Bull. Nov. Comp. Center, Comp. Science, 3 (1995), 19–37

On compositional model checking in the modal Mu-calculus and its extension with multiple clocks*

S.A. Berezin

This paper includes several results concerning with compositional model checking (CMC) for finite-state systems with multiple clocks. We describe a model of time [6], and a method of reducing a timed (infinite) model to a finite untimed model. We also present a new Finite Process Algebra (FPA) and its extension by multiple clocks. Further we give "Pure" Compositional Model Checking (PCMC) for the V-free fragment of the modal μ-calculus with its generalization for the full μ-calculus as a) tableau-based model checking proof system and b) direct decomposition of a formula. The worst case complexity analysis is done for all variants. In particular, we extend the result of H. Andersen in [2] on the complexity of model checking to the FPA and μ-calculus with more general simultaneous fixed points. We show also that any finite Labeled Transition System (LTS) with 2^n states can be equivalently represented by a compositional system of processes with the sum of all their states $2n$. It means that for any logic with at least linear model checking procedure on the size of a model, no efficient compositional proof system can be built, that does not use any other special properties except hierarchy.

1. Introduction

Model Checking (MC) appeares to be a very useful and powerful tool for checking correctness of big and complicated systems. One of the most expressive logics used to describe properties of such systems is the modal μ-calculus [23]. Hüttel in [22] proved that the expressiveness of the logic is equivalent to the expressive power of SnS. Interesting that SnS is decidable only in its canonical model, while the μ-calculus is decidable in an arbitrary model.

The model checking problem for μ-calculus is in $NP \cap$co-NP [17, 8]. Algorithms with rather low complexity were developed, for example $O(|T| \cdot |\Phi| \cdot (|S| \cdot \frac{|\Phi|}{\mathrm{ad}(\Phi)})^{\mathrm{ad}(\Phi)-1})$, where $|T| = |S|+| \rightarrow |+|I|$ is the size of a LTS , and $\mathrm{ad}(\Phi)$ is the *alternation depth* of nested fixed points μ and ν in the formula Φ [14]. H. Andersen obtained a close estimate: $O(|\Phi|^k \cdot |S|^{k-1} \cdot |T|)$ in [2],

*Partially supported by the International Association for the Promotion and Cooperation with Scientists from the Independent States of the Former Soviet Union (INTAS) Contract No. 1010-CT93-0048.

where $k = \max\{\mathrm{ad}(\Phi), 1\}$. In [24] this estimate was refined: the power of an exponential decreases twice as much, i.e., for formulas with $\mathrm{ad}(\Phi) \geq 2$ the complexity became better approximately as a square root. For the variety of more popular logics like CTL and PDL, polynomial and even linear estimates are known [8]. Polynomial estimates were also obtained for two special fragments of μ-calculus [17], where restrictions are made on the use of conjunction. From [14] polynomial estimates follow also for fragments of μ-calculus with $\mathrm{ad}(\Phi) \leq \mathrm{const}$ [6].

But the direct application of even linear algorithms is impossible, if a system has more than $10^5 \div 10^6$ states. A number of methods were developed, which indeed allow to verify properties of large systems with the number of reachable states till 10^{100} and beyond. The most significant was the discovery of Ordered Binary Decision Diagrams (OBDDs) and their efficient implementation [10, 11]. But still for a wide class of systems and properties this technique is not applicable because of the exponential growth of the size of OBDDs. So, the use of additional properties of a model is very helpful. For instance, in [30] the decrease of the number of reachable states is obtained by identifying stuttering-equivalent states. Since the exact solution of the problem is very computationally expensive, an approximation based on the use of partial orders is offered. The use of symmetry [12, 18] also benefits a lot, when a model clearly exhibits repeated fragments. Though some difficulties appear, for example, with choosing a representative state from the set of equivalent states. Very interesting works in my opinion are [15, 16] and [4]. The set of states is split into classes of states indistinguishable by subformulas of a concrete formula being checked. Since for any CTL formula of the size n there exist models of the size 2^n, in which all possible evaluations of the formula are realized, hence it is natural to expect that short formulas can be checked very efficiently even on very large models.

Incremental Model Checking [27] seems to be also interesting. When we know the evaluation of a formula in the initial model M, we can compute the evaluation of the same formula in a slightly changed model M', which differs from M only by the *set of changes* Δ. In some cases this recomputation can be done with just a linear complexity on the size of Δ. It could be very convenient in debugging of a system, when we make small corrections and want to know the consequences quickly.

