Timed Automata with Disjoint Activity

Marco Muñiz, Bernd Westphal, and Andreas Podelski

Albert-Ludwigs-Universität Freiburg, 79110 Freiburg, Germany
{muniz,westphal,podelski}@informatik.uni-freiburg.de

Abstract. The behavior of timed automata consists of idleness and activity, i.e. delay and action transitions. We study a class of timed automata with periodic phases of activity. We show that, if the phases of activity of timed automata in a network are disjoint, then location reachability for the network can be decided using a concatenation of timed automata. This reduces the complexity of verification in Uppaal-like tools from quadratic to linear time (in the number of components) while traversing the same reachable state space. We provide templates which imply, by construction, the applicability of sequential composition, a variant of concatenation, which reflects relevant reachability properties while removing an exponential number of states. Our approach covers the class of TDMA-based (Time Division Multiple Access) protocols, e.g. FlexRay and TTP. We have successfully applied our approach to an industrial TDMA-based protocol of a wireless fire alarm system with more than 100 sensors.

1 Introduction

Timed real world applications may include a large number of components. These components can often be modeled using timed automata [1] and their composition. The behavior of timed automata consists of idleness and activity, i.e. delay and action transitions. In many cases the activity of the timed automata (which model an application) is disjoint i.e. one automaton is active while the other ones are idle, except at time points where they may synchronize. In timed automata with disjoint activity their parallel product will introduce many edges and locations which are not relevant given the disjoint activity assumption (unreachable locations). These many edges are unnecessarily evaluated and increase the verification costs in tools like Uppaal [3].

In this paper, we formalize a notion of timed automata with disjoint activity and characterize a class of timed automata with periodic cycles of activity. Then, we define a semantic concatenation operator which, when applied to timed automata with disjoint activity, produces automata bisimilar to the one obtained by their parallel composition. The automaton obtained by the concatenation operator has a reduced number of edges and locations than the one obtained

M. Jurdziński and D. Ničković (Eds.): FORMATS 2012, LNCS 7595, pp. 188-203, 2012.

by the parallel composition operator. Identifying the periodic cyclic phases of activity of a timed automata, and if these activity phases are disjoint can be as hard as verifying a property. Therefore, we introduce templates for timed automata which by construction ensure periodic phases of activity. In addition, a syntactic check in the instances of the templates suffices to ensure that their corresponding phases of activity are disjoint. We used our approach for verifying a real world *wireless fire alarm* system with up to 125 sensors. By using our approach in Uppaal the verification times decreased from quadratic to linear on the number of sensors.

An important class of systems which can be verified using our approach is the class of real-time network protocols which use the *Time Division Multiple Access* design principle. Well-known examples in this class of *TDMA protocols* include the Bluetooth protocol of [6], the time triggered protocol of [12], and the time triggered architectures of [11, 7, 10].

1.1 Related work

Since the cost of model checking for a network of timed automata increases exponentially in the number of components, much research has been directed towards techniques that demonstrate a potentially exponential speedup in interesting applications (see, e.g., [2]). It turns out that, for the class of timed automata with disjoint activity , the cost increases quadratically in the number of components (which, as confirmed in our experiments, can be bad enough with an increasing number of components); thus, the best one can hope for, is to be able to demonstrate a potentially linear speedup (see Figure 1, page 202). Little research has been devoted to the (comparatively modest, albeit occasionally relevant) goal of a linear speedup; in particular, we are not aware of techniques that are directly related to our approach. Still, let us note that the technique of active clock reduction of [4] and its generalisation in [2], which may seem relevant in this context, are orthogonal to our approach; in fact, when we present our experimental evaluation, we evaluate the improvement obtained by the syntactic transformation (for a non-optimized model) with respect to the execution time for an optimized model which has only one clock.

In [5, 9, 14] Communication-Closed Layers and timed automata are studied. The approach presented in [14] and ours are complementary. The main differences are that the approach in [14] is action based, whereas our approach is time based. In addition, we consider cyclic timed automata, whereas in [14], automata can not perform actions after reaching their corresponding final location.

In Section 6 we introduce sequential timed automata and present in Definition 9 the notion of an overclock for two clocks. We use this notion to reduce the number of clocks in sequential timed automata. In [8] this notion is generalized to quasi-equal clocks and a more general reduction method for quasi-equal clocks is presented. However, we show that for the context of sequential timed automata, the Sequentialisation method proposed in Section 6 yields an Automaton in which verification can be carried out in linear time on the number of sequential timed automata, whereas by using the method proposed in [8] the

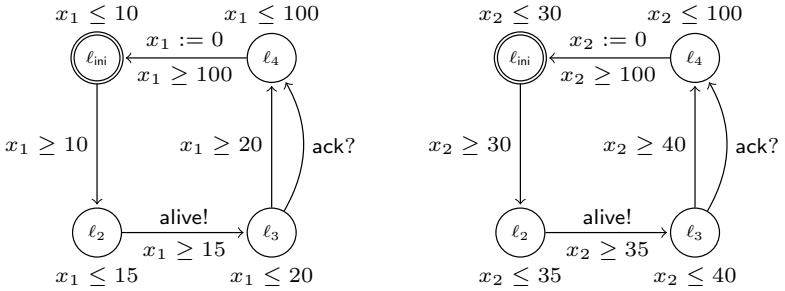

Fig. 1. Timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ modeling the behavior of two simplified sensors. The constants $10, 20$ and $30, 40$ denote the start resp. the end of the i-th time slot interval. The i-th sensor waits for the start of its designated time slot, sends its alive signal and waits for the ack signal from the central unit to arrive before the end.

verification will be carried out in quadratic time. This is illustrated in our case study in Section 7.

2 Applications

First, we elucidate our method by explaining it through an example. Next, we describe how our method could be applied to a Time-Triggered Architecture system.

2.1 Example: Fire alarm system

In the simplified (and simplistic) version that we consider for the example of a TDMA protocol, the wireless fire alarm system is a network of a central unit and a number n of sensors; here, $n = 10$. Figure 1 shows two timed automata which model two (simplified) sensors. The protocol operates in cycles of fixed length of time, say 100. The cycle is split in n time slot intervals, each of the same length of time. In each cycle, the central unit listens at the i-th time slot to an alive message from the i-th sensor. If the alive message is received, the central unit replies with an ack message.

Figure 2 shows the timed automaton that is 'equivalent' to the parallel product of the timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$; it is obtained from applying the operation that we will introduce in Section 6. We note three phenomena that we observe on the example (and describe the concepts that we will introduce in the corresponding section in order to investigate the phenomena). These phenomena constitute the premises of our approach.

(1) The initial location of $\mathcal{A}_i$ is visited infinitely often, with a regular period (in Section 4, we define notions of cyclicity and periodicity of timed automata).

(2) The i-th sensor mostly (but not exclusively!) performs action transitions at the i-th time slot (in Section 5, we formally characterize the notion of activity

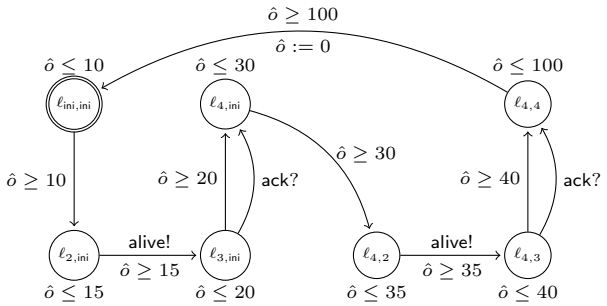

Fig. 2. The timed automaton $\mathcal{A}_1 \fatsemi \mathcal{A}_2$ that is 'equivalent' to the parallel product of the timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ from Figure 1, obtained from applying the operation that we will introduce in Section 6.

for timed automata, give a semantics-based definition of two timed automata being sequentialisable, and introduce the concatenation '$\cdot$' of sequentialisable timed automata).

