Bridging the gap between Timed Automata and Bounded Time Petri Nets

Bernard Berthomieu, Florent Peres, and François Vernadat

LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse, France
fax: +33 (0)5.61.33.64.11 tel: +33 (0)5.61.33.63.63
e-mail:{Bernard.Berthomieu|Florent.Peres|Francois.Vernadat}@laas.fr

Abstract. Several recent papers investigate the relative expressiveness of Timed Automata and Time Petri Nets, two widespread models for realtime systems. It has been shown notably that Timed Automata and Bounded Time Petri Nets are equally expressive in terms of timed language acceptance, but that Timed Automata are strictly more expressive in terms of weak timed bisimilarity. This paper compares Timed Automata with Bounded Time Petri Nets extended with static Priorities, and shows that two large subsets of these models are equally expressive in terms of weak timed bisimilarity.
Keywords: Time Petri nets, priorities, Timed Automata, weak timed bisimilarity, real-time systems modeling and verification.

1 Introduction

Among the many models proposed for the specification and verification of real time systems, two are prominent: Time Petri nets and Timed Automata.

Time Petri nets (TPN) [15] extend Petri nets with temporal intervals associated with transitions, specifying firing delay ranges for the transitions. Assuming transition t became last enabled at time θ, and the end-points of its time interval are α and β, then t cannot fire earlier than time $\theta + \alpha$ and must fire no later than $\theta + \beta$, unless disabled by firing some other transition. Firing a transition takes no time. Many other Petri net based models with time extensions have been proposed, but none reaches the acceptance of Time Petri nets. Availability of effective analysis methods, prompted by [5], certainly contributed to their widespread use, together with their ability to cope with a wide variety of modeling problems for realtime systems.

Timed Automata (TA) [2] extend finite automata with clocks, guards, resets and a product operation. Transitions are guarded by boolean conditions on clock values. When taken, they may emit a label and perform resets of some clocks. All clocks progress synchronously as time elapses. Some versions of timed automata support progress annotations as location invariants limiting elapsing of time when at that location, urgency requirements, or transition deadlines. Timed automata are convenient for modeling a large class of realtime problems. They prompted a considerable amount of research work and benefit from a rich theory.

These two models, as well as their analysis techniques, were developed independently for years, though they bear strong relationships. State space abstractions for TPN's preserving various classes of properties can be computed in terms of so-called state classes [5] [4] [6]. State classes represent sets of states by a marking and a polyhedron capturing temporal information. State space abstractions for (Networks of) Timed Automata are based upon geometric regions characterizing sets of states by a location of the underlying automaton and a convex set capturing temporal information. In both cases, the convex sets can be represented by difference systems, or DBM's.

In spite of many technical resemblances and their overlapping application domains, few material was available until recently comparing expressiveness of these two models. A number of recent works finally addressed the issue. [11] translated a subclass of TA's into TPN's, preserving timed language acceptance. Later, [9] proposed a structural encoding of TPN's into TA's, improving an earlier semantics based encoding in [14]. [3] proves than TPN's and TA's are equivalent w.r.t. timed language acceptance, but that TA's are strictly more expressive in terms of timed bisimilarity, they also discuss the subclass of TA's weakly timed bisimilar with some TPN.

In this article, we first extend Time Petri nets with static priorities. In TPN's with Priorities ($PrTPN$'s for short), a transition is not allowed to fire if some transition with higher priority is firable at the same instant. Such priorities have many applications in realtime systems, in scheduling, arbitration, synchronization, and others problems. We then develop an encoding of Timed Automata (without progress requirements) into $PrTPN$'s, preserving weak timed bisimilarity. Next, we extend TA's with invariants and show that TA's with invariants built from $\{\leq, \wedge\}$ can be encoded into $PrTPN$'s. Finally, extending the encoding of [9] of TPN's into TA's, we show that TA's with invariants built from $\{\leq, \wedge\}$ are equally expressive than $PrTPN$'s with unbounded or right-closed intervals. Some corollaries follow that extend available equivalence results between TA's and TPN's (without priorities).

The paper is organized as follows. Section 2 recalls the essentials about timed transition systems (TTS), the common semantic domain for TA's and $PrTPN$'s. Section 3 reviews the terminology of Timed automata and their semantics. Section 4 introduces Time Petri nets with Priorities, and compares their expressiveness with that of TPN's. Section 5 explains how to encode Timed Automata (without invariants) into weakly timed bisimilar $PrTPN$'s. Section 6 discusses progress requirements, extends the encoding to TA's with invariants built from $\{\leq, \wedge\}$, and derives a number of ordering or equivalence results. Finally, Section 7 discusses some consequences, side issues, and prospective work.

2 Timed Transition Systems

The semantics of Timed Automata (TA) and Time Petri Nets ($PrTPN$) will be given in terms of Timed Transition Systems (TTS), as described in e.g. [13]. We review here their terminology and some key concepts used in the next sections.

Timed Transition Systems : $\mathbf{R}^+$ is the set of nonnegative reals. A *Timed Transition System* is a structure $\langle Q, q^0, \Sigma \cup \{\epsilon\}, \rightarrow \rangle$ where:

- Q is a set of *states*
- $q^0 \in Q$ is the *initial state*
- Σ is a finite set of *actions* not containing the *silent* action ϵ
- $\rightarrow \subseteq Q \times (\Sigma \cup \{\epsilon\} \cup \mathbf{R}^+) \times Q$ is the *transition* relation.

$(q, a, q') \in \rightarrow$ is also written $q \xrightarrow{a} q'$. Σ^ϵ abbreviates $\Sigma \cup \{\epsilon\}$. The transitions belonging to $Q \times \Sigma^\epsilon \times Q$ are called *discrete* transitions, those from $Q \times \mathbf{R}^+ \times Q$ are the *continuous* transitions. Continuous transitions are typically required to obey the following properties $(\forall d, d', d'' \in \mathbf{R}^+)$:

- *0-delay*: $q \xrightarrow{0} q' \Leftrightarrow q = q'$
- *additivity*: $q \xrightarrow{d} q' \wedge q' \xrightarrow{d'} q'' \Rightarrow q \xrightarrow{d+d'} q''$
- *continuity*: $q \xrightarrow{d+d'} q' \Rightarrow (\exists q'')(q \xrightarrow{d} q'' \wedge q'' \xrightarrow{d'} q')$
- *time-determinism*: $q \xrightarrow{d} q' \wedge q \xrightarrow{d} q'' \Rightarrow q' = q''$