The model of *periodic* (but finite!) sets' representation was introduced in [9]. Authors offer to store the set of states as a system of equations $x \equiv a(\mathrm{mod}\ n)$ with restrictions $M \leq x \leq N$. This representation allows us to perform set-theoretic operations for such sets symbolically.

But presently the most promising method seems to be Compositional Model Checking (CMC). One of the first works in this direction is [28], where the compositional proof system for Hennessy-Milner logic and CCS and SCCS was developed. The logical completion of the work, containing

a complete proof system for a generalized version of SCCS and the full μ-calculus is [3]. An attempt of generalization of [28] without loss of efficiency was [7]. But, as it is shown below, Pure CMC is a very limited approach. The most promising, in the author's opinion, are such topics as combining CMC with other methods of fighting the state explosion problem, for example, the reduction of a model in [4] and Theorem Proving [21, 19, 20]

Section 2 describes the syntax and semantics of the modal μ-calculus for an arbitrary Kripke structure. In Section 3 we introduce the notion of Finite Process Algebra and its extension to FPA with multiple clocks. Section 4 is concerned with the method of "Pure" CMC with complexity estimates. The limits of the method will also be revealed for sufficiently expressive logics (Subsection 4.5). In particular, we will consider two different approaches to CMC (Subsections 4.1 and 4.3), the generalization of the first one to the full μ-calculus using Tableau-based Model Checking (Subsection 4.2), and will build another general efficient MC algorithm for FPA and the full μ-calculus (Subsection 4.4).

2. μ-calculus: syntax and semantics

Assume we have infinite and disjoint alphabets of propositional constants $\{P, \ldots\}$, propositional variables $\{X, Y, \ldots\}$ and action symbols $\mathfrak{A} = \{a, b, \ldots\}$. Then, formulas of the Modal μ-Calculus are the following:

1) P, where P is a propositional constant;
2) X, where X is a propositional variable;
3) $\neg\Phi$, $\Phi_1 \vee \Phi_2$, $\Phi_1 \wedge \Phi_2$;
4) $\langle a \rangle \Phi$, $[a]\Phi$, where a is an action symbol;
5) $\mu X.\Phi$, $\nu X.\Phi$, if X occurs in Φ positively only;
6) there are no formulas other than those constructed by the rules 1–5.

Here Φ, Φ_1 and Φ_2 are formulas of the μ-calculus. A variable occurs in a formula positively, if it is in the scope of even number of negations.

A Kripke structure is a triple $M = (S, \rightarrow, I)$, where S is a set of states, $\rightarrow \subseteq S \times \mathfrak{A} \times S$ is a transition relation, and I is an interpretation of propositional constants and variables by the sets of states: $I(P) \subseteq S$. We extend I to the set of all formulas in the following way:

1) $I(\neg\Phi) = S \setminus I(\Phi)$;
   $I(\Phi_1 \vee \Phi_2) = I(\Phi_1) \cup I(\Phi_2)$;
   $I(\Phi_1 \wedge \Phi_2) = I(\Phi_1) \cap I(\Phi_2)$;
2) $I(\langle a \rangle \Phi) = \{s \in S \mid \exists s' \in I(\Phi) : s \xrightarrow{a} s'\}$;
   $I([a]\Phi) = \{s \in S \mid \forall s' \in S : (s \xrightarrow{a} s') \Rightarrow s' \in I(\Phi)\}$;

3) $I(\mu X.\Phi(X)) = \bigcap\{L \subseteq S \mid I(\Phi(X))[X \leftarrow L] \subseteq L\}$
$I(\nu X.\Phi(X)) = \bigcup\{L \subseteq S \mid L \subseteq I(\Phi(X))[X \leftarrow L]\}$.

Let us note that in μ-calculus De Morgan's laws and the following identities hold: $\langle a\rangle \Phi \equiv \neg[a]\neg\Phi$, $[a]\Phi \equiv \neg\langle a\rangle\neg\Phi$, $\mu X.\Phi(X) \equiv \neg\nu X.\neg\Phi(\neg X)$ and $\nu X.\Phi(X) \equiv \neg\mu X.\neg\Phi(\neg X)$. Therefore, all negations always can be moved to the atomic formulas. Note, that the variables bounded by fixed-point constructors will occur in corresponding fixed-point subformulas without negations. So we assume below that formulas no longer contain negations.