(3) In the interleaving semantics of the parallel composition of the automata $\mathcal{A}_1$ and $\mathcal{A}_2$, the two different (zero-time) action transitions with the reset of the clock x_1 resp. the reset of the clock x_2 diverge, i.e., they lead to states where the values for x_1 and x_2 are different (in Section 6, when we define the sequential composition '$\fatsemi$' of sequentialisable timed automata, we recover the determinacy of the behavior of the parallel product through the concept of the *overclock*, i.e., the introduction of a new clock $\hat{o}$).

In addition, we observe that the automaton $\mathcal{A}_1 || \mathcal{A}_2$, has an increased number of edges in comparision to $\mathcal{A}_1 \fatsemi \mathcal{A}_2$. The reduced number of edges yield a linear reduction in the verification time, justified by Lemma 3.

2.2 Potential Application: Steer-by-wire architecture using TTP/C

In order to illustrate the applicability of our method, we propose an informal model for a system based on the Time-Triggered Architecture.

In the Time-Triggered Architecture every node consist of a local cpu, a Communication Network Interface and a TTP/C controller. The data communication over TTP/C is organized in TDMA rounds. A TDMA round is divided into slots and every node is assigned to a slot. A recurring sequence of TDMA rounds constitutes a cluster. The system can be globally monitored based on bus tracing.

A steer-by-wire system would require from 8 to 30 nodes. Interesting properties to verify might include: If a node fails, is this node detected as malfunctioning in within a TDMA round; or if a node fails, does the system still satisfy a given property.

Let us consider that the system has n nodes and one global monitor. The monitor could be modeled by one timed automaton $\mathcal{A}_m$. Every node could be modeled by Two timed Automata; one for the cpu $\mathcal{A}_{cpu}$ and one for the TTP/C

communication $\mathcal{A}_{TTP}$. The automata corresponding to a node could communicate via shared variables. A TDMA round would have the following form, $\mathcal{A}_m||\mathcal{A}_{cpu_1}||\ldots||\mathcal{A}_{cpu_n}||\mathcal{A}_{TTP_1}||\ldots||\mathcal{A}_{TTP_n}$. Since Automaton $\mathcal{A}_{TPP_i}$ will only perform actions on its corresponding slot for $i \in \{1,\ldots,n\}$, we can apply our method and obtain: $\mathcal{A}_m||\mathcal{A}_{cpu_1}||\ldots||\mathcal{A}_{cpu_n}||(\mathcal{A}_{TTP_1} \fatsemi \ldots \fatsemi \mathcal{A}_{TTP_n})$. Which we believe, would lead to an improvement on the verification time. Given a fixed number of TDMA rounds we can use the above method to construct a cluster.

3 Preliminaries

The formal basis for our work are timed automata [1]. In the following, we briefly recall the main definitions, our presentation follows [13]. Note that, in contrast to [13], we do not distinguish transition sequences and (time-stamped) computation paths. Here, the configurations of the labelled transition system are already time-stamped.

Let $\mathbb{X}$ be a set of clocks. The set $\Phi(\mathbb{X})$ of simple clock constraints over $\mathbb{X}$ is defined by the grammar $\varphi ::= x \sim C \mid x - y \sim C \mid \varphi_1 \wedge \varphi_2$ where $x, y \in \mathbb{X}$, $C \in \mathbb{Q}_0^+$, and $\sim \in \{<, \leq, \geq, >\}$. We assume the canonical satisfaction relation "$\models$" between *valuations* of the clocks $\nu : \mathbb{X} \to \mathsf{Time}$ and simple clock constraints, with $\mathsf{Time} = \mathbb{R}_{>0}$.

A Timed Automaton (TA) $\mathcal{A}$ is a tuple $(L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$, which consists of a finite set of *locations* L, with typical element ℓ, a finite set Σ of *actions* comprising the *internal action* τ, a finite set of *clocks* $\mathbb{X}$, a mapping $I : L \to \Phi(\mathbb{X})$, that assigns to each location a *clock constraint*, and a set of *edges* $E \subseteq L \times \Sigma \times \Phi(\mathbb{X}) \times \mathcal{P}(\mathbb{X}) \times L$. An edge $e = (\ell, \alpha, \varphi, Y, \ell') \in E$ from ℓ to ℓ' involves an action $\alpha \in \Sigma$, a *guard* $\varphi \in \Phi(\mathbb{X})$, and a *reset set* $Y \subseteq \mathbb{X}$.

The operational semantics of the timed automaton $\mathcal{A}$ is the labelled transition system $\mathcal{TS}(\mathcal{A}) = (\mathit{Conf}(\mathcal{A}), \mathsf{Time} \cup \Sigma, \{\xrightarrow{\lambda} \mid \lambda \in \mathsf{Time} \cup \Sigma\}, C_{\mathsf{ini}})$. The set of *configurations* $\mathit{Conf}(\mathcal{A}) = \{(\langle \ell, \nu\rangle, t) \in L \times (\mathbb{X} \to \mathsf{Time}) \times \mathsf{Time} \mid \nu \models I(\ell)\}$ consists of time-stamped pairs of a location $\ell \in L$ and a valuation of the clocks $\nu : \mathbb{X} \to \mathsf{Time}$ which satisfies the clock constraint $I(\ell)$. The set of initial configurations is $C_{\mathsf{ini}} = \{(\langle \ell_{\mathsf{ini}}, \nu_{\mathsf{ini}}\rangle, 0)\} \cap \mathit{Conf}(\mathcal{A})$ where $\nu_{\mathsf{ini}}(x) = 0$ for all clocks $x \in \mathbb{X}$. There is a *delay transition* from configuration $\langle \ell, \nu\rangle, t$ to $\langle \ell, \nu + t'\rangle, t + t'$, i.e. $\langle \ell, \nu\rangle, t \xrightarrow{t'} \langle \ell, \nu + t'\rangle, t + t'$, if and only if $\nu + t'' \models I(\ell)$ for all $t'' \in [0, t']$, where $\nu + t'$ denotes the valuation obtained from ν by time shift t'. There is an *action transition* between $\langle \ell, \nu\rangle, t$ and $\langle \ell', \nu'\rangle, t$, i.e $\langle \ell, \nu\rangle, t \xrightarrow{\alpha} \langle \ell', \nu'\rangle, t$, if and only if there exists an edge $(\ell, \alpha, \varphi, Y, \ell') \in E$ with $\nu \models \varphi$, $\nu' = \nu[Y := 0]$, and $\nu' \models I(\ell')$, where $\nu[Y := 0]$ denotes the valuation obtained from ν by resetting exactly the clocks in Y. We write $\ell(c)$, $\nu(c)$, and $t(c)$, to denote the location ℓ, valuation ν, and time-stamp t of a configuration $c = \langle \ell, \nu\rangle, t$.

An infinite or maximally finite sequence $\pi = c_0 \xrightarrow{\lambda_0} c_1 \xrightarrow{\lambda_1} c_2 \ldots$ is called a *computation* of $\mathcal{A}$ if and only if $c_0 \in C_{\mathsf{ini}}$ and $(c_i, c_{i+1}) \in \xrightarrow{\lambda}$ for all $i \in \mathbb{N}_0$. We write π_j to denote the j-th configuration $c_j = \langle \ell_j, \nu_j\rangle, t_j$ in π, and λ_j^π to denote the label of j-th transition in π, or simply λ_j if π is clear from the context. We write $\pi \in \mathcal{TS}(\mathcal{A})$ if and only if π is a computation of $\mathcal{A}$.