Product of TTS : Let $S_1 = \langle Q_1, q_1^0, \Sigma_1^\epsilon, \rightarrow_1 \rangle$ and $S_2 = \langle Q_2, q_2^0, \Sigma_2^\epsilon, \rightarrow_2 \rangle$ be two TTS. We assume that every action of Σ_i labels some transition of S_i. The product of S_1 by S_2 is the TTS $S_1 \parallel S_2 = \langle Q_1 \times Q_2, q_1^0 \parallel q_2^0, \Sigma_1^\epsilon \cup \Sigma_2^\epsilon, \rightarrow \rangle$ where $\rightarrow$ is the smallest relation obeying the following rules $(a \in \Sigma_1 \cup \Sigma_2 \cup \{\epsilon\} \cup \mathbf{R}^+)$:

$$\frac{q_1 \xrightarrow{a}_1 q_1'}{q_1 \parallel q_2 \xrightarrow{a} q_1' \parallel q_2} \; (a \in \Sigma_1^\epsilon \backslash \Sigma_2) \qquad \frac{q_2 \xrightarrow{a}_2 q_2'}{q_1 \parallel q_2 \xrightarrow{a} q_1 \parallel q_2'} \; (a \in \Sigma_2^\epsilon \backslash \Sigma_1)$$

$$\frac{q_1 \xrightarrow{a}_1 q_1' \quad q_2 \xrightarrow{a}_2 q_2'}{q_1 \parallel q_2 \xrightarrow{a} q_1' \parallel q_2'} \; (a \neq \epsilon)$$

Timed Bisimilarity : Let $S_1 = \langle Q_1, q_1^0, \Sigma_1^\epsilon, \rightarrow_1 \rangle$ and $S_2 = \langle Q_2, q_2^0, \Sigma_2^\epsilon, \rightarrow_2 \rangle$ be two TTS and $\sim \subseteq Q_1 \times Q_2$. Then S_1 and S_2 are *strongly timed bisimilar* iff $q_1^0 \sim q_2^0$ and, whenever $q_1 \sim q_2$ and $a \in \Sigma_1^\epsilon \cup \Sigma_2^\epsilon \cup \mathbf{R}^+$:

(1) $q_1 \xrightarrow{a}_1 q_1' \Rightarrow (\exists q_2')(q_2 \xrightarrow{a}_2 q_2' \wedge q_1' \sim q_2')$
(2) $q_2 \xrightarrow{a}_2 q_2' \Rightarrow (\exists q_1')(q_1 \xrightarrow{a}_1 q_1' \wedge q_1' \sim q_2')$

Strong timed bisimilarity is often too strong a requirement. A coarser equivalence relation, hiding silent transitions, is obtained from relation $\overset{a}{\Longrightarrow}$, defined from $\xrightarrow{a}$ as follows $(a \in \Sigma \cup \mathbf{R}^+,\ d \in \mathbf{R}^+)$:

$$\frac{q \xrightarrow{a} q'}{q \overset{a}{\Longrightarrow} q'} \quad \frac{q \overset{a}{\Longrightarrow} q' \quad q' \xrightarrow{\epsilon} q''}{q \overset{a}{\Longrightarrow} q''} \quad \frac{q \xrightarrow{\epsilon} q' \quad q' \overset{a}{\Longrightarrow} q''}{q \overset{a}{\Longrightarrow} q''} \quad \frac{q \overset{d}{\Longrightarrow} q' \quad q' \overset{d'}{\Longrightarrow} q''}{q \overset{d+d'}{\Longrightarrow} q''}$$

Two timed transition systems are *weakly timed bisimilar* when conditions (1) and (2) above hold, with relations $\xrightarrow{a}_i$ replaced by $\overset{a}{\Longrightarrow}_i$ $(i \in \{1, 2\})$.

3 Timed Automata

$\mathbf{Q}^+$ is the set of nonnegative rationals. Given a finite set of clocks X, the set $\mathcal{C}(X)$ of clock constraints over X is defined by the grammar:

$$g ::= x\#c \mid g \wedge g \mid \texttt{true} \text{ where } x \in X,\ c \in \mathbf{Q}^+ \text{ and } \# \in \{\leq, <, \geq, >\}$$

Definition 1. *A* Timed Automaton *(TA), is a tuple $\langle Q, q^0, X, \Sigma^\epsilon, T\rangle$ in which:*

- Q *is a finite set of* locations
- $q^0 \in Q$ *is the* initial location
- X *is a finite set of clocks*
- Σ *is a finite set of actions, assumed not to contain the* silent *action* ϵ
- $T \subseteq Q \times (\mathcal{C}(X) \times \Sigma^\epsilon \times 2^X) \times Q$ *is a finite set of transitions, or edges.*

$(q, g, a, R, q') \in T$ may be written $q \xrightarrow{g,a,R} q'$. A *configuration* of a Timed Automata is a pair (q, v), where $q \in Q$ and v is a vector of clock values (one for each clock in X). $v[R := 0]$ denotes the vector in which the components corresponding to clocks in R are 0 and the others are as in v. $|E|$ is $card(E)$.

Definition 2. *The semantics of a Timed Automaton $\langle Q, q^0, X, \Sigma^\epsilon, T\rangle$ is the TTS $\langle S, s^0, \Sigma^\epsilon, \rightarrow\rangle$ where $S = Q \times (\mathbf{R}^+)^{|X|}$, $s^0 = (q^0, \overline{0})$, and $\rightarrow$ is defined by:*

- $(q, v) \xrightarrow{a} (q', v')$ *iff* $(\exists (q, g, a, R, q') \in T)(g(v) \wedge v' = v[R := 0])$
- $(q, v) \xrightarrow{d} (q, v + d)$ *iff* $d \in \mathbf{R}^+$

Product of timed automata : A product construction $||$ is defined for timed automata, allowing one to express complex realtime systems as synchronized components. A definition of this product can be found in e.g. [1]. That product construction is compositional in that the TTS denoted by a product of timed automata is the product of the TTS's respectively denoted by the components.

Progress requirements : Our definition of TA, taken from [2], does not include any means of enforcing progress. The need for enforcing progress has been recognized early and several solutions have been proposed to extend TA with such requirements, including location invariants, urgency and transition deadlines. Progress enforcement will be added to TA's and discussed in Section 6.

Priorities : We only consider here static priorities. The definition of TA's is extended with a partial irreflexive, asymmetric and transitive priority relation among transitions. The semantics is updated accordingly: a transition can only be taken when no transition with higher priority can be taken at the same instant.

Static priorities do not add expressiveness to TA's. If t_1, with guard g_1, has lower priority than t_2, with guard g_2, then it suffices to replace g_1 by $g_1 \wedge \neg g_2$. Negating a guard may introduce disjunctions, but these can be removed by distributing the components of the disjunction among several transitions, the transformation preserves strong timed bisimilarity. Composition and analysis of TA with priorities raise specific issues out of the scope of this paper.