3. Finite process algebra

Definition 1. Let the triple $G = (S, \rightarrow, \mathfrak{A})$ be the transition graph of a finite process, where S is the finite set of states (the nodes of the graph G), $\mathfrak{A}$ is the alphabet of action symbols with the special *idling* action $* \in \mathfrak{A}$, and $\rightarrow \subseteq S \times \mathfrak{A} \times S$ is the transition relation, which maps each symbol from the alphabet of actions to the set of edges of the graph G. For $(s, a, s') \in \rightarrow$ we will write $s \xrightarrow{a} s'$, and $\xrightarrow{a} \subseteq S \times S$ will denote an obvious binary relation for the action a. The graph G should contain the total unit relation $\xrightarrow{*} = \{(s, s) \mid s \in S\}$.

Definition 2. Let the pair $p = (G_p, s_p)$ be an atomic process, where $G_p = (S_p, \rightarrow_p, \mathfrak{A}_p)$ is the transition graph of a process p, and $s_p \in S_p$ is the initial state of this process.

We will write $p \xrightarrow{a} p'$ if $G_p = G_{p'}$ and $s_p \xrightarrow{a} s_{p'}$.

Definition 3. Processes are defined by the grammar (in the spirit of [3]):

$$p ::= p_A \mid p_0 \times p_1 \mid p \restriction \Lambda \mid p\{\Xi\},$$

where p_A is an atomic process. The operators are called product, restriction and relabelling, respectively. The restricting set Λ must always contain $*$, and $\Xi : \mathfrak{A} \rightarrow \mathfrak{A}$ must respect $*$: $\Xi(*) = *$.

The semantics of the transition relation $p \xrightarrow{a} q$ is extended to arbitrary p and q as the least relation satisfying the rules in Figure 1.

The set of states of a compound process is a Cartesian product $S_{p\times q} = S_p \times S_q = \{(s_1, s_2) \mid s_1 \in S_p, s_2 \in S_q\}$. The sets of states of a restricted and relabelled processes remain the same.

The set of actions of a compound process is a Cartesian product $\mathfrak{A}_{p\times q} = \mathfrak{A}_p \times \mathfrak{A}_q = \{ab \mid a \in \mathfrak{A}_p, b \in \mathfrak{A}_q\}$. The operators of the product of processes

"×" and actions "·" are neither commutative nor assotiative. The only rule is $* \cdot * = *$.

Now we can formulate the problem of verification. The problem is whether a formula Φ is true for a process p. We will denote this by $p \models_{M_p} \Phi$. We say that a sequent $p \models_{M_p} \Phi$ is valid iff the corresponding problem decision is positive.

$$p \xrightarrow{*} p \qquad \frac{p \xrightarrow{\alpha} p' \; q \xrightarrow{\beta} q'}{p \times q \xrightarrow{\alpha \times \beta} p' \times q'}$$

$$\frac{p \xrightarrow{\alpha} p'}{p\{\Xi\} \xrightarrow{\beta} p'\{\Xi\}}, \; \Xi(\alpha) = \beta \qquad \frac{p \xrightarrow{\alpha} p'}{p \restriction \Lambda \xrightarrow{\alpha} p' \restriction \Lambda}, \; \alpha \in \Lambda.$$

Figure 1. Operational rules for FPA

3.1. Models with multiple clocks

In this section we introduce a model of discrete multiple clocks and a special infinite Kripke structure representing this model of time. We refer to the previous work [6] for its correctness and the decidability of the MC problem.

First we give an informal description of a transition system with multiple clocks. This system is the generalization of the transition system with real time described in [26, 5].

Consider a transition graph $G = (S, \rightarrow, \mathfrak{A})$. To each action $b \in \mathfrak{A}$ two constant vectors $\vec{l}, \vec{u} \in (\mathcal{N} \cup \{\infty\})^m$ are assigned, where $\vec{l} \leq \vec{u}$ component-wise. They are lower and upper time bounds of a duration interval for the action. We will denote it by ${}_{\vec{l}}b^{\vec{u}}$. The global vector variable $\vec{t}$ ranging over $\mathcal{N}^m$ will keep global system's time.

Informally, each component t_i of $\vec{t}$ can be considered as an independent clock. The only relationship it has with the other clocks is the restriction on its tact duration: it can "tick" (i.e., increase by 1) no earlier than l_{ij} and no later than u_{ij} "ticks" of the clock t_j, where $\{l_{ij}, u_{ij}\}$ are constants from $(\mathcal{N} \cup \{\infty\})^m$. Less strict, it means that at any instant of time all clocks have to meet the set of inequations:

$$l_{ij} t_j \leq t_i \leq u_{ij}(t_j + 1).$$

An action may fire, if it has been ready to fire in the corresponding untimed graph for at least $\vec{l}$ and no more than $\vec{u}$ time units component-wise.