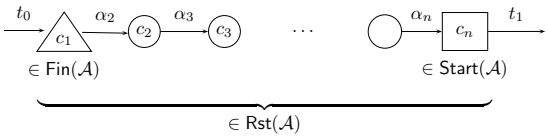

Fig. 3. Start configurations are in the initial location and have an action predecessor and a delay successor, final configurations a delay predecessor and an action successor. Configurations on an action-only path between a final and a start configuration are called restart configurations.

The *parallel composition* of two timed automata $\mathcal{A}_i = (L_i, \Sigma_i, \mathbb{X}_i, I_i, E_i, \ell_{\mathsf{ini},i})$, $i = 1, 2$, with disjoint sets of clocks $\mathbb{X}_1$ and $\mathbb{X}_2$ yields the timed automaton $\mathcal{A}_1 \| \mathcal{A}_2 \stackrel{\mathsf{def}}{=} (L_1 \times L_2, \Sigma_1 \cup \Sigma_2, \mathbb{X}_1 \cup \mathbb{X}_2, I, E, (\ell_{\mathsf{ini},1}, \ell_{\mathsf{ini},2}))$ where $I(\ell_1, \ell_2) := I_1(\ell_1) \wedge I_2(\ell_2)$, for each $\ell_1 \in L_1, \ell_2 \in L_2$, and where E consists of *handshake* and *asynchronous edges* defined as follows. There is a handshake transition $((\ell_1, \ell_2), \tau, \varphi_1 \wedge \varphi_2, Y_1 \cup Y_2, (\ell_1', \ell_2')) \in E$ if there are *complementary* actions α and $\bar{\alpha}$ in $\Sigma_1 \cup \Sigma_2$ such that $(\ell_1, \alpha, \varphi_1, Y_1, \ell_1') \in E_1$ and $(\ell_2, \bar{\alpha}, \varphi_2, Y_2, \ell_2') \in E_2$. For each edge $(\ell_1, \alpha, \varphi_1, Y_1, \ell_1') \in E_1$ and each location $\ell_2 \in L_2$, there is an asynchronous transition $((\ell_1, \ell_2), \alpha, \varphi_1, Y_1, (\ell_1', \ell_2)) \in E$, and analogously for each transition in E_2.

4 Periodic Cyclic Timed Automata

Timed automata models of, e.g., TDMA-based protocols can be cyclic and periodic in the following sense. Intuitively, a timed automaton is cyclic if the initial location is visited infinitely often on all computations, the corresponding configurations are called start configuration. A timed automaton is periodic with period pt if configurations containing the initial location are reached only at integer multiples of the period and are reached from a unique final location.

In the following, we formally define periodic cyclic timed automata in terms of the new notions of start, restart, and final configurations (cf. Figure 3).

Definition 1 (Start and Final Configuration). *Let $\mathcal{A} = (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ be a timed automaton. The set* $\mathsf{Start}(\mathcal{A})$ *of* start configurations *of $\mathcal{A}$ consists of those configuration of $\mathcal{TS}(\mathcal{A})$ that are at location ℓ_{ini} and occur in a computation $\pi \in \mathcal{TS}(\mathcal{A})$ as source of a delay transition and as destination of an action transition, i.e.*

$$\mathsf{Start}(\mathcal{A}) \stackrel{\mathsf{def}}{=} \{c = \langle \ell_{\mathsf{ini}}, \nu \rangle, t \in \mathit{Conf}(\mathcal{A}) \mid \exists \pi \in \mathcal{TS}(\mathcal{A}), m \in \mathbb{N}_0 \bullet \\ \pi_m = c \wedge \lambda_m \in \mathsf{Time} \wedge (\lambda_{m-1} \in \Sigma \vee m = 0)\}.$$

The set $\mathsf{Rst}(\mathcal{A})$ *of* restart configurations *consists of those configuration of $\mathcal{TS}(\mathcal{A})$ that occur in a computation $\pi \in \mathcal{TS}(\mathcal{A})$ as action-predecessor of a start*

configuration, i.e.

$$\mathsf{Rst}(\mathcal{A}) \stackrel{\mathsf{def}}{=} \{c \in \mathit{Conf}(\mathcal{A}) \mid \exists \pi \in \mathcal{TS}(\mathcal{A}), m, i \in \mathbb{N}_0 \bullet m \leq i \wedge \pi_m = c \\ \wedge\, \pi_i \in \mathsf{Start}(\mathcal{A}) \wedge \pi_m \xrightarrow{\lambda_m} \ldots \xrightarrow{\lambda_{i-1}} \pi_i \wedge \forall m \leq j \leq i \bullet \lambda_j \in \Sigma\}.$$

The set $\mathsf{Fin}(\mathcal{A})$ *of* final configurations *consists of the maximal restart configurations of* $\mathcal{TS}(\mathcal{A})$*, that is, restart configurations which are the destination of a delay transition, i.e.*

$$\mathsf{Fin}(\mathcal{A}) \stackrel{\mathsf{def}}{=} \{c \in \mathsf{Rst}(\mathcal{A}) \mid \exists \pi \in \mathcal{TS}(\mathcal{A}), m \in \mathbb{N}_0 \bullet \\ \pi_m = c \wedge (\lambda_{m-1} \in \mathsf{Time} \vee m = 0)\}.$$

The set L_{rst} *of* restart locations *consists of those locations that occur in a restart configuration, i.e.*

$$L_{\mathsf{rst}} \stackrel{\mathsf{def}}{=} \{\ell \in L \mid \exists \nu : \mathbb{X} \to \mathsf{Time}, t \in \mathsf{Time} \bullet \langle \ell, \nu \rangle, t \in \mathsf{Rst}(\mathcal{A})\}.$$

Definition 2 (Periodic Cyclic). *A timed automaton* $\mathcal{A} = (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ *is called* periodic cyclic *with period* $pt \in \mathsf{Time}$ *if and only if each computation comprises infinitely many start configurations which occur at a regular period of time and if there is a unique* final location ℓ_{fin}*, i.e.*

$$\mathsf{peCy}(\mathcal{A}, pt) \stackrel{\mathsf{def}}{\Longleftrightarrow} \forall \pi \in \mathcal{TS}(\mathcal{A}), p \in \mathbb{N}_0\ \exists c = (\langle \ell, \nu \rangle, t) \in \mathsf{Start}(\mathcal{A}), i \in \mathbb{N}_0 \bullet \\ \pi_i = c \wedge t = pt \cdot p\ \wedge \\ \forall \pi \in \mathcal{TS}(\mathcal{A}), c = (\langle \ell, \nu \rangle, t) \in \mathsf{Start}(\mathcal{A}), i \in \mathbb{N}_0\ \bullet \\ \pi_i = c \Rightarrow \exists p \in \mathbb{N}_0 \bullet t = pt \cdot p\ \wedge \\ \exists \ell_{\mathsf{fin}} \in L\ \forall (\langle \ell, \nu \rangle, t) \in \mathsf{Fin}(\mathcal{A}) \bullet \ell = \ell_{\mathsf{fin}}.$$

Theorem 1. *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be periodic cyclic timed automata with period* $pt \in \mathsf{Time}$*. Then* $\mathcal{A}_1 \| \mathcal{A}_2$ *is periodic cyclic with period* pt*.*

5 Concatenation of Sequentialisable Timed Automata

We say a timed automaton is active at a point in time if there exists a computation where an action transition is taken at that time. Two periodic cyclic timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are called sequentialisable if, they have the same period and within each period, all activity of $\mathcal{A}_1$ lies strictly before all activity of $\mathcal{A}_2$, except for integer multiples of the period. In the following, we formally define activity and sequentialisability. We define the concatenation of two sequentialisable timed automata and show in Theorem 2 that the result satisfies exactly the same reachability and leads-to properties as the parallel composition of the two. In Lemma 3, we discuss the relation between outgoing and enabled edges in the parallel composition and our concatenation.

