Compositional Model Checking of Real Time Systems *

François Laroussinie, Kim G. Larsen

BRICS** , Aalborg Univ., Denmark ***

Abstract. We present a compositional model checking technique for networks of timed automata. This method is based on the same idea as the recent method proposed by Andersen [4] for untimed case. We present a quotient construction, which allows timed automata components to be gradually moved from the network expression into the specification. The intermediate specifications are kept small using minimization heuristics suggested by Andersen. The potential of the combined technique is demonstrated using a prototype implemented in CAML.

1 Introduction

Within the last decade model checking has turned out to be a useful technique for verifying temporal properties of finite state systems. Efficient model checking algorithms for finite state systems have been obtained with respect to a number of logics, and in the last few years, model checking techniques have been extended to real–time systems and logics using the region technique of Alur, Courcoubetis and Dill [1]. However, the major problem in applying model checking even to moderate–size systems is the potential combinatorial explosion of the state space arising from parallel composition. For real–time systems an additional explosion is induced in the number of regions. In order to avoid this problem algorithms have been sought that avoid exhaustive state (region) space exploration, either by *symbolic* representation of the states space such as in [13] and in the use of Binary Decision Diagrams [8], by application of *partial order* methods [12, 23] which suppresses unnecesarry interleavings of transitions, or by application of *abstractions* and *symmetries* [9, 10, 11].

So far the most successful results for larger systems have been obtained using the heuristics of Binary Decision Diagrams. However, recent work by Andersen [4] introduces a new very promising heuristic model checking technique for finite state systems, for which experimental results on specific examples shows an improvement compared with Binary Decision Diagrams. The technique is based on compositional proof rules for parallel composition.

* This work has been supported by the European Communities under CONCUR2, BRA 7166

** **B**asic **R**esearch in **C**omputer **S**cience, Centre of the Danish National Research Foundation.

*** Dept. of Computer Science, Aalborg University, Fredrik Bajers Vej 7-E, DK-9220 Aalborg, Denmark, (email: {fl,kgl}@iesd.auc.dk) fax: (45) 98.15.81.29

The aim of this paper is to make this new successful (compositional) model checking technique by Andersen [4] applicable to real–time systems. For example, consider the following typical model checking problem $(\mathcal{S}_1 \mid \ldots \mid \mathcal{S}_n) \models \varphi$ where the $\mathcal{S}_i$'s are real–time systems (described as timed automata [2]). We want to verify that the parallel composition of these timed automata satisfies the formula φ without having to construct the complete state (region) space of the process $(\mathcal{S}_1 \mid \ldots \mid \mathcal{S}_n)$. We will avoid this complete construction by removing the components $\mathcal{S}_i$ one by one while simultaneously transforming the formula accordingly. Thus, when removing the component $\mathcal{S}_n$ we will transform the formula φ into the *quotient* formula $\varphi \,/\, \mathcal{S}_n$ such that

$$\Big(\mathcal{S}_1 \mid \ldots \mid \mathcal{S}_n\Big) \models \varphi \quad \text{if and only if} \quad \Big(\mathcal{S}_1 \mid \ldots \mid \mathcal{S}_{n-1}\Big) \models \varphi \,/\, \mathcal{S}_n \tag{1}$$

Now clearly, if the quotient is not much larger than the original formula we have succeeded in simplifying the problem. Repeated application of quotienting yields

$$\Big(\mathcal{S}_1 \mid \ldots \mid \mathcal{S}_n\Big) \models \varphi \quad \text{if and only if} \quad \mathbf{1} \models \varphi \,/\, \mathcal{S}_n \,/ \mathcal{S}_{n-1} \,/\, \ldots \,/\, \mathcal{S}_1 \tag{2}$$

where $\mathbf{1}$ is the unit with respect to parallel composition.

For finite state systems the quotient with respect to parallel composition is an immediate application of work on compositional reasoning due to Andersen, Larsen, Stirling, Winskel and Xinxin [17, 18, 6, 5]. However, based on these ideas alone, (2) provides no solution to the problem as the explosion will now occur in the size of the final quotient formula instead. The crucial and experimentally "verified" observation by Andersen was that each quotienting should be followed by a *minimization* of the formula based on a collection of few, efficiently implementable strategies.

In this paper we provide the basis for and make an initial experimental investigation of the application of Andersen's compositional model checking technique for real–time systems (timed automata). In particular,

- We give an effective construction of the quotient formula $\varphi \,/\, \mathcal{S}$ satisfying the requirement of (1) for φ a formula of the timed logic L_ν introduced in [16] and $\mathcal{S}$ a real–time system given in terms of a timed automaton;
- Based on a prototype implemented in CAML we make an experimental investigation of the above quotient construction combined with (some of) the minimization heuristics of Andersen. In the examples we consider the minimized quotient formulas have been subject to dramatic reductions and are comparable in size to the original formulas. Thus, we may expect compositional model checking to be successful also for real–time systems.

The remainder of this paper is organized as follows: in the next section we give a short presentation of the notion of timed automata and composition used in this paper; in section 3, the logic L_ν is shortly presented, and in section 4 we review the region technique by Alur, Courcoubetis and Dill [1]. Section 5 presents the quotient construction, whereas section 6 presents and investigates the consequences of minimization.

2 Timed Transition Systems and Automata

Let $\mathcal{A}$ be a finite set of actions ranged over by $a, b, c, \ldots$. We denote by $\mathbf{N}$ the set of natural numbers and by $\mathbf{R}$ the set of non–negative real numbers. $\mathcal{D}$ denotes the set of delay actions $\{\epsilon(d) \mid d \in \mathbf{R}\}$, and $\mathcal{L}$ denotes the union $\mathcal{A} \cup \mathcal{D}$.

Definition 1. *A* timed transition system *over $\mathcal{A}$ is a tuple $\mathcal{S} = \langle S, s_0, \longrightarrow \rangle$, where S is a set of states, s_0 is the initial state and $\longrightarrow \subseteq S \times \mathcal{L} \times S$ is a transition relation. We require that for any $s \in S$ and $d \in \mathbf{R}$, there exists a (unique) state s^d such that $s \xrightarrow{\epsilon(d)} s^d$. Moreover $(s^d)^e = s^{d+e}$* [4].

Obviously, we may apply the standard notion of bisimulation [21, 19] to timed transition systems. For $\mathcal{S} = \langle S, s_0, \longrightarrow \rangle$ a timed transition system, strong (timed) bisimulation $\sim$ is defined as the largest symmetric relation over S such that whenever $s_1 \sim s_2$, $\ell \in \mathcal{A} \cup \mathcal{D}$ and $s_1 \xrightarrow{\ell} s_1'$ then there exists s_2' such that $s_2 \xrightarrow{\ell} s_2'$ and $s_1' \sim s_2'$. We may compare states from different timed transition systems by considering their disjoint union. Two timed transition systems $\mathcal{S}_1$ and $\mathcal{S}_2$ are said to be strong (timed) bisimilar, written $\mathcal{S}_1 \sim \mathcal{S}_2$, in case their initial states are strong bisimilar.