4 Time Petri nets with priorities

$PrTPN$'s extend TPN's with a priority relation on transitions. Since we want to discuss bisimulations, we also add an alphabet of actions and a labeling function for transitions. $\mathbf{I}^+$ is the set of nonempty real intervals with nonnegative rational end-points. For $i \in \mathbf{I}^+$, $\downarrow i$ denotes its left end-point, and $\uparrow i$ its right end-point (if i bounded) or ∞. For any $\theta \in \mathbf{R}^+$, $i \dot{-} \theta$ denotes the interval $\{x - \theta | x \in i \wedge x \geq \theta\}$.

Definition 3. *A* Time Petri Net with Priorities *(or $PrTPN$ for short) is a tuple $\langle P, T, \mathbf{Pre}, \mathbf{Post}, m^0, Is, Pr, \Sigma^\epsilon, L\rangle$ in which:*

- *$\langle P, T, \mathbf{Pre}, \mathbf{Post}, m^0\rangle$ is a Petri net, with places P, transitions T, initial marking $m^0 : P \rightarrow \mathbf{N}^+$ and $\mathbf{Pre}, \mathbf{Post} : T \rightarrow P \rightarrow \mathbf{N}^+$,*
- *$Is : T \rightarrow \mathbf{I}^+$ is a function called the* Static Interval *function,*
- *$Pr \subseteq T \times T$ is the Priority relation, irreflexive, asymmetric and transitive,*
- *Σ is a finite set of* Actions, *or* Labels, *not containing the* Silent *action ϵ,*
- *$L : T \rightarrow \Sigma^\epsilon$ is a function called the* Labeling *function.*

For $f, g : P \rightarrow \mathbf{N}^+$, $f \geq g$ means $(\forall p \in P)(f(p) \geq g(p))$ and $f\{+|-\}g$ maps $f(p)\{+|-\}g(p)$ with every p. A marking is a function $m : P \rightarrow \mathbf{N}^+$, $t \in T$ is *enabled* at m iff $m \geq \mathbf{Pre}(t)$, $\mathcal{E}_N(m)$ denotes the set of transitions enabled at m in net N. $(t_1, t_2) \in Pr$ is written $t_1 \succ t_2$ or $t_2 \prec t_1$ (t_1 has priority over t_2).

Definition 4. *The semantics of $PrTPN$ $\langle P, T, \mathbf{Pre}, \mathbf{Post}, m^0, Is, Pr, \Sigma, L\rangle$ is the timed transition system $\langle S, (m^0, Is^0), \Sigma, \rightarrow\rangle$ where:*

- *Is^0 is function Is restricted to the transitions enabled at m^0*
- *the states of S are pairs (m, I) in which m is a marking and $I : T \rightarrow \mathbf{I}^+$ associates a time interval with every transition enabled at m,*
- *$(m, I) \xrightarrow{L(t)} (m', I')$ iff $t \in T$ and*
  1. *$m \geq \mathbf{Pre}(t)$*
  2. *$0 \in I(t)$*
  3. *$(\forall k \in T)(m \geq \mathbf{Pre}(k) \wedge 0 \in I(k) \Rightarrow \neg(k \succ t))$*
  4. *$m' = m - \mathbf{Pre}(t) + \mathbf{Post}(t)$*
  5. *$(\forall k \in T)(m' \geq \mathbf{Pre}(k) \Rightarrow$*
     $I'(k) = \ \mathbf{if}\ \ k \neq t\ \wedge\ m - \mathbf{Pre}(t) \geq \mathbf{Pre}(k)\ \ \mathbf{then}\ \ I(k)\ \ \mathbf{else}\ \ Is(k))$
- *$(m, I) \xrightarrow{d} (m, I')$ iff $(\forall k \in T)(m \geq \mathbf{Pre}(k) \Rightarrow d \leq \uparrow I(k) \wedge I'(k) = I(k) \dot{-} d)$*

Transition t may fire from (m, I) if t is enabled at m, firable instantly, and no transition with higher priority satisfies these conditions. In the target state, the transitions that remained enabled while t fired (t excluded) retain their intervals, the others are associated with their static intervals. A continuous transition by d is possible iff d is not larger than any $\uparrow I(t)$.

Boundedness : A PN is *bounded* if the marking of each place is bounded, boundedness implies finiteness of the set of reachable markings. Boundedness is undecidable for TPN's, and thus for $PrTPN$'s, but there are a number of decidable sufficient conditions for this property [5]. All nets considered in this paper are assumed bounded.

Product of PN, TPN, $PrTPN$: A product construction for labeled Petri nets has been used in many works, a definition appeared e.g. in [12]. It can be seen as a generalization to concurrent systems of the product of automata.

Definition 5. *Given a PN* $\langle P, T, \mathbf{Pre}, \mathbf{Post}, m^0\rangle$ *and* $E \subseteq T$*, the* product *of the transitions in* E *is the transition* t *such that, for any* $p \in P$*:*

$$\mathbf{Pre}(t)(p) = \sum_{k \in E} \mathbf{Pre}(k)(p) \quad \textit{and} \quad \mathbf{Post}(t)(p) = \sum_{k \in E} \mathbf{Post}(k)(p)$$

Definition 6. *Consider two labeled PN's not sharing any place or transition* $N_1 = \langle P_1, T_1, \mathbf{Pre}_1, \mathbf{Post}_1, m_1^0, \Sigma_1^\epsilon, L_1\rangle$, $N_2 = \langle P_2, T_2, \mathbf{Pre}_2, \mathbf{Post}_2, m_2^0, \Sigma_2^\epsilon, L_2\rangle$. *The product* $N = N_1||N_2$ *of* N_1 *and* N_2 *can be built as follows:*

- *Start with* N *made of the union of nets* N_1 *and* N_2*, after having removed from each the transitions labeled on* $\Sigma_1 \cap \Sigma_2$*,*
- *Then, for each pair* $(t_1, t_2) \in T_1 \times T_2$ *such that* $L_1(t_1) = L_2(t_2) \neq \epsilon$*, add to* N *a transition defined as the product of* t_1 *and* t_2*, inheriting their label.*

Product $||$ can be extended to $PrTPN$'s by specifying the static interval of a synchronized pair of transitions (t_1, t_2) as $Is_1(t_1) \cap Is_2(t_2)$, assuming it nonempty, and defining Pr from the local priority relations as follows:

- Let $R = \{(x, y) \in T \times T | (\exists (t, t') \in Pr_1 \cup Pr_2)(x \in S(t) \wedge y \in S(t')\}$ where, for any $t \in T_1 \cup T_2$, $S(t)$ is either $\{t\}$ if t is not synchronized, or the set of transitions of T obtained as a product involving t otherwise (see Def. 6),
- Pr is the transitive closure of R, assuming it asymmetric.