Definition 3 (Activity). *The set of* activity points $\mathsf{Active}(\mathcal{A}) \subseteq \mathsf{Time}$ *of a timed automaton* $\mathcal{A} = (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ *consists of those points in time at which action transitions take place in some computation, i.e.*

$$\mathsf{Active}(\mathcal{A}) \stackrel{\mathsf{def}}{=} \{t \in \mathsf{Time} \mid \exists \pi \in \mathcal{TS}(\mathcal{A}), j \in \mathbb{N} \bullet \lambda_j \in \Sigma \wedge t(\pi_j) = t\}.$$

Definition 4 (Sequentialisable). *Two timed automata* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *are called* sequentialisable *if and only if*

1. $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *have disjoint sets of clocks,*
2. $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *are periodic cyclic with period* $pt \in \mathsf{Time}$*, and*
3. *for each* $p \in \mathbb{N}_0$*,* within *the* p*-th period,* $\mathcal{A}_1$ *is active strictly before* $\mathcal{A}_2$*, i.e.*

$$sup(\mathsf{Active}_p(\mathcal{A}_1)) < inf(\mathsf{Active}_p(\mathcal{A}_2))$$

where $\mathsf{Active}_p(\mathcal{A}_i) \stackrel{\mathsf{def}}{=} \mathsf{Active}(\mathcal{A}_i) \cap]pt \cdot p, pt \cdot (p+1)[$*,* $i = 1, 2$.

Note 1. Within the p-th period, $p \in \mathbb{N}_0$, the activity points of two sequentialisable timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are disjoint, i.e.

$$\mathsf{Active}(\mathcal{A}_1) \cap \mathsf{Active}(\mathcal{A}_2) \cap]pt \cdot p, pt \cdot (p+1)[= \emptyset.$$

If $\mathcal{A}_1$ and $\mathcal{A}_2$ are sequentialisable. Then on each period, first $\mathcal{A}_1$ is active and reaches its final location while $\mathcal{A}_2$ is at its initial location. Subsequently, $\mathcal{A}_2$ is active and reaches its final location while $\mathcal{A}_1$ is at its final location. At the end of the period both $\mathcal{A}_1$ and $\mathcal{A}_2$ are active at locations corresponding to their reset configurations.

Lemma 1. *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be sequentialisable timed automata with period* pt*.*

1. *For all points of time different than the integer multiples of* pt *and within the activity of* $\mathcal{A}_1$*,* $\mathcal{A}_2$ *is in its initial location* ℓ_{ini_2}*, i.e.*

$$\begin{aligned}&\forall p \in \mathbb{N}_0, t \in \mathsf{Time} \bullet t \in [inf(\mathsf{Active}_p(\mathcal{A}_1)), sup(\mathsf{Active}_p(\mathcal{A}_1))] \\ &\quad \Longrightarrow \forall \pi \in \mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)\ \exists \langle (\ell_1, \ell_2), \nu \rangle, t \in Conf(\mathcal{A}_1 \| \mathcal{A}_2), j \in \mathbb{N}_0 \bullet \\ &\qquad \pi_j = \langle (\ell_1, \ell_2), \nu \rangle, t \wedge \ell_2 = \ell_{\mathsf{ini}_2}.\end{aligned}$$

2. *For all points of time different than the integer multiples of* pt *and within the activity of* $\mathcal{A}_2$*,* $\mathcal{A}_1$ *is in its final location* ℓ_{fin_1}*, i.e.*

$$\begin{aligned}&\forall p \in \mathbb{N}_0, t \in \mathsf{Time} \bullet t \in [inf(\mathsf{Active}_p(\mathcal{A}_2)), sup(\mathsf{Active}_p(\mathcal{A}_2))] \\ &\quad \Longrightarrow \forall \pi \in \mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)\ \exists \langle (\ell_1, \ell_2), \nu \rangle, t \in Conf(\mathcal{A}_1 \| \mathcal{A}_2), j \in \mathbb{N}_0 \bullet \\ &\qquad \pi_j = \langle (\ell_1, \ell_2), \nu \rangle, t \wedge \ell_1 = \ell_{\mathsf{fin}_1}.\end{aligned}$$

3. *In each computation, both,* $\mathcal{A}_1$ *and* $\mathcal{A}_2$*, are simultaneously at a restart location at integer multiples of* pt*, i.e.*

$$\begin{aligned}&\forall p \in \mathbb{N}_0, t \in \mathsf{Time} \bullet t = pt \cdot p \\ &\quad \Longrightarrow \forall \pi \in \mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)\ \exists \langle (\ell_1, \ell_2), \nu \rangle, t \in Conf(\mathcal{A}_1 \| \mathcal{A}_2), j \in \mathbb{N}_0 \bullet \\ &\qquad \pi_j = \langle (\ell_1, \ell_2), \nu \rangle, t \wedge \ell_1 \in L_{\mathsf{rst}} \wedge \ell_2 \in L_{\mathsf{rst}}.\end{aligned}$$

For sequentialisable automata $\mathcal{A}_1$ and $\mathcal{A}_2$ with period pt, Lemma 1 suggests that for time points different than $pt \cdot p$ for some p. It is not necessary to compute the product of the locations on $\mathcal{A}_1$ and $\mathcal{A}_2$. The following concatenation operator exploits this fact and computes the product of locations only if both $\mathcal{A}_1$ and $\mathcal{A}_2$ are active. i.e. at time points $pt \cdot p$.

Definition 5 (Concatenation).
Let $\mathcal{A}_1 = (L_1, \Sigma_1, \mathbb{X}_1, I_1, E_1, \ell_{\mathsf{ini}_1})$ *and* $\mathcal{A}_2 = (L_2, \Sigma_2, \mathbb{X}_2, I_2, E_2, \ell_{\mathsf{ini}_2})$ *be sequentialisable timed automata with period* $pt \in \mathsf{Time}$. *Let* ℓ_{fin_i} *denote the final location and let* L_{rst_i} *denote the set of restart locations of automaton* $\mathcal{A}_i$, $i \in \{1, 2\}$.

The concatenation *of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *yields the timed automaton*

$$\mathcal{A}_1 \cdot \mathcal{A}_2 \stackrel{\mathsf{def}}{=} (L, \Sigma_1 \cup \Sigma_2, \mathbb{X}_1 \cup \mathbb{X}_2, I, E, \ell_{\mathsf{ini}})$$

where

- $L = (L_1 \times \{\ell_{\mathsf{ini}_2}\}) \cup (\{\ell_{\mathsf{fin}_1}\} \times L_2) \cup (L_{\mathsf{rst}_1} \times L_{\mathsf{rst}_2})$
- $I(\ell_1, \ell_2) = I_1(\ell_1) \wedge I_2(\ell_2)$, $\ell_1 \in L_1, \ell_2 \in L_2$,
- $E = \{((\ell_1, \ell_2), \alpha, \varphi_1, Y_1, (\ell_1', \ell_2)) \mid (\ell_1, \alpha, \varphi_1, Y_1, \ell_1') \in E_1 \wedge \ell_2 \in L_{\mathsf{rst}_2}\}$
  $\cup \{((\ell_1, \ell_2), \alpha, \varphi_2, Y_2, (\ell_1, \ell_2')) \mid (\ell_2, \alpha, \varphi_2, Y_2, \ell_2') \in E_2 \wedge \ell_1 \in L_{\mathsf{rst}_1}\}$, *and*
- $\ell_{\mathsf{ini}} = (\ell_{\mathsf{ini}_1}, \ell_{\mathsf{ini}_2})$.