In order to study compositionality problems we introduce a parallel composition between timed transition systems. Following [14] we suggest a composition parameterized with a synchronization function generalizing a large range of existing notions of parallel compositions. A *synchronization function* f is a partial function $(\mathcal{A} \cup \{0\}) \times (\mathcal{A} \cup \{0\}) \hookrightarrow \mathcal{A}$, where 0 denotes a distinguished no–action symbol [5]. Now, let $\mathcal{S}_i = \langle S_i, s_{i,0}, \longrightarrow_i \rangle$, $i = 1, 2$, be two timed transition systems and let f be a synchronization function. Then the *parallel composition* $\mathcal{S}_1 \,|_f\, \mathcal{S}_2$ is the timed transition system $\langle S, s_0, \longrightarrow \rangle$, where $s_1 \,|_f\, s_2 \in S$ whenever $s_1 \in S_1$ and $s_2 \in S_2$, $s_0 = s_{1,0} \,|_f\, s_{2,0}$, and $\longrightarrow$ is given by the rule [6]:

$$\frac{s_1 \xrightarrow{a}_1 s_1' \qquad s_2 \xrightarrow{b}_2 s_2'}{s_1 \,|_f\, s_2 \xrightarrow{c} s_1' \,|_f\, s_2'} \; f(a, b) \simeq c$$

and the requirement that for any $d \in \mathbf{R}$, $(s_1 \,|_f\, s_2)^d = ({s_1}^d \,|_f\, {s_2}^d)$.

Syntactically, timed transition systems are described by timed automata [2], which are finite automata extended with a finite collection of real–valued clocks [7]. If C is a set of clocks, $\mathcal{B}(C)$ denotes the set of formulas built using boolean connectives over atomic formulas of the form $x \leq m$, $m \leq x$, $x \leq y + m$ and $y + m \leq x$ with $x, y \in C$ and $m \in \mathbf{N}$. Moreover $\mathcal{B}_M(C)$ denotes the subset of $\mathcal{B}(C)$ with no constant greater than M.

---

[4] The existence of s^d corresponds to transition liveness, its unicity corresponds to time-determinism and the property $(s^d)^e = s^(d + e)$ corresponds to time-continuity (or time-additivity) in [24].

[5] We extend the transition relation of a timed transition system such that $s \xrightarrow{0} s'$ iff $s = s'$.

[6] $f(a, b) \simeq c$ holds if f is defined for the pair (a, b) and the result is c.

[7] Timed transition systems may alternatively be described using timed process calculi.

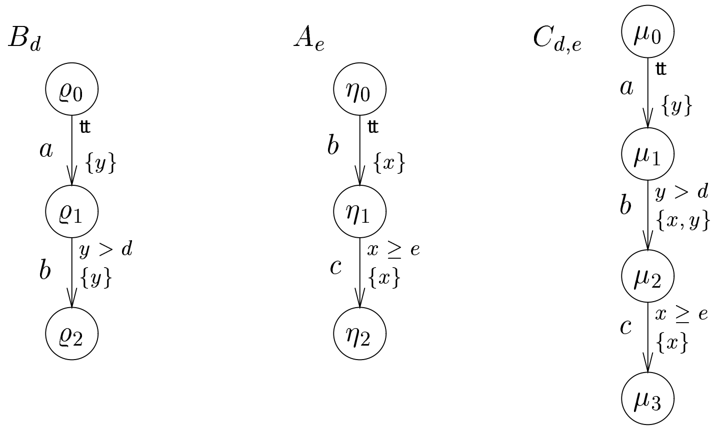


Fig. 1. Three timed automata

Definition 2. *A timed automaton A over $\mathcal{A}$ is a tuple $\langle N, \eta_0, C, E\rangle$ where N is a finite set of nodes, η_0 is the initial node, C is a finite set of clocks, and $E \subseteq N \times N \times \mathcal{A} \times 2^C \times \mathcal{B}(C)$ corresponds to the set of edges. $e = \langle \eta, \eta', a, r, b\rangle \in E$ represents an edge from the node η to the node η' with action a, r denoting the set of clocks to be reset and b is the enabling condition over the clocks of A.*

Example 1. Consider the automata A_e, B_d and $C_{d,e}$ in Figure 1 ($d, e \in \mathbf{N}$). The automaton $C_{d,e}$ has four nodes, μ_0, μ_1, μ_2 and μ_3, two clocks x and y, and three edges. The edge between μ_1 and μ_2 has b as action, $\{x, y\}$ as reset set and the enabling condition for the edge is $y > d$. $\square$

Informally, the system starts at node η_0 with all its clocks initialized to 0. The values of the clocks increase synchronously with time. At any time, the automaton whose current node is η can change node by following an edge $\langle \eta, \eta', a, r, b\rangle \in E$ provided the current values of the clocks satisfy b. With this transition the clocks in r get reset to 0.

Formally a time assignment v for C is a function from C to $\mathbf{R}$. We denote by $\mathbf{R}^C$ the set of time assignments for C. For $v \in \mathbf{R}^C$, $x \in C$ and $d \in \mathbf{R}$, $v + d$ denotes the time assignment which maps each clock x in C to the value $v(x) + d$. For $C' \subseteq C$, $[C' \mapsto 0]v$ denotes the assignment for C which maps each clock in C' to the value 0 and agrees with v over $C \backslash C'$. Whenever $v \in \mathbf{R}^C$, $u \in \mathbf{R}^K$ and C and K are disjoint vu denotes the time assignment over $C \cup K$ such that $(vu)(x) = v(x)$ if $x \in C$ and $(vu)(x) = u(x)$ if $x \in K$. Given a condition $b \in \mathcal{B}(C)$ and a time assignment $v \in \mathbf{R}^C$, $b(v)$ is a boolean value describing whether b is satisfied by v or not. Finally a k–clock automata is a timed automata $\langle N, \eta_0, C, E\rangle$ such that $|C| = k$.

A *state* of an automaton A is a pair (η, v) where η is a node of A and v a time assignment for C. The initial state of A is (η_0, v_0) where v_0 is the time assignment mapping all clocks in C to 0.

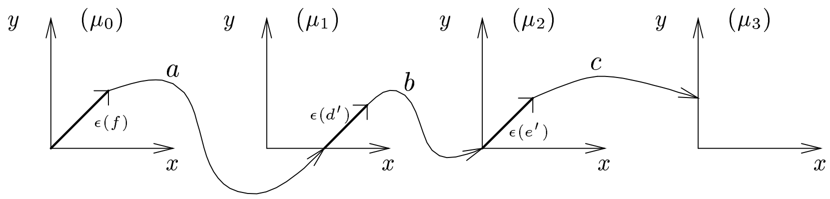


Fig. 2. Partial Behaviour of $C_{d,e}$.