Product $||$ is commutative and associative. For Petri nets, it is compositional: the transition system denoted by $N_1||N_2$ is the product of those denoted by N_1 and N_2. For TPN's or $PrTPN$'s, compositionality does not hold in general, but it holds when all synchronized transitions have interval $[0, \infty[$ and no pair of non-synchronized transitions is in the priority relation. The latter condition implies that Pr exactly coincides with relation R above. Compositionality in this special case is expressed by Theorem 1, its proof is given in the Appendix.

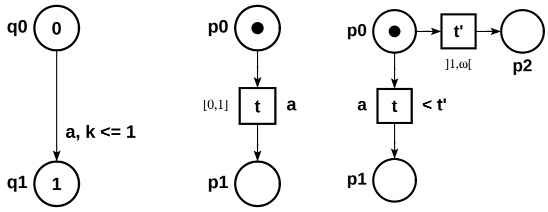
Theorem 1 (Restricted compositionality of product $||$ for $PrTPN$'s).

Let $\lceil n \rceil$ *be the* TTS *associated with* $PrTPN$ n,
$N_1 = \langle P_1, T_1, \mathbf{Pre}_1, \mathbf{Post}_1, m_1^0, Is_1, Pr_1, \Sigma_1^\epsilon, L_1\rangle$
$N_2 = \langle P_2, T_2, \mathbf{Pre}_2, \mathbf{Post}_2, m_2^0, Is_2, Pr_2, \Sigma_2^\epsilon, L_2\rangle$
be two labeled $PrTPN$*'s with disjoint sets of places and transitions,*
$N = \langle P, T, \mathbf{Pre}, \mathbf{Post}, m^0, Is, Pr, \Sigma^\epsilon, L\rangle$ *be their product by* $||$,
$T_{i\backslash j}$ *be the subset of transitions of* T_i *labeled over* $\Sigma_i^\epsilon \setminus \Sigma_j$
$T_{1\times 2} = T \backslash (T_{1\backslash 2} \cup T_{2\backslash 1})$
Then $(\forall t \in T_{1\backslash 2})(Is_1(t) = [0, \infty[) \wedge (\forall t \in T_{2\backslash 1})(Is_2(t) = [0, \infty[)$
$\wedge$ $Pr \cap (T_{1\backslash 2} \times T_{2\backslash 1}) = Pr \cap (T_{2\backslash 1} \times T_{1\backslash 2}) = \emptyset$
$\Rightarrow \lceil N_1 \,||\, N_2 \rceil = \lceil N_1 \rceil \,||\, \lceil N_2 \rceil$

The following Theorem will be needed in Section 5, it follows from Def. 4:

Theorem 2. *Assume PrTPN N has two transitions t_1 and t_2 unrelated by the priority relation, both with interval $[0, \infty[$, and not sharing any input place. Then adding to N the transition t defined as the product of t_1 and t_2 does not change its state space and t can fire exactly at the times at which both t_1 and t_2 can.*

The power of priorities : Contrarily to TA's, priorities add expressiveness to bounded TPN's. To illustrate this, consider the TA and nets in Figure 1.



(a) (b) (c)

Fig. 1. Expressiveness of priorities in TPN's

As mentioned in [3], no TPN is, even weakly, timed bisimilar with TA 1(a). In particular, the TPN 1(b) is not: when at location q_0 in the TA, time can elapse of an arbitrary amount, while time cannot progress beyond 1 in TPN 1(b). Consider now the $PrTPN$ 1(c). Priorities are specified by transition annotations, we have here $t \prec t'$. Transition t' is silent and firable at any time greater than 1, transition t bears label a and interval $[0, \infty[$. Transition t may fire at any time less than or equal to 1, but not later, since t' is then firable and it has priority over t. Indeed, $PrTPN$ 1(c) is weakly timed bisimilar with TA 1(a).

Priorities do not forbid time to elapse, but enrich firing constraints for transitions. It is shown in the next Section that the above trick generalizes to any TA. Addition of progress requirements will be addressed in Section 6.

5 Encoding guards

5.1 Notations

Let $\mathcal{A} = \langle Q, q^0, X, \Sigma^\epsilon, T \rangle$ be some TA. Without loss of generality, it is assumed that every clock in X is involved in some guard, that every clock which is reset in some transition is used in some guard, and that $\mathcal{A}$ is not expressed as a product.

- For each transition $t \in T$, let:
  - σ_t be the action of $\Sigma \cup \{\epsilon\}$ associated with t,
  - X_t be the set of clocks involved in transition t (in its guard or reset),
  - E_t^k, for $k \in X_t$, be the set of atomic guards in the guard of t involving clock k, augmented with $(k := 0)$ if t resets k,

- $E_t = \sigma_t \cup \bigcup_{k \in X_t} \{E_t^k\}$, E_t holds action σ_t and all sets E_t^k for $k \in X_t$.

– For each clock $k \in X$, let G_k be the set of atomic guards in $\mathcal{A}$ involving k.

5.2 Encoding atomic clock guards and resets

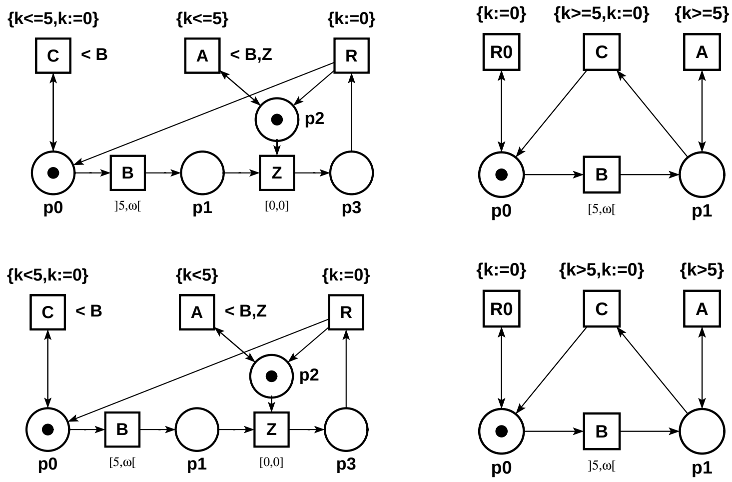


Fig. 2. $PrTPN$'s encoding atomic guards and resets for clock k and constant $c = 5$

Each atomic guard will be modeled by a $PrTPN$, Figure 2 shows the four required models for clock k and constant $c = 5$. The upper-left (resp. lower-left) net models $k \leq 5$ (resp. $k < 5$), the upper-right (resp. lower-right) net models $k \geq 5$ (resp. $k > 5$). Priorities are specified by transition annotations, e.g. in the upper-left net, we have $C \prec B$, $A \prec B$ and $A \prec Z$. The transitions are either unlabeled (implicitly labeled with silent action ϵ), or carry a label made of a clock constraint and/or a reset.