Definition 6 (Bisimulation). *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be timed automata and*

$$\mathcal{TS}_i(\mathcal{A}_i) = (\mathit{Conf}(\mathcal{A}_i), \mathsf{Time} \cup \Sigma_i, \{\xrightarrow{\lambda^i} \mid \lambda^i \in \mathsf{Time} \cup \Sigma_i\}, C_{\mathsf{ini}_i})$$

the corresponding labelled transition systems. A relation $\mathcal{R} \subseteq \mathit{Conf}(\mathcal{A}_1) \times \mathit{Conf}(\mathcal{A}_2)$ *is called* bisimulation *of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *if and only if it satisfies the following conditions.*

1. $\forall c_1 \in C_{\mathsf{ini}_1}\ \exists c_2 \in C_{\mathsf{ini}_2} \bullet (c_1, c_2) \in \mathcal{R}$ *and* $\forall c_2 \in C_{\mathsf{ini}_2}\ \exists c_1 \in C_{\mathsf{ini}_1}) \bullet (c_1, c_2) \in \mathcal{R}$
2. *for all* $(c_1 = (\langle \ell_1, \nu_1 \rangle, t_1),\ c_2 = (\langle \ell_2, \nu_2 \rangle, t_2)) \in \mathcal{R}$,
   (a) $\nu_1 = \nu_2$, $t_1 = t_2$,
   (b) $\forall c_1 \xrightarrow{\lambda^1} c_1'\ \exists c_2 \xrightarrow{\lambda^2} c_2 \bullet (c_1', c_2') \in \mathcal{R}$
   (c) $\forall c_2 \xrightarrow{\lambda^2} c_2'\ \exists c_1 \xrightarrow{\lambda^1} c_1' \bullet (c_1', c_2') \in \mathcal{R}$.

$\mathcal{A}_1$ *is called bisimilar to* $\mathcal{A}_2$ *iff there exists a bisimulation of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$.

For sequentialisable timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$, the implications of Lemma 1 and the definition of the concatenation operator imply that the transition system $\mathcal{TS}(\mathcal{A}_1 \cdot \mathcal{A}_2)$ corresponds to the reachable part of $\mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)$.

Theorem 2. *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be sequentialisable timed automata. Then* $\mathcal{TS}(\mathcal{A}_1 \cdot \mathcal{A}_2)$ *is bisimilar to* $\mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)$.

Theorem 2, ensures that the start configurations of $\mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2)$ are in $\mathcal{TS}(\mathcal{A}_1 \cdot \mathcal{A}_2)$. Therefore, the following theorem holds.

Theorem 3. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be sequentialisable timed automata with period pt. Then $\mathcal{A}_1 \cdot \mathcal{A}_2$ is periodic cyclic with period pt.*

Lemma 2 (Bisimulation). *Reachability properties are preserved under bisimulation, i.e. given bisimilar timed automata $\mathcal{A}_1$ and $\mathcal{A}_2$ and a state assertion φ, i.e., an expression over clock constraints and locations, we have*

$$(\exists \pi \in \mathcal{TS}(\mathcal{A}_1)\ \forall j \in \mathbb{N}_0 \bullet \pi_j \models \varphi) \iff (\exists \pi \in \mathcal{TS}(\mathcal{A}_2)\ \forall j \in \mathbb{N}_0 \bullet \pi_j \models \varphi)$$
$$(\forall \pi \in \mathcal{TS}(\mathcal{A}_1)\ \forall j \in \mathbb{N}_0 \bullet \pi_j \models \varphi) \iff (\forall \pi \in \mathcal{TS}(\mathcal{A}_2)\ \forall j \in \mathbb{N}_0 \bullet \pi_j \models \varphi).$$

Definition 7 (Enabled Edges). *Let $\mathcal{A} = (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ be a timed automaton and $c \in Conf(\mathcal{A})$ a configuration.*

We use $out(c)$ to denote the set of outgoing edges in c, i.e. the edges $e = (\ell, \alpha, \varphi, Y, \ell') \in E$ where $\ell = \ell(c)$.

Edge e is called enabled *if and only if its guard is satisfied and the effect of resets satisfies the clock constraint of the destination of e, i.e., if $\nu(c) \models \varphi$ and $\nu(c)[Y := 0] \models I(\ell')$. We use $enab(c)$ to denote the set of edges enabled in c.*

Note 2. Let $\mathcal{A}_1, \ldots, \mathcal{A}_n$ be timed automata with pairwise disjoint edge sets and let $c = \langle (\ell_1, \ldots, \ell_n), \nu \rangle, t \in Conf(\mathcal{A}_1 \| \ldots \| \mathcal{A}_n)$ be a configuration.

1. Enabled edges are in particular outgoing, i.e. $enab(c) \subseteq out(c)$.
2. The set of outgoing edges in the parallel composition is determined by the components, i.e.

$$out(c) = \bigcup_{1 \le i \le n} out(\langle \ell_i, \nu|_{\mathbb{X}_i} \rangle, t), enab(c) = \bigcup_{1 \le i \le n} enab(\langle \ell_i, \nu|_{\mathbb{X}_i} \rangle, t),$$

thus (with disjoint edge sets)

$$|out(c)| = \Sigma_{1 \le i \le n} |out(\langle \ell_i, \nu|_{\mathbb{X}_i} \rangle, t)|, |enab(c)| = \Sigma_{1 \le i \le n} |enab(\langle \ell_i, \nu|_{\mathbb{X}_i} \rangle, t)|.$$

Since, outgoing edges are evaluated, a reduction on the number of outgoing edges yields a reduction on time complexity. This reduction can go from quadratic to linear time, as the following lemma shows.

Lemma 3. *Let $\mathcal{A}_1, \ldots, \mathcal{A}_n$ be sequentialisable timed automata with period pt with disjoint edge sets and with exactly one outgoing edge per location.*

1. *Let $c \in Conf(\mathcal{A}_1 \| \ldots \| \mathcal{A}_n)$ be a configuration of the parallel composition of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ where the time-stamp is not an integer multiple of the period pt, i.e. where $\nexists p \in \mathbb{N}_0 \bullet t(n) = p \cdot pt$.*
   Then $|out(c)| = n$ and $|enab(c)| = 1$.
2. *Let $c \in Conf(\mathcal{A}_1 \cdot \ldots \cdot \mathcal{A}_n)$ be a configuration of the concatenation of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ where the time-stamp is not an integer multiple the period pt.*
   Then $|out(c)| = |enab(c)| = 1$.

6 Sequential Composition of Sequential Timed Automata

As the decision whether a given pair of timed automata is sequentialisable is in general at least as difficult as the considered analysis problem, we provide a syntactical pattern such that instances of the pattern are sequentialisable and such that a specialized sequential composition applies.