The semantics of A is given by the timed transition system $\mathcal{S}_A = \langle S_A, \sigma_0, \longrightarrow_A \rangle$, where S_A is the set of states of A, σ_0 is the initial state (η_0, v_0), and $\longrightarrow_A$ is the transition relation defined as follows:

$$(\eta, v) \xrightarrow{a} (\eta', v') \text{ iff } \exists r, b.\ \langle \eta, \eta', a, r, b \rangle \in E \ \wedge \ b(v) \ \wedge \ v' = [r \mapsto 0]v$$
$$(\eta, v) \xrightarrow{\epsilon(d)} (\eta', v') \text{ iff } \eta = \eta' \text{ and } v' = v + d$$

Example 2. Reconsider the automaton $C_{d,e}$ of Figure 1. The coordinate systems in Figure 2 indicates (some of) the states of $C_{d,e}$. Each point of the coordinate systems represents a unique time assignment, with the coordinate systems representing states involving the nodes μ_0, μ_1, μ_2 and μ_3 respectively. We have indicated the following typical transition sequence (where $f \geq 0$, $d' > d$ and $e' \geq e$):

$$(\mu_0, (0,0)) \xrightarrow{\epsilon(f)} (\mu_0, (f,f)) \xrightarrow{a} (\mu_1, (f,0)) \xrightarrow{\epsilon(d')} (\mu_1, (f+d', d'))$$
$$(\mu_1, (f+d', d')) \xrightarrow{b} (\mu_2, (0,0)) \xrightarrow{\epsilon(e')} (\mu_2, (e', e')) \xrightarrow{c}$$

□

Parallel composition may now be extended to timed automata in the obvious way: for two timed automata A and B and a synchronization function f, the parallel composition $A \,|_f\, B$ denotes the timed transition system $\mathcal{S}_A \,|_f\, \mathcal{S}_b$. For two automata A and B and a synchronization function f one may effectively construct an automaton $A \otimes_f B$ such that $\mathcal{S}_{A \otimes_f B}$ is strong bisimilar to $\mathcal{S}_A \,|_f\, \mathcal{S}_B$. The nodes of $A \otimes_f B$ is simply the product of A's and B's nodes, the set of clocks is the (disjoint) union of A's and B's clocks, and the edges are based on synchronizable A and B edges with enabling conditions conjuncted and reset–sets unioned.

Example 3. Let f be the synchronization function completely specified by $f(a,0) = a$, $f(b,b) = b$ and $f(0,c) = c$. Then the automaton $C_{d,e}$ in Figure 1 is isomorphic to the part of $B_d \otimes_f A_e$ which is reachable from (ρ_0, η_0). □

3 Timed Logic

We consider a dense–time logic L_ν with clocks and recursion. This logic may be seen as a certain fragment [8] of the μ–calculus T_μ presented in [13]. In [16] it has

[8] allowing only maximal recursion and using a slightly different notion of model

been shown that this logic is sufficiently expressive that for any timed automaton one may construct a single *characteristic* formula uniquely characterizing the automaton up to timed bisimilarity. Also, decidability of a satisfiability [9] problem is demonstrated.

Definition 3. *Let K a finite set of clocks,* Id *a set of identifiers and k an integer. The set L_ν of formulae over K,* Id *and k is generated by the abstract syntax with φ and ψ ranging over L_ν:*

$$\begin{aligned}\varphi \ ::= \ & \mathsf{tt} \mid \mathsf{ff} \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \exists\, \varphi \mid \forall\!\!\!\forall\, \varphi \mid \langle a \rangle\, \varphi \mid [a]\, \varphi \\ & \mid x \ \mathsf{in}\, \varphi \mid x + n \bowtie y + m \mid \mid x \bowtie m \mid Z\end{aligned}$$

where $a \in \mathcal{A}$; $x, y \in K$; $n, m \in \{0, 1, \ldots, k\}$; $\bowtie \in \{=, <, \leq, >, \geq\}$ and $Z \in \mathsf{Id}$.

The meaning of the identifiers is specified by a declaration $\mathcal{D}$ assigning a formula of L_ν to each identifier. When $\mathcal{D}$ is understood we write $Z \stackrel{\text{def}}{=} \varphi$ for $\mathcal{D}(Z) = \varphi$. The K clocks are called *formula clocks* and a formula φ is said to be *closed* if every formula clock x occurring in φ is in the scope of an "x in $\ldots$" operator.

Given a timed transition system $\mathcal{S}$ we interpret the L_ν formulas over an *extended state* $\langle s, u \rangle$, where s is a state of $\mathcal{S}$ and u is a time assignment for K. Informally, $\exists\, \varphi$ holds in an extended state if there is a delay transition leading to an extended state satisfying φ. Thus $\exists$ denotes existential quantification over (arbitrary) delay transitions. Similarly, $\forall\!\!\!\forall$ denotes universal quantification over delay transitions, and $\langle a \rangle$ (resp. $[a]$) denotes existential (resp. universal) quantification over a–transitions. The formula $(x \mathsf{in} \varphi)$ initializes the formula clock x to 0; i.e. an extended state satisfies the formula in case the modified state with x being reset to 0 satisfies φ. Introduced formula clocks are used by formulas of the type $(x + n \bowtie y + m)$, which is satisfied by an extended state provided the values of the named formula clocks satisfies the required relationship. Finally, an extended state satisfies an identifier Z if it satisfies the corresponding declaration (or definition) $\mathcal{D}(Z)$. Formally, the satisfaction relation $\models_{\mathcal{D}}$ between extended states and formulas is defined as the largest relation satisfying the implications of Table 1. We have left out the cases for $\vee$, $[a]$ and $\forall\!\!\!\forall$ as they are immediate duals.

Any relation satisfying the implications in Table 1 is called a *satisfiability* relation. It follows from standard fixpoint theory [22] that $\models_{\mathcal{D}}$ is the union of all satisfiability relations and that the implications in Table 1 in fact are bi–implications for $\models_{\mathcal{D}}$. We say that $\mathcal{S}$ satisfies a closed L_ν formula φ and write $\mathcal{S} \models \varphi$ when $\langle s_0, u \rangle \models_{\mathcal{D}} \varphi$ for any u. Note that if φ is closed, then $\langle s, u \rangle \models_{\mathcal{D}} \varphi$ iff $\langle s, u' \rangle \models_{\mathcal{D}} \varphi$ for any $u, u' \in \mathbf{R}^K$. Similarly, we say that a timed automaton A satisfies a closed L_ν formula φ in case $\mathcal{S}_A \models_{\mathcal{D}} \varphi$. We write $A \models_{\mathcal{D}} \varphi$ in this case.