Theorem 3. *Let k be the time elapsed since the initial markings of the nets in Figure 2 were last established, then, for each of these nets:*

- *the transitions whose label includes a condition on k are firable exactly at the times at which that condition holds,*
- *all transitions whose label includes $k := 0$ restore the initial state of the net,*
- *from any state a sequence of duration 0 firing a transition whose label includes $k := 0$ is firable*

Proof. For net $k \leq 5$: If $k \leq 5$, C and A may fire, firing C restores the initial state, firing A preserves the current state. At any $k > 5$, B has priority over A and C, B may be arbitrarily delayed. After B fired, A and Z are enabled but Z has higher priority, and time cannot elapse since Z carries $[0, 0]$. After Z fired, only R may fire, restoring the initial state. The proof is similar for net $k < 5$. The nets for $k \geq 5$ and $k > 5$ are self-explanatory, they do not use priorities.

5.3 Encoding a clock

For each $k \in X$, let N_k be the net built as follows:

1. Assume $\{c_1, \ldots, c_n\}$ is the set of nets encoding the guards in G_k. Let K be the product $(c_1\backslash F \mid\mid \ldots \mid\mid c_n\backslash F)\backslash H$, with relabelings F and H as follows:
   - F relabels any transition whose label includes $k := 0$ with ρ (ρ new),
   - H relabels any t obtained from a product of transitions by the union of their labels in nets c_i.
2. Next, starting with $N_k = K$, add to N_k, for each $E \subseteq G_k$ with $card(E) > 1$, a transition labeled E defined as the product (Def. 6) of all transitions of K with their label intersecting E.

The first step synchronizes resets among all the nets encoding atomic guards for clock k. Step 2 adds transitions checking all possible conjunctions of atomic guards for k (without reset). Net N_k has as transitions:

- The transitions not labeled (internal) in the component nets,
- For each nonempty $E \subseteq G_k$, a transition labeled with set E,
- For each (possibly empty) $E \subseteq G_k$, a transition labeled $E \cup \{k := 0\}$.

Theorem 4. *Let k be the time elapsed since the initial marking of net N_k was last established, then:*

- *the transitions whose label includes a condition (atomic or compound) are firable exactly at the times at which that condition holds,*
- *all transitions whose label includes $k := 0$ restore the initial state of the net,*
- *from any state a sequence of duration 0 firing a transition whose label includes $k := 0$ is firable*

Proof. Follows from Theorems 1, 2 and 3, and definitions of relabelings F and H. Note that all assumptions required by Theorem 1 hold.

5.4 Encoding the timed automaton $\mathcal{A}$

The $PrTPN$ $\mathcal{N}$ encoding $\mathcal{A} = \langle Q, q^0, X, \Sigma^\epsilon, T\rangle$ is obtained as follows.

1. Let N_A be the net built as follows:
   - For each $q \in Q$ add a new place to N_A, and mark the place encoding q^0,
   - For each transition $q \xrightarrow{t} q'$ of $\mathcal{A}$, add to N_A a transition between the places encoding q and q', labeled by E_t (as computed in Section 5.1).
2. Next, let N_K be the net built as follows:
   - Start with $N_K = ||_{k \in X}\ N_k$ where N_k encodes clock k (see Section 5.3),
   - Then, for each E_t, add to N_K a transition labeled E_t defined as the product of all transitions of N_K with their label belonging to E_t,
   - Remove all labeled transitions of N_K whose label is not in any E_t,
3. Finally, let $\mathcal{N} = N_A \mid\mid N_K$ and relabel each labeled transition by the action from Σ^ϵ belonging to its label.

Intuitively, N_A after step 1 is a copy of the underlying automaton of $\mathcal{A}$, with each transition labeled by the set of atomic guards of the corresponding transition of A, augmented with resets, and partitioned per clock. N_K at step 2 is the product of the nets encoding clocks, augmented with transitions checking all guards of $\mathcal{A}$ and performing resets if needed. The transitions removed at step 2 correspond to combinations of atomic guards not used in $\mathcal{A}$ or to transitions resetting clocks which are never reset in $\mathcal{A}$. Step 3 restores the labeling of $\mathcal{A}$.

Theorem 5. *The above TA $\mathcal{A}$ and $PrTPN$ $\mathcal{N}$ are weakly timed bisimilar.*

Proof. Follows again from the properties of the operations used to built the net and of those the nets being composed. The nets N_k encoding clocks do not share any label. After step 2, and from Theorems 2 and 4, the labeled transitions of N_K are firable exactly when the conjunction of the constraints they are labeled with hold. At step 3, guard checks and resets are ordered like in $\mathcal{A}$.

Theorem 6. *Any TA can be encoded into a $PrTPN$ without right-open intervals, preserving weak timed bisimilarity, or:*

$TA \lesssim_W PrTPN$ with unbounded or right-closed intervals.

Proof. The encoding applied to $\mathcal{A}$ is applicable to any TA (without progress requirements). Further, it produces $PrTPN$'s without right-open intervals.

6 Encoding Time Automata Invariants

6.1 Expressing progress requirements

Enforcing progress conditions in Timed Automata is undoubtedly necessary for modeling systems with hard time constraints, but there is no consensus yet about how to express them. Progress requirements are most often expressed using location invariants or transition deadlines. Location invariants (as e.g. in [1]) are compositional but may introduce timelocks (sink states resulting from elapsing of time), transition deadlines [7] avoid timelocks but break compositionality [10]. We will concentrate here on invariants, leaving alternatives for further work.

Definition 7. *A TA with Invariants is a tuple $\langle Q, q^0, X, \Sigma^\epsilon, T, I\rangle$ in which:*

- *$\langle Q, q^0, X, \Sigma^\epsilon\rangle$ is a TA, as in Definition 1,*
- *$I : Q \to \mathcal{C}(X)$ maps a clock constraint with every location. These constraints are typically restricted to conjunctions of constraints of form $k \leq c$ or $k < c$.*

Definition 8. *The semantics of TA with invariants only differs by the rule concerning continuous transitions (see Definition 2), which is replaced by:*

- $(q, v) \xrightarrow{d} (q, v + d)$ *iff* $d \in \mathbf{R}^+ \wedge (\forall d')(0 \leq d' \leq d \Rightarrow I(q))$

Time may elapse at some location as long as the invariant at that location holds. Some variations of this rule are found in the literature, as well as further conditions related to progress and not addressed here, such as non-Zenoeness.