Definition 8 (Sequential Timed Automaton). *A* sequential timed automaton *(STA) is a tuple*

$$\mathcal{A} = (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}}, \ell_{\mathsf{fin}}, \mathsf{sta}, \mathsf{fin}, pt, e_{\mathsf{fin}}, \hat{x})$$

where $\mathcal{A}_0 \stackrel{\mathsf{def}}{=} (L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ *is a timed automaton,* ℓ_{fin} *is a final location,* $\mathsf{sta}, \mathsf{fin} \in \mathbb{Q}_0^+$ *are* start *and* final time*,* $pt \in \mathbb{Q}_0^+$ *is a period,* $e_{\mathsf{fin}} \in E$ *is an edge of the form* $(\ell_{\mathsf{fin}}, \varnothing, \hat{x} \geq pt, Y \cup \{\hat{x}\}, \ell_{\mathsf{ini}})$ $\hat{x} \in \mathbb{X}$ *is a master clock, which satisfies the following syntactical constraints:*

- *the start time is positive and strictly smaller than the final time, which is strictly smaller than the period, i.e.*

$$0 < \mathsf{sta} \wedge \mathsf{sta} < \mathsf{fin} \wedge \mathsf{fin} < pt, \tag{saActive}$$

- *the initial location is left if* $\hat{x}$ *reaches the start time, i.e.*

$$I(\ell_{\mathsf{ini}}) = \hat{x} \leq \mathsf{sta}, \tag{saStart}$$
$$\forall (\ell, \alpha, \varphi, Y, \ell') \in E \bullet \ell = \ell_{\mathsf{ini}} \implies \varphi = \hat{x} \geq \mathsf{sta}, \tag{saStartTime}$$

- *locations connected by an edge to the final location are only assumed until* $\hat{x}$ *reaches the final time, i.e.*

$$\forall (\ell, \alpha, \varphi, Y, \ell') \in E \bullet \ell' = \ell_{\mathsf{fin}} \Rightarrow I(\ell) = \hat{x} \leq \mathsf{fin}, \tag{saFinalTime}$$

- *the final location is only assumed until* $\hat{x}$ *reaches the period, i.e.*

$$I(\ell_{\mathsf{fin}}) = \hat{x} \leq pt, \tag{saPeriod}$$

- *the master clock is reset exactly on edge* e_{fin}*, i.e.*

$$\forall (\ell, \alpha, \varphi, Y, \ell') \in E \bullet \hat{x} \in Y \Rightarrow (\ell, \alpha, \varphi, Y, \ell') = e_{\mathsf{fin}}, \tag{saOneReset}$$

- ℓ_{fin} *and* ℓ_{ini} *are connected exactly by edge* e_{fin}*, i.e.*

$$\forall (\ell, \alpha, \varphi, Y, \ell') \in E \bullet \ell = \ell_{\mathsf{fin}} \wedge \ell' = \ell_{\mathsf{ini}} \implies (\ell, \alpha, \varphi, Y, \ell') = e_{\mathsf{fin}}, \tag{saOneFin}$$

and the following semantical constraint:

- *whenever the initial location is assumed, the final location is finally reached, i.e.*

$$\begin{aligned} &\forall \pi \in \mathcal{TS}(\mathcal{A}_0), j \in \mathbb{N}_0, \langle \ell, \nu \rangle, t \in \mathit{Conf}(\mathcal{A}_0) \bullet \pi_j = \langle \ell, \nu \rangle, t \wedge \ell = \ell_{\mathsf{ini}} \\ &\quad \implies \exists k \in \mathbb{N}_0, \langle \ell', \nu' \rangle, t' \in \mathit{Conf}(\mathcal{A}_0) \bullet \\ &\qquad k \geq j \wedge \pi_k = \langle \ell', \nu' \rangle, t' \wedge \ell' = \ell_{\mathsf{fin}} \end{aligned} \tag{saCyclic}$$

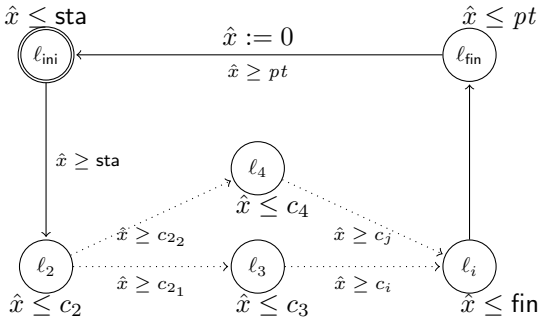

Fig. 4. A syntactical pattern for sequential timed automata.

Let $\mathcal{A}'$ be the automaton obtained by replacing synchronization transitions in $\mathcal{A}$ by internal transitions. Then, saCyclic can be checked for $\mathcal{A}'$ in a model checker. Figure 4 depicts a purely syntactically restricted template which ensures the instances to be sequential automata.

Theorem 4. *Let* $(L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}}, \ell_{\mathsf{fin}}, \mathsf{sta}, \mathsf{fin}, pt, e_{\mathsf{fin}}, \hat{x})$ *be a sequential timed automaton. Then* $(L, \Sigma, \mathbb{X}, I, E, \ell_{\mathsf{ini}})$ *is periodic cyclic with period* pt*.*

Before applying the sequential operator we must be sure that the activity phases of the automata are disjoint. In sequential automata sequentialisability can be syntacticaly proven, as the following lemma shows.

Lemma 4. *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be sequential timed automata with the same period* pt *and final time* fin_1 *of* $\mathcal{A}_1$ *strictly smaller than the start time* sta_2 *of* $\mathcal{A}_2$*, i.e.* $\mathsf{fin}_1 < \mathsf{sta}_2$*. Then* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *are sequentialisable.*

Consider the parallel product or the concatenation of n sequential automata (see Figure 4) which are sequentialisable. The automaton will include a diamond like structure corresponding to the product of the n final locations. This structure will have 2^n locations. In this locations the master clocks $\hat{x}_1, \ldots, \hat{x}_n$ will not be equal, but note that this happends in zero time. Therefore, we define a new clock *overclock* $\hat{o}$ which preserves the quasi-equalitiy of the clocks, and allows to further optimize the resulting automaton.

Definition 9 (Overclock). *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be timed automata with clocks* x_1 *and* x_2*, respectively. A clock* $\hat{o}$ *of* $\mathcal{A}_1$ *or* $\mathcal{A}_2$ *is an* overclock *for* x_1 *and* x_2 *in* $\mathcal{A}_1 \| \mathcal{A}_2$ *if and only if*

$$\forall \pi \in \mathcal{TS}(\mathcal{A}_1 \| \mathcal{A}_2), j \in \mathbb{N}_0 \langle (\ell_1, \ell_2), \nu \rangle, t \in \mathit{Conf}(\mathcal{A}_1 \| \mathcal{A}_2) \bullet \pi_j = \langle (\ell_1, \ell_2), \nu \rangle, t$$
$$\implies \nu \models (x_1 = \hat{o} \wedge x_2 = \hat{o}) \vee \ell_1 \in L_{\mathsf{rst}_1} \vee \ell_2 \in L_{\mathsf{rst}_2}.$$

Lemma 5. *Given sequential timed automata,* $\mathcal{A}_1, \mathcal{A}_2$ *with period* pt *and masterclocks* $\hat{x}_1$*,* $\hat{x}_2$ *respectively. Then, there exist an overclock* $\hat{o}$ *for* $\hat{x}_1$ *and* $\hat{x}_2$*.*

The diamond structure which we described above, will have 2^n locations and $n!$ zero time interleaving sequences, which occur at time points $pt \cdot p$ for $p \in \mathbb{N}_+$. Note that clocks $\hat{x}_1, \ldots, \hat{x}_n$ are always equal up to the time points $pt \cdot p$. The sequential composition operator which we define below, will remove this diamond structure and will replace all the clocks by an overclock.

Definition 10 (Sequential Composition). *Let* $\mathcal{A}_i$,

$$\mathcal{A}_i = (L_i, \Sigma_i, \mathbb{X}_i, I_i, E_i, \ell_{\mathsf{ini}_i}, \ell_{\mathsf{fin}_i}, \mathsf{sta}_i, \mathsf{fin}_i, pt, e_{\mathsf{fin}_i}, \hat{x}_i), i = 1, 2,$$

be sequential timed automata.