[9] Bounded in the number of clocks and maximal constant allowed in the satisfying automata.

$$
\begin{array}{rcl}
\langle s,u\rangle \models_{\mathcal{D}} \mathsf{tt} & \Rightarrow & \text{true} \\
\langle s,u\rangle \models_{\mathcal{D}} \mathsf{ff} & \Rightarrow & \text{false} \\
\langle s,u\rangle \models_{\mathcal{D}} \varphi \wedge \psi & \Rightarrow & \langle s,u\rangle \models_{\mathcal{D}} \varphi \ \text{ and } \ \langle s,u\rangle \models_{\mathcal{D}} \psi \\
\langle s,u\rangle \models_{\mathcal{D}} \exists\!\!\exists\, \varphi & \Rightarrow & \exists d \in \mathbf{R}.\ \ \langle s^d, u+d\rangle \models_{\mathcal{D}} \varphi \\
\langle s,u\rangle \models_{\mathcal{D}} \langle a\rangle\, \varphi & \Rightarrow & \exists\, s'.\ \ s \xrightarrow{a} s' \ \text{ and } \ \langle s',u\rangle \models_{\mathcal{D}} \varphi \\
\langle s,u\rangle \models_{\mathcal{D}} x+m \bowtie y+n & \Rightarrow & u(x)+m \bowtie u(y)+n \\
\langle s,u\rangle \models_{\mathcal{D}} x \ \mathsf{in}\ \varphi & \Rightarrow & \langle s,u'\rangle \models_{\mathcal{D}} \varphi \ \text{ where } \ u' = [\{x\} \mapsto 0]u \\
\langle s,u\rangle \models_{\mathcal{D}} Z & \Rightarrow & \langle s,u\rangle \models_{\mathcal{D}} \mathcal{D}(Z)
\end{array}
$$

Table 1. Definition of satisfiability.

Example 4. Consider the following declaration $\mathcal{F}$ of the identifiers X_g and Z_g where $g \in \mathbf{N}$.

$$
\mathcal{F} \;=\; \left\{\ X_g \stackrel{\text{def}}{=} [a]\Big(y \ \mathsf{in}\ Z_g\Big) \quad , \quad Z_g \stackrel{\text{def}}{=} (y > g) \vee \Big([c]\mathsf{ff} \wedge [a]Z_g \wedge [b]Z_g \wedge \forall\!\!\forall Z_g\Big)\ \right\}
$$

That is X_g expresses the property that the accumulated time between an initial a– and a following c–transition must exceed g. Thus for any transition sequence of the form (where $\alpha_i \in \{a,b\}$): $s_0 \xrightarrow{a} t_0 \xrightarrow{\epsilon(d_1)} s_1 \xrightarrow{\alpha_1} t_2 \xrightarrow{\epsilon(d_2)} s_2 \xrightarrow{\alpha_2} \cdots t_n \xrightarrow{\epsilon(d_n)} s_n \xrightarrow{c}$
With $\sum d_i > g$. Now, reconsider the automata A_e, B_d and $C_{d,e}$ of Figure 1 and Examples 1 and 2. Then it may be argued that $C_{d,e} \models_{\mathcal{F}} X_{d+e}$ and (consequently), that $B_d \,|_f\, A_e \models_{\mathcal{F}} X_{d+e}$. □

Combining the parameterized parallel composition with L_ν we are able to express timed bisimilarity between timed transition systems as a single formula. This provides a timed extension of a similar characterization of strong bisimilarity for finite state systems [3]: First we close the action set $\mathcal{A}$ under tagging; i.e. $\mathcal{A}^t = \mathcal{A} \cup \mathcal{A} \times \{l\} \cup \mathcal{A} \times \{r\}$. Now consider the 'interleaving' synchronization function h over $\mathcal{A}^t$ completely defined by $h(a,0) = a_l$ and $h(0,a) = a_r$ where $a \in \mathcal{A}$ (i.e. h is undefined for all other pairs). Consider the following declaration $\mathcal{E}$ of the identifier $\mathcal{Z}$:

$$
\mathcal{Z} \stackrel{\text{def}}{=} \bigwedge_{a \in \mathcal{A}} \Big([a_l]\langle a_r\rangle \mathcal{Z} \ \wedge\ [a_r]\langle a_l\rangle \mathcal{Z}\Big) \ \wedge\ \forall\!\!\forall \mathcal{Z}
$$

Then timed bisimilarity between timed transition systems is characterized as follows [10]:

Theorem 4. Let $\mathcal{S}$ be a timed transition system over $\mathcal{A}$, and let s_1, s_2 be states of $\mathcal{S}$. Then $s_1 \sim s_2$ if and only if $s_1 \,|_h\, s_2 \models_{\mathcal{E}} \mathcal{Z}$.

[10] It may be shown that the speed relation of [20] is characterized in a similar manner by Y defined recursively by $Y \stackrel{\text{def}}{=} \bigwedge_{a \in \mathcal{A}} ([a_1]\exists\!\!\exists\langle a_2\rangle Y \ \wedge\ [a_2]\langle a_1\rangle Y) \ \wedge\ \forall\!\!\forall Y$.

4 Regions

The model-checking problem for L_ν consists in deciding if a given timed automata A satisfies a given specification φ in L_ν. In [16] this problem has been shown decidable using the region technique of Alur and Dill [2, 1]. The region technique provides an abstract interpretation of timed automata sufficiently complete that all information necessary for model–checking with respect to L_ν is maintained, yet finitary and thus enabling standard algorithmic model–checking techniques to be applied.

The basic idea is that, given a timed automaton A, two states (η, v_1) and (η, v_2) which are close enough with respect to their clocks values (we will say that v_1 and v_2 are in the same *region*) can perform the same actions, and two extended states $\langle(\eta, v_1), u_1\rangle$ and $\langle(\eta, v_2), u_2\rangle$ where $v_1\, u_1$ and $v_2\, u_2$ are in the same region, satisfy the same L_ν formulas [11].

In fact the regions are defined as equivalence classes of a relation $\doteq$ over time assignments [13]. Formally, given C a set of clocks and k an integer, we say $u \doteq v$ if and only if u and v satisfy the same conditions of $\mathcal{B}_k(C)$. $[u]$ denotes the region which contains the time assignment u. $\mathcal{R}_k^C$ denotes the set of all regions for a set C of clocks and the maximal constant k. From a decision point of view it is important to note that $\mathcal{R}_k^C$ is finite.