To represent this model by a Kripke structure we need a more formal description of our timed transition graph.

Definition 4. Let $G_t = (S, \rightarrow, \mathfrak{A}_t, \{\vec{T}_a | a \in \mathfrak{A}_t\}, \{\vec{l}_a | a \in \mathfrak{A}_t\}, \{\vec{u}_a | a \in \mathfrak{A}_t\}, \vec{t})$ be a *timed transition graph*, where

- $\mathfrak{A}_t = \mathfrak{A} \cup \{tick_1, \ldots, tick_m\}$, and $m = |\vec{t}|$; the actions $tick_i$ are reliable for time 'ticking';
- $G = (S, \to, \mathfrak{A})$ is an (untimed) transition graph;
- each $\vec{T}_a$ is the individual timer of an action a;
- $\vec{l}_a$ and $\vec{u}_a$ are the lower and upper time bounds of the action a; for $a = tick_i$ $(\vec{l}_a)_i = 0$ and $(\vec{u}_a)_i = \infty$;
- $\vec{t}$ are the global multiple clocks.

An action $b \in \mathfrak{A}$ is *enabled* in a state s iff $\exists s'.s \xrightarrow{b} s'$. Let $En(b)$ be the set of all states, where the action b is enabled.

Definition 5. A configuration of a system is a pair (s, H), where s is a state, and H is a matrix $m \times n$, where m is the number of *tick*s, and n is the number of different actions. H consists of the values of all individual timers of the system:

$$H = \begin{pmatrix} \vec{T}_{a_1} \\ \vdots \\ \vec{T}_{a_n} \end{pmatrix} = \begin{pmatrix} h_{11} & \ldots & h_{1m} \\ \vdots & & \vdots \\ h_{n1} & \ldots & h_{nm} \end{pmatrix}.$$

Definition 6. An extended configuration of a system is a triple of the form: $(s, H, \vec{t})$, where (s, H) is a configuration, and $\vec{t}$ is a value of the time variable $\vec{t}$.

The semantics of the transition relation $\to$ is extended to the set of extended configurations as follows:

- for $a \neq tick_j$ $(s, \vec{T}_1, \ldots, \vec{T}_a, \ldots, \vec{T}_n, \vec{t}) \xrightarrow{a} (s', \vec{T}_1', \ldots, \vec{0}, \ldots, \vec{T}_n', \vec{t}) \iff$ $s \xrightarrow{a} s'$ and $\vec{l}_a \leq \vec{T}_a \leq \vec{u}_a$ and for all $b \neq a$: $\vec{T}_b' = \begin{cases} \vec{T}_b, & s' \in En(b), \\ \vec{0}, & \text{otherwise}, \end{cases}$ ($\vec{t}$ remains the same);
- for $a = tick_i$ $(s, \vec{T}_1, \ldots, \vec{T}_{tick_i}, \ldots, \vec{T}_n, \vec{t}) \xrightarrow{tick_i} (s, \vec{T}_1', \ldots, \vec{0}, \ldots, \vec{T}_n', \vec{t}')$ $\iff$ $\vec{l}_{tick_i} \leq \vec{T}_{tick_i} \leq \vec{u}_{tick_i}$ and $t_i' = t_i + 1$ and for all $a \neq tick_i$: $(\vec{T}_a)_i < (\vec{u}_a)_i$ and $(\vec{T}_a')_i = \begin{cases} (\vec{T}_a)_i + 1, & s \in En(a), \\ \vec{0}, & \text{otherwise}, \end{cases}$ (s remains the same).

In the 1-dimentional case ($m = 1$) we have a system with real time presented by a single clock, described in [26] and [5]. Actions like ${}_la^u$ in this system occur with time delay, no less than l and no greater than u

tacts. These tacts are counted out by the single time action ${}_0tick^{\infty}$. In the multi-dimentional case ($m \geq 2$) the notion of the tact as a time unit disappears with the conventional notion of time itself. Instead, some independent clocks, whose tact durations meet the following inequations appear: $l_{ij}t_j \leq t_i \leq u_{ij}t_j$ for all $i \neq j$. This model can describe, for example, communication between some processes, which have their own clocks going with different speed and finite precision.