Then the sequential composition *of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *yields the tuple*

$$\mathcal{A}_1 \fatsemi \mathcal{A}_2 \stackrel{\mathsf{def}}{=} (L, \Sigma_1 \cup \Sigma_2, \mathbb{X}_1 \cup \mathbb{X}_2, I, E, (\ell_{\mathsf{ini}_1}, \ell_{\mathsf{ini}_2}), (\ell_{\mathsf{fin}_1}, \ell_{\mathsf{fin}_2}), \mathsf{sta}_1, \mathsf{fin}_2, pt, e_{\mathsf{fin}}, \hat{o})$$

where

- *the master clock* $\hat{o}$ *is an overclock for* $\hat{x}_1$, $\hat{x}_2$.
- *the set of locations consists of pairs where* $\mathcal{A}_2$ *assumes an initial or* $\mathcal{A}_1$ *assumes a final location, i.e.*

$$L = ((L_1 \setminus \{\ell_{\mathsf{fin}_1}\}) \times \{\ell_{\mathsf{ini}_2}\}) \cup (\{\ell_{\mathsf{fin}_1}\} \times L_2),$$

- *the final edge* e_{fin} *is*

$$((\ell_{\mathsf{fin}_1}, \ell_{\mathsf{fin}_2}), \varnothing, \varphi_1 \wedge \varphi_2, Y_1 \cup Y_2, (\ell_{\mathsf{ini}_1}, \ell_{\mathsf{ini}_2}))$$

  given the final edges $e_{\mathsf{fin}_i} = (\ell_{\mathsf{fin}_i}, \varnothing, \varphi_i, Y_i, \ell_{\mathsf{ini}_i})$, $i = 1, 2$,
- *the clock constraint of location* (ℓ_1, ℓ_2) *is the conjunction of the corresponding clock constraints in* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *where substitute each* $\hat{x}_1$ *and* $\hat{x}_2$ *is syntactically substituted by* $\hat{o}$, *i.e.*

$$I(\ell_1, \ell_2) = (I_1(\ell_1) \wedge I(\ell_2))[\hat{x}_1/\hat{o}, \hat{x}_2/\hat{o}],$$

- *the set of edges comprises* e_{fin} *and compositions of* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *edges where* $\hat{x}_1$ *and* $\hat{x}_2$ *are substituted by* $\hat{x}$ *in guards and reset sets, i.e.*

$$E = \{e_{\mathsf{fin}}\} \cup \{((\ell_1, \ell_{\mathsf{ini}_2}), \alpha, \tilde{\varphi}_1, \tilde{Y}_1, (\ell_1', \ell_{\mathsf{ini}_2})) \mid (\ell_1, \alpha, \varphi_1, Y_1, \ell_1') \in E_1 \setminus \{e_{\mathsf{fin}_1}\}\}$$
$$\cup \{((\ell_{\mathsf{fin}_1}, \ell_2), \alpha, \tilde{\varphi}_2, \tilde{Y}_2, (\ell_{\mathsf{fin}_1}, \ell_2')) \mid (\ell_2, \alpha, \varphi_2, Y_2, \ell_2') \in E_2 \setminus \{e_{\mathsf{fin}_2}\}\}$$

  where $\tilde{\varphi}_i = \varphi_i[\hat{x}_1/\hat{o}, \hat{x}_2/\hat{o}]$, $i = 1, 2$, *and* $\tilde{Y}_i = Y_i[\hat{x}_1/\hat{o}, \hat{x}_2/\hat{o}]$, $i = 1, 2$.

Theorem 5. *Let* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *be sequential timed automata with the same period* pt *and final time* fin_1 *of* $\mathcal{A}_1$ *strictly smaller than the start time* sta_2 *of* $\mathcal{A}_2$, *i.e.* $\mathsf{fin}_1 < \mathsf{sta}_2$. *Then* $\mathcal{A}_1 \fatsemi \mathcal{A}_2$ *is periodic cyclic with period* pt.

In Section 5, we have shown that For sequential automata $\mathcal{A}_1, \ldots, \mathcal{A}_n$ which are sequentialisable $\mathcal{A}_1 || \ldots || \mathcal{A}_n$ and $\mathcal{A}_1 \cdot \cdots \cdot \mathcal{A}_n$ are bisimilar. In what follows, we will show that $\mathcal{A}_1 || \ldots || \mathcal{A}_n$ and $\mathcal{A}_1 \fatsemi \ldots \fatsemi \mathcal{A}_n$ are weak-bisimilar (which reflects the effect of removing the diamond like structure in the sequential composition of the n sequential automata). We use the following definition in order to simplify a definition of weak-bisimulation.

Definition 11 (Action Reachability). *Let $\mathcal{A}$ be a timed automaton and $c, c' \in Conf(\mathcal{A})$ configurations. We say c' is* reachable *in $\mathcal{A}$ from c* via action transitions*, denoted by* $\mathsf{actReach}(c, c', \mathcal{A})$*, if and only if*

$$\mathsf{actReach}(c, c', \mathcal{A}) \overset{\mathsf{def}}{\Longleftrightarrow} \exists c_0, c_1, c_2, \ldots, c_n \in Conf(\mathcal{A}), i \in \mathbb{N}_0 \bullet c_0 = c \wedge c_n = c' \\ \wedge \forall 0 \leq j < n \in \mathbb{N}_0 \bullet c_j \xrightarrow{\lambda_j} c_{j+1} \wedge \lambda_j \in \Sigma.$$

Definition 12 (Weak-bisimulation). *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be sequential automata with $\hat{o} \in \mathbb{X}_1$ and $\hat{x}_1, \hat{x}_2 \in \mathbb{X}_2$ such that $\hat{o}$ is an overclock for $\hat{x}_1, \hat{x}_2$ and let*

$$\mathcal{TS}_i(\mathcal{A}_i) = (Conf(\mathcal{A}_i), \mathsf{Time} \cup \Sigma_i, \{\xrightarrow{\lambda^i} | \ \lambda^i \in \mathsf{Time} \cup \Sigma_i\}, C_{\mathsf{ini}_i}), i = 1, 2,$$

be the corresponding labelled transition systems.

A relation $\mathcal{W} \subseteq Conf(\mathcal{A}_1) \times Conf(\mathcal{A}_2)$ is called weak-bisimulation *of $\mathcal{A}_1$ and $\mathcal{A}_2$ if and only if it satisfies the following conditions.*

1. $\forall c_1 \in C_{\mathsf{ini}_1} \ \exists c_2 \in C_{\mathsf{ini}_2} \bullet (c_1, c_2) \in \mathcal{W}$ *and* $\forall c_2 \in C_{\mathsf{ini}_2} \ \exists c_1 \in C_{\mathsf{ini}_1}) \bullet (c_1, c_2) \in \mathcal{W}$
2. *for all* $(c_1 = (\langle \ell_1, \nu_1 \rangle, t_1), c_2 = (\langle \ell_2, \nu_2 \rangle, t_2)) \in \mathcal{W}$,
   (a) $\beta(\nu_1) = \nu_2$, $t_1 = t_2$ *with* $\beta(\nu) = \nu|_{\hat{o}} \cup \{\nu_{\hat{x}_1} \mapsto \nu(\hat{o}), \nu_{\hat{x}_2} \mapsto \nu(\hat{o})\}$,
   (b) $\forall c_1 \xrightarrow{\lambda^1} c_1' \bullet (\exists c_2 \xrightarrow{\lambda^2} c_2' \bullet (c_1', c_2') \in \mathcal{W}) \vee (\ell(c_1) = \ell_{\mathsf{fin}_1} \wedge$

$$\exists c_2'' \bullet \mathsf{actReach}(c_2, c_2'', \mathcal{A}_2) \wedge \ell(c_2'') = \ell_{\mathsf{ini}_2} \wedge (c_1', c_2'') \in \mathcal{W}),$$

   (c) $\forall c_2 \xrightarrow{\lambda^2} c_2' \ \exists c_1 \xrightarrow{\lambda^1} c_1' \bullet (c_1', c_2') \in \mathcal{W} \vee (\ell(c_2) = \ell_{\mathsf{fin}_2} \wedge$

$$\exists c_2'' \bullet \mathsf{actReach}(c_2, c_2'', \mathcal{A}_2) \wedge \ell(c_2'') = \ell_{\mathsf{ini}_2} \wedge (c_1', c_2'') \in \mathcal{W}),$$

$\mathcal{A}_1$ is called weak-bisimilar to $\mathcal{A}_2$ iff there is a weak-bisimulation of $\mathcal{A}_1$ and $\mathcal{A}_2$.