For a region $\gamma \in \mathcal{R}_k^C$, we can define $b(\gamma)$ as the truth value of $b(u)$ for any u in γ. Conversely given a region γ, we can easily build a formula of $\mathcal{B}(C)$, called $\beta(\gamma)$, such that $\beta(\gamma)(u) = \mathtt{tt}$ iff $u \in \gamma$. Thus, given a region γ', $\beta(\gamma)(\gamma')$ is mapped to the value $\mathtt{tt}$ precisely when $\gamma = \gamma'$. Finally, note that $\beta(\gamma)$ itself can be viewed as a L_ν formula.

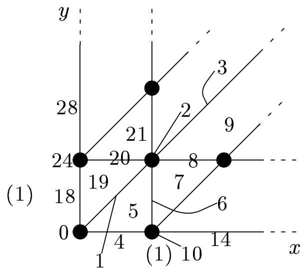
Given a region $[u]$ in $\mathcal{R}_k^C$ and $C' \subseteq C$ we define the following reset operator: $[C' \mapsto 0][u] = [[C' \mapsto 0]u]$. Moreover given a region γ, we can define the successor region of γ (denoted by $succ(\gamma)$): Informally the change from γ to $succ(\gamma)$ correspond to the minimal elapse of time which can modify the enabled actions of the current state(a formal definition is given in [16]).

We denote by γ^l the l^{th} successor region of γ (i.e. $\gamma^l = succ^l(\gamma)$). From each region γ, it is possible to reach a region γ' s.t. $succ(\gamma') = \gamma'$, and we denote by l_γ the required number of step s.t. $\gamma^{l_\gamma} = succ(\gamma^{l_\gamma})$.

Example 5. Figure 3 gives an overview of the set of regions defined by two clocks x and y, and the maximal constant 1. In this case there are 32 different regions, of which only 17 are numbered in the figure. Corresponding $\mathcal{B}(C)$–formulas as well as successor regions are indicated for some of the regions. In general successor regions are determined by following 45^o lines upwards to the right. □

Given a timed automata $A = \langle \mathcal{A}, N, \eta_0, C, E\rangle$, let k_A be the maximal constant occurring in the enabling condition of the edges E. Then for any $k \geq k_A$ we can define a symbolic semantics of A over symbolic states $[\eta, \gamma]_A$ where $\eta \in N$ and $\gamma \in \mathcal{R}_k^C$ as follows: for any $[\eta, \gamma]$ we have $[\eta, \gamma]_A \xrightarrow{a} [\eta', \gamma']_A$ iff $\exists\, u \in \gamma,\ \langle \eta, u\rangle_A \xrightarrow{a} \langle \eta', u'\rangle_A$ and $u' \in \gamma'$.

[11] W.l.o.g. we will in all the following assume that K and C are disjoint.



$$succ(\gamma_0) = \gamma_1 \qquad \beta(\gamma_1) = \begin{array}{l} 0 < x < 1 \wedge \\ 0 < y < 1 \wedge \\ y = x \end{array}$$

$$succ(\gamma_5) = \gamma_6$$

$$\ldots$$

$$\beta(\gamma_0) = (x = 0 \wedge y = 0) \qquad \beta(\gamma_5) = \begin{array}{l} 0 < x < 1 \wedge \\ 0 < y < 1 \wedge \\ y < x \end{array}$$

$$\beta(\gamma_6) = (0 < y < 1 \wedge x = 1)$$

Fig. 3. $\mathcal{R}_k^C$ with $C = \{x, y\}$ and $k = 1$

Consider now L_ν with respect to formula clock set K and maximal constant k_L. Also consider a given timed automata $A = \langle N, \eta_0, C, E \rangle$ (s.t. K and C are disjoint). Then an *extended symbolic state* is a pair $[\eta, \gamma]_{A^+}$ where $\eta \in N$ and $\gamma \in \mathcal{R}_k^{C \cup K}$ with $k = max(k_A, k_L)$. Using the finitary symbolic semantics of timed automata a symbolic interpretation of L_ν formulas over extended symbolic states is defined in [16] faithfully reflecting the standard interpretation in Table 1. This symbolic interpretation provides the basis for decidability of model–checking for L_ν, and consequently (due to Theorem 4) for decidability of timed bisimilarity between timed automata.

5 Quotienting

Given an L_ν formula φ, and two timed transition systems $\mathcal{S}_1$ and $\mathcal{S}_2$ we aim at constructing a formula (called the *quotient*) $\varphi/_{\!f}\, \mathcal{S}_2$ such that

$$\mathcal{S}_1 \,|_f\, \mathcal{S}_2 \models \varphi \quad \text{if and only if} \quad \mathcal{S}_1 \models \varphi/_{\!f}\, \mathcal{S}_2 \tag{3}$$

The bi–implication indicates that we are moving parts of the parallel system into the formula. Clearly, if the quotient is not much larger than the original formula, we have simplified the task of model checking, as the (symbolic) semantics of $\mathcal{S}_1$ is significantly smaller than that of $\mathcal{S}_1 \,|_f\, \mathcal{S}_2$.

In general it will not be possible to express within L_ν such a quotienting formula. However, whenever $\mathcal{S}_2$ is described using a timed automata we are able to express the quotient using the (extended) symbolic semantics of automata. More precisely, whenever φ is a formula over K and A is a timed automaton over C, we define quotient formulas $\varphi/_{\!f}\, [\eta, \gamma]$ over $C \cup K$, where $[\eta, \gamma]$ is an extended symbolic state. For η_0 the initial node of A and γ_0 the initial region, $\varphi/_{\!f}\, [\eta_0, \gamma_0]$ will express the sufficient and necessary requirement to a timed transition system $\mathcal{S}$ in order that $\mathcal{S} \,|_f\, \mathcal{S}_A$ satisfies φ — see also Corollary 6 below. The quotient construction is defined structurally in Table 2. We have left out the cases for $[a]$, $\vee$, and $\forall$ as they are immediate duals. The Table uses the following notation [12]:

[12] $\gamma_{|C}$ (resp. $\gamma_{|K}$) denotes the set of time-assignments in γ restricted to the automata (resp. formula) clocks.

$$\mathsf{tt}/_f\,[\eta,\gamma] = \mathsf{tt} \qquad \mathsf{ff}/_f\,[\eta,\gamma] = \mathsf{ff} \qquad Z/_f\,[\eta,\gamma] = Z^{[\eta,\gamma]}$$

$$(\varphi_1 \wedge \varphi_2)/_f\,[\eta,\gamma] = (\varphi_1/_f\,[\eta,\gamma]) \wedge (\varphi_2/_f\,[\eta,\gamma])$$

$$\langle a\rangle\varphi/_f\,[\eta,\gamma] = \bigvee_{b,[\eta',\gamma']\in E(\eta,\gamma,a)} \langle b\rangle\left(r_{\gamma'}\ \mathsf{in}\ \varphi/_f\,[\eta'\gamma']\right)$$

$$\exists\varphi/_f\,[\eta,\gamma] = \exists\Big(\bigvee_{l=0\ldots l_\gamma} \beta(\gamma^l) \wedge \varphi/_f\,[\eta,\gamma^l]\Big)$$

$$(x + c \bowtie y + d)/_f\,[\eta,\gamma] = (x + c \bowtie y + d)(\gamma)$$

$$(x\ \mathsf{in}\ \varphi)/_f\,[\eta,\gamma] = x\ \mathsf{in}\ \left(\varphi/_f\,[\eta,\gamma']\right) \text{ where } \gamma' = [\{x\}\to 0]\gamma$$

Table 2. Structural definition of quotient, $\varphi/_f\,[\eta,\gamma]$.