It is easy to understand that actions like ${}_{(...,\infty,...)}a^{(...,\infty,...)}$ will never occur, and hence, can be excluded from the system. For actions like ${}_{(...,l_i,...)}a^{(...,\infty,...)}$ the appropriate components of the individual timers suffice to reach l_i and then to maintain this value as long as the action is enabled, because the action will never actually have to occur before the next $tick_i$. At last, for actions like ${}_{(...,l_i,...)}a^{(...,u_i,...)}$, where $0 \leq l_i \leq u_i < \infty$ the i-th components of the individual timers are limited by u_i. Therefore, we may redefine our system so that every timer has a finite range. So, we can say that D_c is finite, and D_t is countable.

It is obvious that any situation which is possible for the transition graph with multiple clocks can be fully described in terms of extended configurations.

Definition 7. Let $M_t = (D_t, \rightarrow_t, I_t)$ be a *timed model*, if D_t is the set of all extended configurations. We will say that M_t is *induced* by the timed transition graph if the transition relation $\rightarrow_t$ corresponds to $\rightarrow$ in the graph.

We consider also an *untimed* model $M_c = (D_c, I_c)$, where D_c is a set of all configurations, and $\rightarrow_c$ and I_c are projections of $\rightarrow_t$ and I_t respectively on D_c. Formally, it means that two following conditions hold:

1) for any formula F, $d \in I_c(F)$ iff there exists $\vec{t} \in \mathcal{N}^m$, such that $d_{\vec{t}} \in I_t(F)$;

2) for any action b, $d \xrightarrow{b}_c d'$ iff there exist $\vec{t}, \vec{t'} \in \mathcal{N}^m$, that $d_{\vec{t}} \xrightarrow{b}_t d'_{\vec{t'}}$.

In [6] it is stated that assuming time-dependent predicates only $t_i \leq k$, $t_i \geq k$ and $t_i \equiv k(mod\, p)$, we can reduce the problem $M_t \models F$ to $M' \models F$, where the model M' is finite.

To introduce a compositional model it is sufficient to insert in each atomic process p its own copies of $tick_i(p)$ and redefine the process product as the following:

$$p \times_t q := (p \times q) \upharpoonright (\mathfrak{A}_{p\times q} \cup \bigcup_i (tick_i(p) \cdot tick_i(q))),$$

where by $\mathfrak{A}_{p\times q}$ we assume all actions of the process except tick's. In the sequel we will always assume $\times_t$ while using $\times$, and all graphs of processes should be derived from the corresponding models M' mentioned above.

4. The verification of compound processes

In general, any problem $p \models_{Mp} \Phi$ is decidable, but the computational complexity of the direct algorithm depends on the size of a model $n = |M_p|$ as $O(n^k)$, where k depends on a specific formula, and $|M| = |S| + |\rightarrow|$. For a compound process $p = r \times q$ the number of states is the product of the number of states of r and q: $n_p = |S_r \times S_q| = n_r n_q$, what considerably restricts the practical application of the direct model checking procedure. So our main task in this section is to develop the procedure of formula evaluation in which it is sufficient to construct the validity sets of a formula only for atomic processes.

There are two approaches to the solution of a problem $p \times q \models_M \Phi$. The principal idea of them is proposed in [28].

I. Given a formula Φ and processes p and q, construct the new formulas Φ_p and Φ_q, such that $(p \models_{M_p} \Phi_p$ and $q \models_{M_q} \Phi_q) \iff p \times q \models_M \Phi$ and then solve two problems $p \models_{M_p} \Phi_p$ and $q \models_{M_q} \Phi_q$.

II. Given a formula Φ and a process p, construct the new formula Φ/p, such that $q \models_{M_q} \Phi/p \iff p \times q \models_M \Phi$ for any q, and then solve the problem $q \models_{M_q} \Phi/p$.

4.1. CMC proof system

Let us discuss approach I, where the original problem for a compound process is split in two problems for the subprocesses, but the size of the projection of a formula does not exceed the size of the original formula. We begin with the restriction on the semantics. Let fix some models $M = (S, I)$, $M_p = (S_p, I_p)$ and $M_q = (S_q, I_q)$ induced by the graphs of processes $p \times q$, p and q respectively. Assume that for all propositional constants and free variables their interpretation $I(P)$ is a Cartesian product $I_p(P) \times I_q(P)$ of their interpretations in M_p and M_q.

Let in some model $M = (S, I)$ induced by the graph of a process $p \times q$, the interpretation of all propositional constants and free variables $I(P)$ be a Cartesian product $I_p(P) \times I_q(P)$, where $I_p(P) \subseteq S_p$ and $I_q(P) \subseteq S_q$ are interpretations in models M_p and M_q of subprocesses p and q respectively.