The class of TA with invariants as above will be noted $TA + \{\leq, <, \wedge\}$, $TA + \{\leq, \wedge\}$ is its subclass in which no invariant constraint is strict, TA is the class with no invariants.

6.2 Encoding TA+$\{\leq, \wedge\}$

Adapting the translation of invariants in [3], our encoding of TA in Section 5 can be extended to handle invariants with non strict constraints.

For encoding the effects of invariant $k_1 \leq c_1 \wedge \ldots \wedge k_n \leq c_n$ at location q_i we will monitor property $k_1 \geq c_1 \vee \ldots \vee k_n \geq c_n$ when the place p_i materializing location q_i in net N_A (see Section 5.4) is marked, and prevent time to progress as soon as the property holds.

As for guards, a $PrTPN$ will be associated with each atomic constraint to be monitored. Because of the above "no-delay" requirement, we cannot use for this the nets implementing guard check $k \geq c$ (Figure 2), we will use instead nets like the one in Figure 3 (this net could be used to check $k \geq 5$ guards as well, though). Note that the transition that checks $k \geq 5$ in that net carries a label distinguishing it from the transition realizing guard check $k \geq 5$. Of course, these monitoring $PrTPN$'s, for each clock k, must have their reset transitions synchronized with those of net N_k implementing clock k (see Section 5.3).

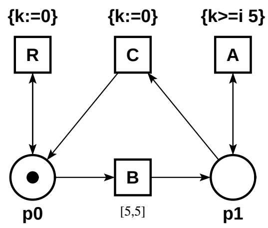


Fig. 3. $PrTPN$ encoding monitor for constraint $k \geq 5$

The encoding of invariants is sketched in Figure 4. For checking an atomic invariant, say $k \leq 5$, at location q_i, a loop will be added to place p_i in net N_A. The transition in this loop will be synchronized with the transition labeled $k \geq i$ 5 in the net monitoring $k \geq 5$, shown Figure 3. After synchronization, the transition will be made "urgent" by assigning it firing interval $[0, 0]$. Consequently, when p_i is marked, time elapsing is prevented as soon as $k = 5$.

For checking compound invariants, one such loop is added for each constraint to be monitored, as shown in Figure 4. The process is repeated for each location carrying an invariant. The constructions of Sections 5 are easily extended to encode invariants that way, details are omitted.

The above encoding does not extend to invariants with strict constraints, unfortunately (other solutions might possibly work, though). At that time, with the above encoding of invariants, we have TA+$\{\leq, \wedge\} \lesssim_{\mathcal{W}} PrTPN$. Observing that encoding such extended TA's yields $PrTPN$'s without right-open intervals, this result can be strengthened to:

Theorem 7. *TA+$\{\leq, \wedge\} \lesssim_{\mathcal{W}} PrTPN$ with unbounded or right-closed intervals*

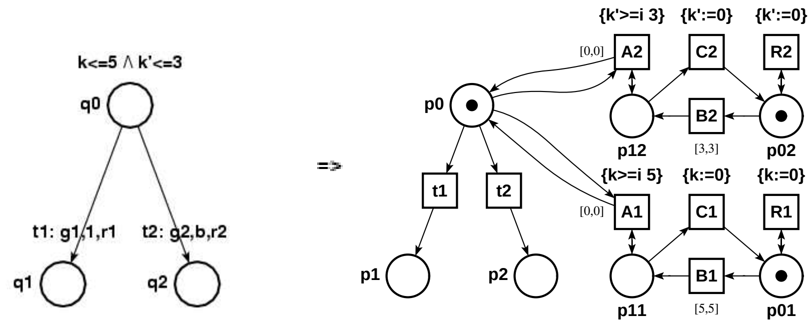


Fig. 4. Checking conjunctions of $k \leq c$ invariants

6.3 From $\boldsymbol{PrTPN}$ to $\boldsymbol{TA}$

Some equivalences, rather than orderings, can be obtained by adapting the translation of [9] of bounded TPN's (without priorities) into TA's.

[9] formalizes the idea that (bounded) TPN's can be encoded into TA's with invariants using one clock per transition of the TPN. Each transition is encoded into a timed automaton mimicking its "behavior": check for enabledness, removal of input tokens and addition of output tokens. The TPN is encoded as a composition of the automata modeling its transitions and of a supervisor TA sequencing the above operations for all transitions simultaneously. "earliest firing time" constraints of the TPN are encoded into guard constraints of form $k \geq c$ and "latest firing time" constraints into invariants of form $k \leq c$[1]. Though the method is defined for TPN's with closed or left-closed unbounded intervals, it is applicable unchanged to arbitrary TPN's.

The method can be adapted to encode $PrTPN$'s into TA's with priorities: It suffices to assign to the transitions labeled *?pre* in their encoding the priorities assigned to the corresponding transitions of the net. Now, as noted in Section 3, $PrTA \approx_{\mathcal{W}} TA$, hence, for Bounded $PrTPN$'s:

Theorem 8. $PrTPN \lesssim_{\mathcal{W}} TA + \{\leq, <, \wedge\}$.

Theorem 9. $TA{+}\{\leq, \wedge\} \approx_{\mathcal{W}} PrTPN$ *with right-closed or unbounded intervals*

Proof. From such restricted $PrTPN$'s, [9] (adapted for $PrTPN$'s) would yield TA's with invariants built from $\{\leq, \wedge\}$. From such TA's, our encoding yields $PrTPN$'s in which all transitions have unbounded or right-closed firing intervals.

As corollaries concerning Bounded TPN's, we obtain (proofs omitted):

Corollary 1. TA *with guards built from* $\{\geq, >, \wedge\} \approx_{\mathcal{W}} TPN$ *with unbounded intervals*

Corollary 2. $TA{+}\{\leq, \wedge\}$ *with guards built from* $\{\geq, >, \wedge\} \approx_{\mathcal{W}} TPN$ *with right-closed or unbounded intervals.*

[1] [9] adds these constraints to guards too, but this is not necessary.

7 Conclusion

Summary : The now available comparison results are shown in the Table below, for Bounded $PrTPN$'s and TPN's. These results hold for safe (1-bounded) nets too. They cannot be strengthened to strong bisimulation or preorders results, due to the necessary internal transitions in the TA translations of Section 5 and the TPN translations of [9]. No previous work compared TA's with $PrTPN$'s. For TPN's, Corollary 2 strengthens Corollaries 5 and 6 in [3].

