives the signal *approach* from the train, it sends, exactly after one minute, the signal for closing the gate. With the same delay it sends the signal *raise* after receiving the signal *exit*. Note that all the signals are forced to be instantaneous by asking for the value of the clocks to be the same at their initiation and completion.

The whole system is obtained by the parallel composition of the three automata. Note that, in this composition, the states should be renamed to obtain disjointness among the automata.

It is important to remark that, by a suitable translation, the above system can be represented by a timed automaton with ϵ edges. Although the automaton obtained in this way does not retain the clarity of the above specification, it can be used to apply the verification techniques developed for timed automata [3, 10, 16, 19].

We would like to remark that our specification allows us to state properties which better describe real constraints than timed automata. As an example, let us quote a property which was checked on the train/gate system in [7]. Such a property is a safety one: "Whenever the train is inside the gate, the gate should be closed". Using parallel timed automata with non-instantaneous actions we can state finer properties. For example we could be interested in verifying the following two ones:

1. "Whenever the train approaches the gate, the gate closes, and when the train initiates to cross the gate, the action of closing it should be completed",
2. "Every time, after the train crosses the gate, the gate must open, and the action of opening the gate should be initiated only when the train has completed the action of crossing it".

Both the properties could be expressed as conditions on the language accepted by the parallel composition ($Train \parallel Gate \parallel Controller$) by specifying that, for property 1., the selected accepted words should refer to the initiation of *crossing* and to the completion of *down*, while for property 2. they should refer to the completion of *crossing* and to the initiation of *up*.

References

1. Aceto, L., Bouyer, P., Burgueño, A. and Guldstrand Larsen, K. The Power of Reachability Testing for Timed Automata. Proc. Foundations Software Technology and Theoretical Computer Science, Springer LNCS 1530, 245–256, 1998.
2. Aceto, L., Burgueño, A. and Guldstrand Larsen, K. Model Checking via Reachability Testing for Timed Automata. Proc. TACAS, Springer LNCS 1384, 263–280, 1998.
3. Alur, R., Courcoubetis, C. and Dill, D.L. Model-Checking in Dense Real-time. *Information and Computation*, 104, 2–34, 1993.
4. Alur, R., Courcoubetis, C., Halbwachs, N., Dill, D.L. and Wong-Toi, H. Minimization of Timed Transition Systems. Proc. CONCUR 1992, Springer LNCS 630, 340–354, 1992
5. Alur, R., Courcoubetis, C. and Henzinger, T.A. The Observational Power of Clocks. Proc. CONCUR 1994, Springer LNCS 836, 162–177, 1994.
6. Alur, R. and Dill, D.L. Automata for Modelin Real-time Systems. Proc. ICALP'90, Springer LNCS 443, 322–335, 1990.
7. Alur, R. and Dill, D.L. A Theory of Timed Automata. *Theoretical Computer Science*, 126, 183–235, 1994.
8. Alur, R., Fix, L. and Henzinger, T.A. Event-Clock Automata: A Determinizable Class of Timed Automata. *Theoretical Computer Science*, 211, 253-273 (1999).
9. Alur, R. and Henzinger, T.A. Back to the Future: Towards a Theory of Timed Regular Languages. Proc. FOCS 1992, 177–186, 1992.
10. Alur, R. and Henzinger, T.A. A Really Temporal Logic. *Journal of ACM*, 41, 181–204, 1994.
11. Alur, R., Itai, A., Kurshan, R.P. and Yannakakis. M. Timing Verification by Successive Approximation. *Information and Computation*, 118, 142–157,1995.
12. Bérard, B., Petit, A., Diekert, V. and Gastin P. Characterization of the Expressive Power of Silent Transitions in Timed Automata. *Fundamenta Informaticae*, 36, 145–182, 1998.
13. Choffrut ,C. and Goldwurm, M. Timed Automata with periodic Clock Constraint. Int. Report 225-98, 1998.
14. Henzinger, T.A. and Kopke, P.W. Verification Methods for the Divergent Runs of Clock Systems. Proc. Formal Techniques in Real-Time and Fault-Tolerant Systems, Springer LNCS 863, 351–372, 1994.
15. Henzinger, T.A., Manna, Z. and Pnueli, A. Temporal Proof Methodologies for Timed Transition Systems. *Information and Computation*, 112, 273–337, (1994).
16. Henzinger, T.A., Nicollin, X., Sifakis, J. and Yovine, S. Symbolic Model Checking for Real-Time Systems. *Information and Computation*, 111, 193–244, 1994.
17. Jahanian, F. and Mok, A.K. A Graph-Theoretic Approach for Timing Analysis and its Implementation. *IEEE Transactions on Computers*, 36, 961-975, 1987.
18. Leveson, N. and Stolzy, J. Analyzing Safety and fault Tolerance using Timed Petri Nets. Proc. Theory and Practice of Software Development, Springer LNCS 186, 339–355, 1985.

19. Yovine, S. Model Checking Timed Automata. Lectures on Embedded Systems, Springer LNCS 1494, 114–152, 1996.