We will use the following notations:

- $\alpha(p) = \{a \in \mathfrak{A}_p \mid \exists p' : p \xrightarrow{a} p'\}$. Note that $* \in \alpha(p)$ for any p.
- For any $b \in \mathfrak{A}_p \quad b(p) = \{q \mid p \xrightarrow{b} q\}$.

First we introduce a new modal operator $\langle\!\langle a \rangle\!\rangle \Phi$ in our logic, where Φ is a formula, and a is an action. The semantics of this operator is $\langle\!\langle a \rangle\!\rangle \Phi \equiv \langle a \rangle \mathrm{True} \wedge [a] \Phi$.

Let us assume that there are no fixed-point subformulas (beginning with μ and ν) in F, no disjunctions and negations, and no []-modalities (we leave only $\langle\rangle$ and $\langle\!]$ modalities).

Then the algorithm (1) of constructing the projections F_p and F_q of the formula F on processes p and q for a problem $p \times q \models_M F$ will be the following:

case F **of**
P: (* prop. const. *) $F_p := P$; $F_q := P$;
| $F' \wedge F''$: $F_p := F'_p \wedge F''_p$; $F_q := F'_q \wedge F''_q$;
(* here all F_p and F_q are computed recursively *)
| $\langle\!]ab[\!\rangle F'$: $F_p := \langle\!]a[\!\rangle F'_p)$
$F_q := \langle\!]b[\!\rangle F'_q)$
end.

The complete proof system of the fragment for all processes with the only restriction on relabelling is shown in Figure 2. The restriction is that $\Xi^{-1}(a)$ should be unique for all $a \in \mathfrak{A}_p$.

$$\frac{p_A \models F}{p_A \vdash F}(p_A \text{ is an atomic process}) \qquad \frac{p \vdash F[\text{False}/\langle\!]a[\!\rangle F', \text{False}/\langle a\rangle F']}{p \upharpoonright \Lambda \vdash F} a \notin \Lambda \text{ and } \Phi \neq \emptyset$$

$$\frac{p \vdash F[\Xi^{-1}(a)/a]}{p\{\Xi\} \vdash F}(\Xi^{-1}(a) \text{ is unique}) \qquad \frac{p \vdash F_p \; q \vdash F_q}{p \times q \vdash F}$$

Figure 2. The proof system for FPA and the fragment of the V-free μ-calculus with the modality operator $\langle\!]a[\!\rangle F$)

Statement 1. *For the formulas F_p and F_q, constructed by the algorithm* (1) *described above the following holds:*

$$(p \models_{M_p} F_p \text{ and } q \models_{M_q} F_q) \iff p \times q \models_M F.$$

The proof can be made by the induction on the structure of a formula.

Note, that the algorithm (1) does not change the structure of a formula, it only replaces symbols of actions in $(a \to \Phi)$ formulas.

Let us define the operation of projection on a subprocess for the constructors of fixed points as $(\mu X.\Phi)_p = \mu X.\Phi_p$ and $(\nu X.\Phi)_p = \nu X.\Phi_p$. In [7] it is proved that this definition is sound. Thus, we have the following

Theorem 1. *Let $M_p = (S_p, I_p)$, $M_q = (S_q, I_q)$ and $M = (S, I)$ be the models induced by the graphs of processes p, q and $p \times q$ respectively, and let the interpretation $I(P)$ of all atomic formulas in M be the Cartesian product of the interpretations $I_p(P) \times I_q(P)$ of the same atomic formula in the models*

M_p and M_q. Then for any formula F of μ-calculus without disjunctions and negations and a modal operator $(a \to \Phi)$ with $\Phi \neq \emptyset$ there exist such formulas F_p and F_q from the same fragment, that

$$p \times q \models_M F \iff (p \models_{M_p} F_p \text{ and } q \models_{M_q} F_q).$$

Moreover, the size of each formula F_p and F_q does not exceed the size of the original formula F.

4.2. Tableau-based model checking

The only serious restriction of the method described above is the absence of disjunctions in formulas. Although for some formulas with disjunctions we can apply this method (mainly by transforming those formulas into the sets of formulas without disjunctions), in general it is impossible.

The most efficiently, in the author's opinion, this method can be used in implementation of tableau-based model checking, developed by R. Cleaveland in [13]. It makes the algorithm (1) applicable to model checking for the full μ-calculus, and makes the result complete.