$b, [\eta'\gamma'] \in E(\eta,\gamma,a)$ iff $[\eta,\gamma_{|C}] \xrightarrow{c} [\eta',\gamma'_{|C}]$, $\gamma'_{|K} = \gamma_{|K}$ and $f(b,c) \simeq a$ for some c. We denote by r_γ the set of C clocks which has the value 0 in γ. Moreover $(r\,\mathsf{in}\,\varphi)$ is an abbreviation for $(x_1\,\mathsf{in}\,(x_2\,\mathsf{in}\ldots(x_n\,\mathsf{in}\,\varphi)))$ whenever r is $\{x_1,\ldots,x_n\}$. Finally, $\langle 0\rangle\varphi = [0]\varphi = \varphi$.

Note that the quotient construction for identifiers introduces new identifiers of the form $Z^{[\eta,\gamma]}$, where $Z \in \mathsf{Id}$. The definition of these are collected in the (quotient) declaration $\mathcal{D}_A$ given by: $Z^{[\eta,\gamma]} \stackrel{\mathrm{def}}{=} \mathcal{D}(Z)/_f\,[\eta,\gamma]$

The following Theorem and Corollary shows that the quotient construction of Table 2 satisfies the requirements of (3). A proof of Theorem 5 is sketched in [15].

Theorem 5. Let $\mathcal{S} = \langle S, s_0, \longrightarrow\rangle$ be a timed transition system, let $A = \langle N, \eta_0, C, E\rangle$ be a timed automaton, and let φ be an L_ν formula over clock set K, all over an action set $\mathcal{A}$. Then for any $s \in S$, $\eta \in N$, $v \in \mathbf{R}^C$, and $u \in \mathbf{R}^K$:

$$\Big\langle\, s\,|_f\,(\eta, v)\,,\,u\,\Big\rangle \models_{\mathcal{D}} \varphi \quad \textit{if and only if} \quad \langle\, s, v\,u\,\rangle \models_{\mathcal{D}_A} \varphi/_f\,\Big[\eta,[v\ u]\Big]$$

Corollary 6. Let $\mathcal{S}$ be a timed transition system, let A be a timed automaton with clock set C, and let φ be a closed L_ν formula over clock set K, all over an action set $\mathcal{A}$. Then: $\mathcal{S}\,|_f\,\mathcal{S}_A \models_{\mathcal{D}} \varphi$ *if and only if* $\mathcal{S} \models_{\mathcal{D}_A} \varphi/_f\,A$

where $\varphi/_f\,A$ abbreviates $\varphi/_f\,[\eta_0,\gamma_0]$ with η_0 being the initial node of A and γ_0 the initial region over $C \cup K$.

Example 6. Recall the timed automata and declaration from Examples 1 and 4. The quotient $X_1/_f\,A_1$ describes the sufficient and necessary requirement to a timed transition system $\mathcal{S}$ in order that $\mathcal{S}\,|_f\,\mathcal{S}_{A_1}$ satisfies X_1. Clearly, we expect the timed automata B_0 to satisfy this property. Now using a CAML prototype implementation of the quotient construction $X_1/_f\,A_1$ is computed. The definition of $X_1/_f\,A_1$ is found in the quotient declaration $\mathcal{F}_{A_1}$ part of which is given below:

$$X_1/_f\,A_1 \stackrel{\mathrm{def}}{=} [a]\,y\ \mathsf{in}\ Z_1^{[\eta_0,\gamma_0]}$$

$$Z_1^{[\eta_0,\gamma_0]} \stackrel{\text{def}}{=} \mathsf{ff} \vee \Big[\mathsf{tt} \wedge ([b]x \text{ in } Z_1^{[\eta_1,\gamma_0]}) \wedge ([a]Z_1^{[\eta_0,\gamma_0]}) \wedge \forall\!\!\!\forall\Big(\beta(\gamma_0) \;\Rightarrow\; Z_1^{[\eta_0,\gamma_0]} \wedge$$

$$\beta(\gamma_1) \;\Rightarrow\; Z_1^{[\eta_0,\gamma_1]} \wedge \beta(\gamma_2) \;\Rightarrow\; Z_1^{[\eta_0,\gamma_2]} \wedge \beta(\gamma_3) \;\Rightarrow\; Z_1^{[\eta_0,\gamma_3]}\Big)\Big]$$

$$Z_1^{[\eta_1,\gamma_0]} \stackrel{\text{def}}{=} \mathsf{ff} \vee \Big[\mathsf{tt} \wedge \mathsf{tt} \wedge ([a]Z_1^{[\eta_1,\gamma_0]}) \wedge \forall\!\!\!\forall\Big(\beta(\gamma_0) \;\Rightarrow\; Z_1^{[\eta_1,\gamma_0]} \wedge \beta(\gamma_1) \;\Rightarrow\; Z_1^{[\eta_1,\gamma_1]} \wedge$$

$$\beta(\gamma_2) \;\Rightarrow\; Z_1^{[\eta_1,\gamma_2]} \wedge \beta(\gamma_3) \;\Rightarrow\; Z_1^{[\eta_1,\gamma_3]}\Big)\Big]$$

The quotient declaration $\mathcal{F}_{A_1}$ contains in total definitions of 96 identifiers. We expect $X_1/_f A_1$ to express that the accumulated time between an initial a–action and a following b–action must be strictly greater than 0 as described by the following property U_0:

$$U_0 \stackrel{\text{def}}{=} [a]\Big(y \text{ in } V_0\Big) \qquad\qquad V_0 \stackrel{\text{def}}{=} (y > 0) \vee \Big([b]\mathsf{ff} \wedge [a]V_0 \wedge \forall\!\!\!\forall V_0\Big)$$

In the next section we will present effective minimization strategies, which essentially transforms the large quotient declaration $\mathcal{F}_{A_1}$ into the small yet equivalent declaration above. □

Now, let π be the synchronization function completely specified by $\pi(0, a) = a$ for all $a \in \mathcal{A}$ (i.e. the left argument to π is completely ignored). Then it is easy to see that for any timed transition system $\mathcal{S}$, $\mathcal{S}_A$ and $\mathcal{S} |_\pi \mathcal{S}_A$ satisfy the same formulas. In particular $\mathcal{S}_A$ and $\mathcal{S}_{\mathbf{1}} |_\pi \mathcal{S}_A$ satisfy the same formula, where $\mathbf{1}$ is the 0–clock timed automaton with just one (initial) node and no edges. Using this observation, the quotient construction can be used to obtain alternative model–checking algorithms for L_ν as follows:

Corollary 7. Let A be a timed automaton with clock set C, and let φ be a closed L_ν formula over clock set K, all over an action set $\mathcal{A}$. Then:

$$A \models_{\mathcal{D}} \varphi \quad \textit{if and only if} \quad \mathbf{1} \models_{\mathcal{D}_A} \varphi/_\pi A \quad \textit{if and only if} \quad \varphi/_\pi A \Leftrightarrow \mathsf{tt}$$

Due to the projective nature of π it is clear that $\varphi/_\pi A$ contains no action modalities, and it is easy to build a special purpose model checker for the simple automata $\mathbf{1}$.