Theorem 6. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be sequential timed automata with the same period pt and final time fin_1 of $\mathcal{A}_1$ strictly smaller than the start time sta_2 of $\mathcal{A}_2$, i.e. $\mathsf{fin}_1 < \mathsf{sta}_2$. Then $\mathcal{A}_1 \fatsemi \mathcal{A}_2$ is weak-bisimilar to $\mathcal{A}_1 \| \mathcal{A}_2$.*

7 Case Study

For the example in Section 2.1, we have used a simplified version of a fire alarm system which monitors the well functioning of n sensors by sending and receiving alive messages. For our case study, we consider a real world fire alarm system which we denote by *FAS* (the system is being developed by a German company; an anonymized version of a model of the system will be made public). The *FAS* monitors n sensors using m channels. In order, for *FAS* to obtain an EU quality certificate it has to be conform, among others, with the following condition: If a sensor is malfunctioning, it has to be recognized in less than 300 seconds. We denote this property by AG *less300*.

System	Q	Broadcast		Sequential	
		t (s)	states	t (s)	states
FAS-CB-SW	φ_1	1103.9	5947.4k	318.1	5971.5k
	φ_2	2240.5	11508.3k	704.2	11614.5k
FAS-CB	φ_1	196.4	1184.7k	120.3	1189.5k
	φ_2	272.6	165.7k	150.3	1666.9k
FAS-SW	φ_1	13.7	104.1k	87.4	104.7k
	φ_2	10.62k	145.5	87.4	146.4k
FAS	φ_1	2.5	20.7k	87.5	20.8k
	φ_2	1.3	20.7k	85.6	20.8k

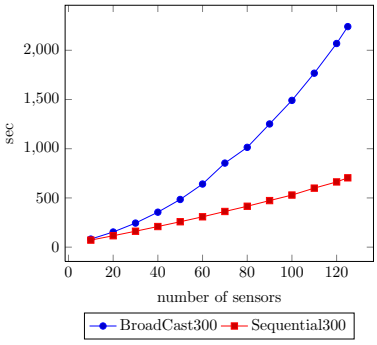

Table 1. Verification times (AMD Opteron 6174 2.2GHz, 64Gb RAM) and visited states for satisfied properties $\varphi_1 = (\mathsf{AG}\ \textit{not deadlock})$ and $\varphi_2 = (\mathsf{AG}\ \textit{less300})$ for the fire alarm system with 125 sensors using Uppaal, and verification times of *FAS-CB-SW* for property ($\mathsf{AG}\ \textit{less300}$) over number of sensors.

In addition, the certifying institution is able to (i) block an arbitrary channel for any number of seconds, then (ii) release the blocked channel for at least 1 second and repeat (i), (ii) any number of times.

In order to model the above mentioned situations, we constructed a sensor switcher *SW* which non-deterministically turns off any sensor. We constructed a channel blocker *CB*, which models the blocking of channels as described above. Now, let *FAS-CB* denote the fire alarm system together with the channel blocker *CB*. Let *FAS-SW* be *FAS* together with the sensor switcher. Let *FAS-CB-SW* be *FAS* together with the channel blocker and a sensor switcher.

Table 1, show the verification results for the satisfied properties $\mathsf{AG}\ \textit{not deadlock}$ and $\mathsf{AG}\ \textit{less300}$ for the corresponding system with 125 sensors by using Uppaal. The times include the parsing time of Uppaal templates. The attempt of modeling a sensor, with its own clock, did not scale to more than 10 sensors. Therefore, our modelers manually optimized the system, such that all 125 sensors share one clock, and synchronizations are performed via a broadcast channel. This optimized systems correspond to the column Broadcast. In addition, the system *FAS-CB-SW* Broadcast corresponds to the system obtained by applying the technique presented in [8].

We observe that for a large state space, sequential is much faster that broadcast as expected by Lemma 3. However, for small state space such as *FAS* broadcast is faster; in this context, note that the parsing time for the large template consisting of 125 sequentialized automata is taking about 85 sec.

Considering the verification times of the system *FAS-CB-SW* and property $\mathsf{AG}\ \textit{less300}$ for 10, 20, ..., 120, and finally 125 sensors, the curve for broadcast is comparable with the statement of Lemma 3 (cf. Table 1).

8 Conclusion and Future Work

We have presented an approach for optimizing the timed model checking method for the class of timed automata with disjoint activity. We have presented a syntactic transformation, by which parallel composition of its component automata is replaced by the application of a new sequential composition operator. The approach uses the syntactic transformation as a preprocessing step with an existing timed model checking method. We have implemented the approach (using Uppaal) and applied it to verify a wireless fire alarm system with more than 100 sensors. The experimental evaluation indicates the practical potential of the approach for improving upon the time cost in a useful manner.

For future work we may consider richer forms of expressing the property of sequentialisability, for example by means of handshaking communication.

References

1. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Science 126(2), 183–235 (1994)
2. Behrmann, G., Bouyer, P., Fleury, E., Larsen, K.G.: Static guard analysis in timed automata verification. In: TACAS. pp. 254–270. No. 2619 in LNCS, Springer (2003)
3. Behrmann, G., David, A., Larsen, K.G.: A tutorial on UPPAAL. In: Bernardo, M., Corradini, F. (eds.) SFM-RT 2004. pp. 200–236. No. 3185 in LNCS, Springer (2004)
4. Daws, C., Yovine, S.: Reducing the number of clock variables of timed automata. In: Proc. RTSS'96, 73-81, IEEE. pp. 73–81. IEEE Computer Society Press (1996)
5. Elrad, T., Francez, N.: Decomposition of distributed programs into communication-closed layers. Sci. Comput. Program. 2(3), 155–173 (1982)
6. Haartsen, J.C.: The Bluetooth radio system. Personal Communications, IEEE 7(1), 28 –36 (Feb 2000)
7. Heiner, G., Thurner, T.: Time-triggered architecture for safety-related distributed real-time systems in transportation systems. In: FTCS. pp. 402–407 (1998)
8. Herrera, C., Westphal, B., Feo-Arenis, S., Muñiz, M., Podelski, A.: Reducing quasi-equal clocks in networks of timed automata. In: FORMATS. pp. 155–170. No. 7595 in LNCS, Springer, Heidelberg (2012)
9. Janssen, W., Poel, M., Xu, Q., Zwiers, J.: Layering of real-time distributed processes. In: FTRTFT. pp. 393–417. No. 863 in LNCS, Springer (1994)
10. Kopetz, H.: The time-triggered approach to real-time systems design. In: Randell, B., et al. (eds.) Predictably Dependable Computing Systems. Springer (1995)
11. Kopetz, H., Bauer, G.: The Time-Triggered Architecture. In: Proc. IEEE. pp. 112–126 (2003)
12. Kopetz, H., Grünsteidl, G.: TTP - a time-triggered protocol for fault-tolerant real-time systems. In: FTCS. pp. 524–533 (1993)
13. Olderog, E.R., Dierks, H.: Real-Time Systems - Formal Specification and Automatic Verification. Cambridge University Press (2008)
14. Olderog, E.R., Swaminathan, M.: Layered composition for timed automata. In: FORMATS. pp. 228–242. No. 6246 in LNCS, Springer (2010)