	TA guards	invariants		TPN intervals	priorities
Theorem 7	$\leq < \geq > \wedge$	$\leq \wedge$	$\lesssim_{\mathcal{W}}$	$(]a\vert[a),(b]\vert\infty[)$	Y
Theorem 8	$\leq < \geq > \wedge$	$\leq < \wedge$	$\gtrsim_{\mathcal{W}}$	$(]a\vert[a),(b]\vert b[\vert\infty[)$	Y
Theorem 9	$\leq < \geq > \wedge$	$\leq \wedge$	$\approx_{\mathcal{W}}$	$(]a\vert[a),(b]\vert\infty[)$	Y
[9]	$\geq > \wedge$	$\leq < \wedge$	$\gtrsim_{\mathcal{W}}$	$(]a\vert[a),(b]\vert b[\vert\infty[)$	N
Corollary 2	$\geq > \wedge$	$\leq \wedge$	$\approx_{\mathcal{W}}$	$(]a\vert[a),(b]\vert\infty[)$	N
Corollary 1	$\geq > \wedge$	$\emptyset$	$\approx_{\mathcal{W}}$	$(]a\vert[a),\infty[$	N

These results improve understanding of the differences between two of the most widely used models of realtime systems. Theorem 9 suggests that, augmenting TPN's with priorities, these differences are small, if any.

They also promise sharing of analysis methods for these two models, at the time rather complementary: Analysis methods for TPN's mostly focused on time abstracting representations, preserving markings and LTL properties [5], states and LTL [6], or states and CTL (time abstracting bisimulations) [6] . Analysis methods for TA's focused more on model-checking of "timed" temporal properties such as those expressible in $TCTL$. Theorem 8 and [9] allow to use for $PrTPN$'s or TPN's analysis methods developed for TA's. Theorem 7 allows the converse for a large subclass of TA's, provided analysis methods for TPN's can be extended to $PrTPN$'s.

Further work : For easing proofs, the encoding of guards presented in Section 5 is rather brute force. It should be convenient in a number of cases, but it yields $PrTPN$'s with many more transitions than the TA, in general. For practical purposes, clock models could be made smaller. Also, *read-arcs*, a well known extension of Petri nets consisting of special arcs that test boolean conditions on markings but do not transfer tokens, would significantly compact encodings. Development of such improved encodings is left as further work.

To apply TPN-style analysis methods to TA's through the encodings explained, these methods should be extended to cope with priorities. Extending to $PrTPN$'s the methods building state space abstractions preserving LTL properties in [5, 6] should be straightforward, it resumes to take into account the extra constraints brought by priorities when developing the state class graph, these constraints are linear. Extending the methods building time abstracting bisimulations [6] is more subtle as continuous transitions cannot all be merged with discrete transitions anymore, but we are confident that they can be adapted.

Finally, encoding of timed automata with alternative proposals for progress requirements needs be investigated.

References

1. R. Alur. Timed automata. In *11th International Conference on Computer Aided Verification, CAV'99, Springer LNCS 1633*, pages 8–22, July 1999.
2. R. Alur and D.L. Dill. A theory of timed automata. *TCS*, 126:183–235, 1994.
3. B. Bérard, F. Cassez, S. Haddad, O. H. Roux, and D. Lime. Comparison of the Expressiveness of Timed Automata and Time Petri Nets. In *Formal Modeling and Analysis of Timed Systems (FORMATS'05), LNCS 3829*, pages 211–225, 2005.
4. B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using time Petri nets. *IEEE Trans. on Software Engineering*, 17(3):259–273, 1991.
5. B. Berthomieu and M. Menasche. An enumerative approach for analyzing time Petri nets. *IFIP Congress Series*, 9:41–46, 1983.
6. B. Berthomieu and F. Vernadat. State class constructions for branching analysis of time Petri nets. In *Proc. Tools and Algorithms for the Construction and Analysis of Systems, Springer LNCS 2619*, 2003.
7. S. Bornot, J. Sifakis, and S. Tripakis. Modeling urgency in timed systems. In *Compositionality: The Significant Difference – International Symposium COMPOS'97, Springer LNCS 1536*, pages 103–129, September 1997.
8. P. Bouyer, S. Haddad, and P-A. Reynier. Extended timed automata and time Petri nets. In *Proc. of 6th Int. Conf. on Application of Concurrency to System Design (ACSD'06)*, Turku, Finland, June 2006. IEEE Computer Society Press.
9. F. Cassez and O. H. Roux. Structural translation from time petri nets to timed automata. *Journal of Systems and Software*, 2006. forthcoming.
10. P. R. D'Argenio and B. Gebremichael. The coarsest congruence for timed automata with deadlines contained in bisimulation. In *CONCUR 2005 - Concurrency Theory, Springer LNCS 3653*, pages 125–140, August 2005.
11. S. Haar, L. Kaiser, F. Simonot-Lion, and J. Toussaint. Equivalence of Timed State Machines and safe Time Petri Nets. In *Proc. of the 6th Workshop on Discrete Events Systems (WODES'02)*, pages 119–124, Zaragoza, Spain, October 2002.
12. M. Hack. Petri net languages. TR 159, MIT, Cambridge, Mass., 1976.
13. K. G. Larsen, P. Pettersson, and W. Y. Yi. Model-checking for real-time systems. In *Fundamentals of Computation Theory, LNCS 965*, pages 62–88, August 1995.
14. D. Lime and O. H. Roux. State class timed automaton of a time Petri net. In *10th International Workshop on Petri Nets and Performance Models, (PNPM'03)*, pages 124–133, Urbana, USA, September 2003. IEEE Computer Society.
15. P. M. Merlin and D. J. Farber. Recoverability of communication protocols: Implications of a theoretical study. *IEEE Tr. Comm.*, 24(9):1036–1043, Sept. 1976.

A Proof of Theorem 1

Let $f[E]$ be the restriction of function f to its domain intersected with E, $f \oplus g$ be the function that behaves like f on the domain of f or like g otherwise, I^∞ be the function that associates interval $[0, \infty[$ with any transition.
It directly follows from Definition 4 that if some transition has static firing interval $[0, \infty[$, then its interval in any state of the net is $[0, \infty[$. Hence we have:

Lemma 1. (i) $(\forall (m, I) \in \lceil N_1 \parallel N_2 \rceil)(\forall t \in T_{1\times 2})(I(t) = [0, \infty[)$
(ii) $(\forall (m_i, I_i) \in \lceil N_i \rceil)(\forall t \in \mathcal{E}_N(m_i) \backslash T_{i\backslash j})(I_i(t) = [0, \infty[)$

Next, consider the mapping $\phi : \lceil N_1 \parallel N_2 \rceil \rightarrow \lceil N_1 \rceil \parallel \lceil N_2 \rceil$ defined by:

$$\begin{aligned}\phi(m, I) = &(m_1, I[T_{1\backslash 2}] \oplus I^\infty[\mathcal{E}_{N_1}(m_1)\backslash T_{1\backslash 2}])) \text{ where } m_1 = m[P_1] \\ &\parallel (m_2, I[T_{2\backslash 1}] \oplus I^\infty[\mathcal{E}_{N_2}(m_2)\backslash T_{2\backslash 1}])) \text{ where } m_2 = m[P_2]\end{aligned}$$