Example 7. Recall once more the timed automata and declaration from Examples 1 and 4. From these Examples we expect that $C_{0,1}$ satisfies the property X_1. Now, using the Corollary 7 we may verify this by showing that the quotient formula $X_1/_\pi C_{0,1}$ is either valid or satisfied by $\mathbf{1}$. The quotient declaration associated to $X_1/_\pi C_{0,1}$ computed by the CAML prototype (with subsequent application of boolean simplifications) contains 8 identifiers without any $\langle a\rangle$ or $[a]$ operator.

□

Finally, combining the characterization of timed bisimulation in Theorem 4 with the quotient construct of Table 2, we can for any timed automaton A construct a *characteristic* L_ν formula uniquely characterizing the automaton up to timed bisimilarity in the following manner: Let $\mathcal{E}$, $\mathcal{Z}$ and h be as in Theorem 4. Then for any timed transition system $\mathcal{S}$ the following holds:

$$\mathcal{S} \sim \mathcal{S}_A \quad \text{if and only if} \quad \mathcal{S} \models_{\mathcal{E}} \mathcal{Z}/_{h}\,[\eta_0, \gamma_0]$$

where η_0 is the initial node of A and γ_0 is the initial region over the clocks of A (note that Z is an L_ν formula over the empty set of clocks). This provides an alternative characteristic formula construction compared with [16].

6 Minimizations

It is evident from the examples in the previous section that repeated quotienting leads to an explosion in the formula. A similar phenomena was observed by Andersen for the quotient construction of modal μ–calculus formulas with respect to finite–state systems. The crucial observation by Andersen is that simple and cost effective transformations of the formulas in practice often lead to significant reductions.

We have implemented in CAML the quotient construction of the previous section as well as (simplified versions of some of) Andersen's minimization strategies. In the investigated examples the minimization strategies lead to dramatic reductions. Below we describe the transformations considered in terms of transformations on formulas and declarations (defining equations).

Reachability: When considering an initial quotient formula $\varphi/_{f}\,[\eta_0, \gamma_0]$ not all identifiers of $\mathcal{D}_A$ may be relevant or reachable. In our CAML implementation an "on–the–fly" technique insures that only the reachable part of $\mathcal{D}_A$ is generated.

Boolean Simplification: Formulas may be simplified using the following simple boolean equations and their duals: $\mathsf{ff} \wedge \varphi \equiv \mathsf{ff}$, $\mathsf{tt} \wedge \varphi \equiv \varphi$, $\langle a\rangle \mathsf{ff} \equiv \mathsf{ff}$, $\exists \mathsf{ff} \equiv \mathsf{ff}$, $x \ \mathsf{in}\ \mathsf{ff} \equiv \mathsf{ff}$, $\langle a\rangle\varphi \wedge [a]\mathsf{ff} \equiv \mathsf{ff}$.

Constant Propagation: Identifiers which are declared to either tt or ff may be removed while substituting their definitions in the declaration of all other identifiers.

Trivial Equation Elimination: Equations of the form $X \stackrel{\mathrm{def}}{=} [a]X$ are easily seen to have $X = \mathsf{tt}$ as solution. More generally, whenever $X \stackrel{\mathrm{def}}{=} \varphi$ where $\varphi[\mathsf{tt}/X]$ [13] can be simplified to tt, we can perform a removal of the identifier X provided the value tt is propagated to all uses of X (as under Constant Propagation).

Equivalence Reduction: If two identifiers X and Y are equivalent (i.e. are satisfied by the same timed transition systems) we may collapse them into a single

[13] $\varphi[\mathsf{tt}/X]$ is the formula obtained by substituting all occurrences of X in φ with the formula tt.

identifier. To obtain a cost effective strategy we approximate equivalence of identifiers with the following check: whenever $X \stackrel{\text{def}}{=} \varphi$ and $Y \stackrel{\text{def}}{=} \vartheta$ such that $\varphi[Y/X]$ is syntactically identical to $\vartheta[Y/X]$ we conclude that X and Y are equivalent and may be identified.

We apply the above transformation strategies repeatedly on quotient formulas and declarations. As can be seen in Table 2, quotienting transforms atomic propositions of the form $x + c \bowtie y + d$ into either $\mathtt{tt}$ or $\mathtt{ff}$, thus yielding high applicability of Boolean Simplification and Constant Propagation. Also, quotient versions of the same original formula tend to have same structure thus making applicability of Equivalence Reduction high.

In our CAML prototype implementation we have implemented the above reported very simple strategies for Trivial Equation Elimination and Equivalence Reduction. Generalized and more sophisticated (yet still efficient) strategies can or course easily be given. However, even with the implemented simple versions we obtain a very high degree of reduction as observed in the following examples.

Example 8. Recall Example 6. Applying the minimization strategies of the CAML prototype we find that $X_1 /\!\!_f\, A_1$ is equivalent to Y_0 with the following definition:

$$
\begin{aligned}
Y_0 &\stackrel{\text{def}}{=} [a]y \ \mathsf{in}\ Y_1 \\
Y_1 &\stackrel{\text{def}}{=} [b]\mathtt{ff} \wedge [a]Y_1 \wedge \forall\!\!\!\forall \Big(\beta(\gamma_0) \ \Rightarrow\ Y_1 \wedge (\beta(\gamma_1) \vee \beta(\gamma_2)) \ \Rightarrow\ Y_2\Big) \\
Y_2 &\stackrel{\text{def}}{=} [a]Y_2 \wedge \forall\!\!\!\forall \Big(\beta(\gamma_0) \ \Rightarrow\ Y_1 \wedge (\beta(\gamma_1) \vee \beta(\gamma_2)) \ \Rightarrow\ Y_2\Big)
\end{aligned}
$$