Definition 8. A *sequent* is an expression $\mathcal{H} \vdash_M p \in F$, where p is a (compound) process, F is a μ-calculus formula, M is a Kripke structure, and $\mathcal{H}$ is the set of hypotheses, i.e., pairs of the form $(q : \sigma X.\Phi)$, $\sigma \in \{\mu, \nu\}$, where q is a process, $\sigma X.\Phi$ is a fixed point formula.

Definition 9. A sequent $\mathcal{H} \vdash_M p \in F$ is called a *leaf* if one of the following holds:

1) F is a propositional constant;
2) F is a formula from the fragment described in Theorem 1;
3) $(p : F) \in \mathcal{H}$;
4) $F \equiv \langle a \rangle \Phi$ or $F \equiv [a] \Phi$ or $F \equiv \langle\!\langle a \rangle\!\rangle \Phi$, and $a(p) =_{df} \{q \mid p \xrightarrow{a} q\} = \emptyset$.

A leaf $\mathcal{H} \vdash_M p \in F$ is successful if either:

- $p \models_M F$ is valid (can be proved using the algorithm (1)); or
- $F \equiv \nu X.\Phi$ and $(p : F) \in \mathcal{H}$; or
- $F \equiv (a \to \Psi)$ and $\{q \mid p \xrightarrow{a}_M q\} = \Psi = \emptyset$.

A leaf is not successful otherwise.

Inference rules:

$$\frac{\mathcal{H} \vdash_M p \in F'}{\mathcal{H} \vdash_M p \in F' \vee F''}, \qquad \frac{\mathcal{H} \vdash_M p \in F' \quad \mathcal{H} \vdash_M p \in F''}{\mathcal{H} \vdash_M p \in F' \wedge F''},$$

$$\frac{\mathcal{H} \vdash_M q_1 \in F \ldots \mathcal{H} \vdash_M q_n \in F}{\mathcal{H} \vdash_M p \in [a]\Psi}(\{q_1,\ldots,q_n\}=a(p))\ ([.]),$$

$$\frac{\mathcal{H} \vdash_M q_1 \in F \ldots \mathcal{H} \vdash_M q_n \in F}{\mathcal{H} \vdash_M p \in \langle\!|a|\!\rangle\Psi}(\{q_1,\ldots,q_n\}=a(p))\ (\langle\!|.|\!\rangle),$$

$$\frac{\mathcal{H} \vdash_M q \in F}{\mathcal{H} \vdash_M p \in \langle a\rangle\Psi}(q \in a(p))\ (\langle.\rangle),$$

$$\frac{\mathcal{H}' \cup \{(p : \sigma X.F)\} \vdash_M p \in F[\sigma X.F/X]}{\mathcal{H} \vdash_M p \in \sigma X.F},$$

where $\sigma \in \{\mu,\nu\}$,$\mathcal{H}' = \mathcal{H} \setminus \{(p:\Phi) \mid \sigma X.F \preceq \Phi\}$.

Here $F \preceq \Phi$ means that F is a subformula of Φ. In the third rule it is allowed that $q_i = q_j$ or $\varphi_i = \varphi_j$ for some $i \neq j$, but it is important that *all* next states and *all* formulas from Ψ appear among the assumptions. A sequent is valid iff there exists a proof in this axiomatic system, ending with successful leaves. Soundness, completeness and decidability are proved in [13]. The verification problem $p \models_M F$ is equivalent to the validity problem of the sequent $\emptyset \vdash_M p \in F$.

In this work the fact is important that the proofs of the most formulas significantly diminish due to the definition of leaves (Definition $9_{1)}$). The original proof system in [13] consists of similar rules (except that for the modal operator: Cleaveland uses ordinary [.]- and $\langle.\rangle$- modalities), but with different definition of leaves. In his system one have to unfold all fixed points and apply other rules until one gets either a propositional constant or the empty set of assumptions or a formula from $\mathcal{H}$. In our system one may stop earlier and continue model checking for subprocesses.

Thus, we completed the construction of the first compositional model checking algorithm for FPA and the modal μ-calculus. In the next section we consider another quite different approach to the same problem, and for the extended version of the μ-calculus.

4.3. Another approach to the verification of compound processes

Let us consider another approach, in which the original problem $p \times q \models F$ is reduced to an equivalent one $q \models F/p$. All we need is to construct the formula F/p using the original formula F and the process p. We offer an algorithm for an arbitrary process p and a formula F from the full modal μ-calculus. It is an improvement of the previous result [7] for the fragment of μ-calculus with alternation-free fixed points.