By Lemma 1 and because N_1 and N_2 are disjoint, ϕ is injective. We have:

$$\begin{aligned}\phi^{-1}((m_1, I_1) \parallel (m_2, I_2)) = (m, I_1[T_{1\backslash 2}] \oplus I^\infty[\mathcal{E}_N(m) \cap T_{1\times 2}] \oplus I_2[T_{2\backslash 1}]) \\ \text{where } m = m_1[P_1] \oplus m_2[P_2]\end{aligned}$$

We show by induction that the pair (ϕ, id), where id is the identity mapping over $\Sigma_1 \cup \Sigma_2 \cup \{\epsilon\} \cup \mathbf{R}^+$, is a graph isomorphism between $\lceil N_1 \parallel N_2 \rceil$ and $\lceil N_1 \rceil \parallel \lceil N_2 \rceil$:

(i) $s_1^0 \parallel s_2^0 = \phi(s^0)$, where s^0 and the s_i^0 are the initial states of $\lceil N \rceil$ and $\lceil N_i \rceil$: By definition, $s^0 = (m^0, Is[\mathcal{E}_N(m^0))$ and $s_i^0 = (m_i^0, Is_i[\mathcal{E}_{N_i}(m_i^0)])$. From the assumptions, $m_i^0 = m^0[P_i]$ and $I_i^0 = I^0[T_{i\backslash j}] \oplus I^\infty[\mathcal{E}_{N_i}(m_i^0)\backslash T_{i\backslash j}]$‘ <, hence $s_1^0 \parallel s_2^0 = \phi(s_0)$.

(ii) $(\forall a)(\forall s, s' \in \lceil N \rceil)(s \xrightarrow{a} s' \Leftrightarrow \phi(s) \xrightarrow{a} \phi(s'))$: Four cases must be considered:

1. $a \in R^+$: Let $(m, I) = s$ and $(m_1, I_1) \parallel (m_2, I_2) = \phi(s)$.
   From Definition 4, we have $(m, I) \xrightarrow{a} (m, I')$ iff:
   (a) $(\forall t \in \mathcal{E}_N(m))(a \leq \uparrow I(t) \wedge I'(t) = I(t) \dot{-} a)$
   From the definition of the TTS product, we have $(m_1, I_1) \parallel (m_2, I_2) \xrightarrow{a} (m_1, I_1') \parallel (m_2, I_2')$ iff $(m_1, I_1) \xrightarrow{a} (m_1, I_1')$ and $(m_2, I_2) \xrightarrow{a} (m_2, I_2')$, or:
   (b1) $(\forall t \in \mathcal{E}_{N_1}(m_1))(a \leq \uparrow I_1(t) \wedge I_1'(t) = I_1(t) \dot{-} a) \wedge$
   (b2) $(\forall t \in \mathcal{E}_{N_2}(m_2))(a \leq \uparrow I_2(t) \wedge I_2'(t) = I_2(t) \dot{-} a)$
   By Lemma 1, (a) (resp. bi) is trivially satisfied for the transitions in $T_{1\times 2}$ (resp. $\mathcal{E}_{N_i}(m_i)\backslash T_{i\backslash j}$). Further, from ϕ, $m_i = m[P_i]$ and I agrees with I_1 (resp. I_2) on the transitions of $T_{1\backslash 2}$ (resp. $T_{2\backslash 1}$), hence (a) $\Leftrightarrow$ (b1) $\wedge$ (b2) and $(m_1, I_1') \parallel (m_2, I_2') = \phi(m, I')$.
2. $a \in \Sigma_1 \cap \Sigma_2$: Let $(m, I) = s$ and $(m_1, I_1) \parallel (m_2, I_2) = \phi(s)$.
   We have $(m, I) \xrightarrow{L(t)} (m', I')$ iff $c_N^1 \wedge \ldots \wedge c_N^5$, where conditions c_N^i are those in Definition 4 for discrete transitions, specialized for N. On the other hand, from the definition of the TTS product, we have $(m_1, I_1) \parallel (m_2, I_2) \xrightarrow{a} (m_1', I_1') \parallel (m_2', I_2')$ iff $(m_1, I_1) \xrightarrow{a} (m_1', I_1')$ by some t_1 and $(m_2, I_2) \xrightarrow{a} (m_2', I_2')$ by some t_2, that is if $(c_{N_1}^1 \wedge \ldots \wedge c_{N_1}^5) \wedge (c_{N_2}^1 \wedge \ldots \wedge c_{N_2}^5)$. As $a \in \Sigma_1 \cap \Sigma_2$, there must be t, t_1 and t_2 such that t is the product of t_1 and t_2.
   Then, we have $c_N^1 \Leftrightarrow c_{N_1}^1 \wedge c_{N_2}^1$ from ϕ and the definitions of products, $c_N^2 \Leftrightarrow c_{N_1}^2 \wedge c_{N_2}^2$ by Lemma 1, and $c_N^3 \Leftrightarrow c_{N_1}^3 \wedge c_{N_2}^3$ by the properties of Pr: t is preempted in N iff either t_1 is preempted in $N1$ or t_2 in $N2$. Finally, it is easily shown that $(m_i, I_i) \xrightarrow{a} (m'[P_i], I'[T_{i\backslash j}] \oplus I^\infty[\mathcal{E}_{N_i}(m_i'[P_i])\backslash T_{i\backslash j}]$, hence $(m_1', I_1') \parallel (m_2', I_2') = \phi(s')$.
3. $a \in \Sigma_1^\epsilon \setminus \Sigma_2$: Let $(m, I) = s$ and $(m_1, I_1) \parallel (m_2, I_2) = \phi(s)$.
   We have $(m, I) \xrightarrow{a} (m', I')$ by t iff $c_N^1 \wedge \ldots \wedge c_N^5$. From the definition of products, we have $(m_1, I_1) \parallel (m_2, I_2) \xrightarrow{a} (m_1', I_1') \parallel (m_2, I_2)$ iff, by the previous t, $(m_1, I_1) \xrightarrow{a} (m_1', I_1')$, or $c_{N_1}^1 \wedge \ldots \wedge c_{N_1}^5$. Then, as for case 2, we have that the preconditions for both transitions are equivalent and that the target states obey $(m_1', I_1') \parallel (m_2', I_2') = \phi(s')$.
4. for $a \in \Sigma_2^\epsilon \setminus \Sigma_1$: like case 3, symmetrically.