During minimization only 23 identifiers of $\mathcal{F}_{A_1}$ were found to be reachable from $X_1 /\!\!_f\, A_1$, 14 respectively 3 of which were found to be equivalent to $\mathtt{tt}$ respectively $\mathtt{ff}$. The remaining 6 identifiers were finally partitioned into 3 classes. Though dramatically reduced, the declaration leaves room for *one* additional simplification as it may be observed that $(\beta(\gamma_1) \vee \beta(\gamma_2)) \ \Rightarrow\ Y_2$ is equivalent to $\mathtt{tt}$. With this observation we finally obtain:

$$
Y_0' \stackrel{\text{def}}{=} [a]y \ \mathsf{in}\ Y_1' \qquad\qquad Y_1' \stackrel{\text{def}}{=} [b]\mathtt{ff} \wedge [a]Y_1' \wedge \forall\!\!\!\forall \Big(\beta(\gamma_0) \ \Rightarrow\ Y_1'\Big)
$$

which clearly meets the expectations of 6. Minimization of the quotient formula $X_7 /\!\!_f\, A_{10}$ leads directly to the formula $\mathtt{tt}$ indicating that for any timed transition system $\mathcal{S}$ the composition $\mathcal{S} \,|_f\, \mathcal{S}_{A_{10}}$ satisfies X_7. Intuitively this is clear as the component A_{10} by itself ensures the delay required by X_7. The corresponding quotient declaration contains 3498 identifiers 617 of which were found to be reachable. Subsequently all these were simplified to $\mathtt{tt}$. $\square$

Example 9. Recall Example 7. Using the minimization strategies of the CAML prototype we find directly that $X_1 /\!\!_\pi\, C_{0,1}$ simplifies to $\mathtt{tt}$ thus implying that $C_{0,1}$ satisfies the property X_1. Similarly, minimization of the quotient formula $X_2 /\!\!_\pi\, C_{0,1}$ yields $\mathtt{ff}$ confirming that $C_{0,1}$ does not satisfy X_2! $\square$

Example 10. Now we want to confirm that B_0 does indeed satisfy the requirement $X_1 /_f A_1$ and hence that $B_0 \,|_f\, A_1$ satisfies X_1. Using the equivalent, minimized formula Y_0 from Example 8 it suffices according to Corollary 7 to verify validity of the quotient formula $Y_0 /_\pi B_0$. The CAML prototype confirms this through immediate minimization to $\mathbf{tt}$. $\square$

7 Conclusion

This paper has presented the basis for a compositional model checking technique for real–time systems. Based on initial experiments with the CAML prototype we conjecture that compositional model checking will prove not only a feasible but also an efficient technique for real–time systems. However, it is clear that many more experiments must be performed before the conjecture can be finally settled, and in this process we will need to extend our prototype with more sophisticated minimization strategies as the success of compositional model checking is completely determined by these.

Our work on quotient formulas extends that of [7], where a quotient construction for 1-clock automata has been given. Moreover it can be generalized trivially to logics with minimal fixpoint constructs (such as T_μ in [13]) as quotienting is easily seen to distribute over negation.

Acknowledgement

The authors would like to thank Henrik Reif Andersen for interesting and enlightening discussions on the topic of compositional (partial) model–checking.

References

1. R. Alur, C. Courcoubetis, and D. Dill. Model–checking for Real–Time Systems. In *Proceedings of Logic in Computer Science*, pages 414–425. IEEE Computer Society Press, 1990.
2. R. Alur and D. Dill. Automata for Modelling Real–Time Systems. *Theoretical Computer Science*, 126(2):183–236, April 1994.
3. H. R. Andersen. A Polyadic Modal μ–calculus. Id–tr: 1994–145, Department of Computer Science, Technical University of Denmark, 1994.
4. H. R. Andersen. Partial Model Checking. *To appear in Proceedings of LICS'95*, 1995.
5. H. R. Andersen, C. Stirling, and G. Winskel. A Compositional Proof System for the Modal Mu–Calculus. *Logic in Computer Science*, 1994.
6. H. R. Andersen and G. Winskel. Compositional checking of satisfaction. *Formal Methods in Systems Design*, 1992. To appear.
7. J. H. Andersen, K. J. Kristoffersen, K. G. Larsen, and J. Niedermann. Automatic Synthesis of Real Time Systems. *Lecture Notes in Computer Science*, 1995. To appear in Proceedings of ICALP'95.
8. J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic Model Checking: 10^{20} states and beyond. *Logic in Computer Science*, 1990.

9. E. M. Clarke, T. Filkorn, and S. Jha. Exploiting Symmetry in Temporal Logic Model Checking. *Lecture Notes in Computer Science*, 697, 1993.
10. E. M. Clarke, O. Grümberg, and D. E. Long. Model Checking and Abstraction. *Principles of Programming Languages*, 1992.
11. E. A. Emerson and C. S. Jutla. Symmetry and Model Checking. *Lecture Notes in Computer Science*, 697, 1993.
12. P. Godefroid and P. Wolper. A Partial Approach to Model Checking. *Logic in Computer Science*, 1991.
13. T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. In *Logic in Computer Science*, 1992.
14. H. Hüttel and K. G. Larsen. The use of static constructs in a modal process logic. *Lecture Notes in Computer Science, Springer Verlag*, 363, 1989.
15. F. Laroussinie and K. G. Larsen. Compositional Model Checking of Real Time Systems. Technical Report RS–95–19, BRICS, 1995. Accessible through WWW: http://www.brics.aau.dk/BRICS.
16. F. Laroussinie, K. G. Larsen, and C. Weise. From Timed Automata to Logic — and Back. 1995. To appear in Proceedings of MFCS'95.
17. K.G. Larsen. *Context–Dependent Bisimulation Between Processes*. PhD thesis, University of Edinburgh, Mayfield Road, Edinburgh, Scotland, 1986.
18. K.G. Larsen and L. Xinxin. Compositionality through an operational semantics of contexts. *Journal of Logic and Computation*, 1(6):761–795, 1991.
19. R. Milner. *Communication and Concurrency*. prentice, Englewood Cliffs, 1989.
20. F. Moller and C. Tofts. Relating Processes with Respect to Speed. Technical Report ECS–LFCS–91–143, Department of Computer Science, University of Edinburgh, 1991.
21. D. Park. Concurrency and automata on infinite sequences. In *Proceedings of 5th GI Conference*, volume 104 of *Lecture Notes in Computer Science, Springer Verlag*, Berlin, 1981. Springer.
22. A. Tarski. A lattice–theoretical fixpoint theorem and its applications. *Pacific Journal of Math.*, 5, 1955.
23. A. Valmari. A Stubborn Attack on State Explosion. *Theoretical Computer Science*, 3, 1990.
24. W. Yi. A Calculus of Real Time Systems. *Lecture Notes in Computer Science*, 458, 1990. In Proceedings of CONCUR.

This article was processed using the LaTeX macro package with LLNCS style