Assume that there are no fixed-point subformulas (beginning with μ and ν) in F, and no negations. Then for the problem $p \times q \models_M F$ an algorithm (2) for constructing the projection F/p of F on the process p is the following.

case F **of**
P: (* prop. const. *) $F/p := P/p$;
| $F' \wedge F''$: $F/p := F'/p \wedge F''/p$;
(* here F'/p and F''/p are computed recursively *)
| $F' \vee F''$: $F/p := F'/p \vee F''/p$;
| $\langle ab \rangle F'$: **if** $a \notin \alpha(p)$ **then** $F/p :=$ False
else $F/p := \langle b \rangle \bigvee_{p' \in a(p)} F'/p'$ **end**;
| $[ab] F'$: **if** $a \notin \alpha(p)$ **then** $F/p :=$ True
else $F/p := [b] \bigwedge_{p' \in a(p)} F'/p'$ **end**;
end.

The semantics of P/p is defined as $I_q(P/p) = \{s \in S_q | (s_p, s) \in I(P)\}$, where s_p is the initial state of the process p, and I and I_q are interpretations in the models induced by the graphs of processes $p \times q$ and q respectively.

The axiomatic system for all FPA is shown in Figure 3.

$$\frac{p_A \models F}{p_A \vdash F}\,(p_A \text{ is an atomic process}) \qquad \frac{p \vdash F[\text{False}/\langle a \rangle \Phi, \text{True}/[a]\Phi]}{p \upharpoonright \Lambda \vdash F}\, a \notin \Lambda$$

$$\frac{p \vdash F\left[\bigvee_{b \in \Xi^{-1}(a)} \langle b \rangle \Phi / \langle a \rangle \Phi, \bigwedge_{b \in \Xi^{-1}(a)} [b]\Phi / [a]\Phi\right]}{p\{\Xi\} \vdash F} \qquad \frac{q \vdash F/p}{p \times q \vdash F}$$

Figure 3. The proof system for FPA and the whole μ-calculus

Statement 2. *For the formula F/p, constructed by the algorithm* (2) *described above the following holds:*

$$q \models_{Mq} F/p \iff p \times q \models_M F.$$

First, we extend our logic by vector expressions $(A_1, \ldots, A_n)$ and simultaneous fixed points

$$\sigma \vec{\Phi}^n . (\vec{X}^m = \vec{P}^m),$$

$\sigma \in \{\mu, \nu\}$ in the spirit of [1]. The last expression corresponds to the vector of formulas $(\Phi_1, \ldots, \Phi_n)$, where all free occurrences of $\vec{X}$ are interpreted as the least (resp. the greatest) solutions of the equation system $\vec{X}^m \equiv \vec{P}^m$. The vector $\vec{P}^m$ in its turn can be either a simple vector of formulas, or a simultaneous fixed point. We call this logic *an extended μ-calculus.* From

the Bekič theorem it follows that any simultaneous fixed point can be equally represented by ordinary fixed point, though in general with an exponential growth in size (see, e.g. [1]).

Assume we are given a problem $p \times q \models F$, where F is a formula of the extended μ-calculus. We will construct the projection of F on p in two steps. First we build *a decision graph* $Gr(F,p)$, and then with the graph we construct a formula F/p of the extended μ-calculus.

The graph $Gr(F,p) = (V,E)$ is built as the following. The set of *nodes* $V = \{(\Phi, s) \mid \Phi$ is not a vector formula, $\Phi \preceq F$, $s \in S_p\}$ is the set of pairs of all subformulas of F and all states of the transition graph of the process p. The set of *edges* E is the least set satisfying the followimg conditions:

- from a node (P,s) there are no outcóming edges, where P is a predicate;
- from (X_j,s) there is one edge to (P_j,s), if X_j is bounded by $\sigma\vec{\Phi}^n$ $(\vec{X}^m = \vec{P}^m)$; $\sigma \in \{\mu,\nu\}$;
- from nodes $(\Phi_1 \vee \Phi_2, s)$ and $(\Phi_1 \wedge \Phi_2, s)$ there are two edges to (Φ_1, s) and (Φ_2, s);
- from nodes $(\langle ab\rangle \Phi, s)$ and $([ab]\Phi, s)$ there are edges to all nodes (Φ, s'), such that $s \xrightarrow{a} s'$.

Assume given a Kripke structure $M_{p\times q} = (S_{p\times q}, \rightarrow_{p\times q}, I_{p\times q})$. The function $\phi : V \mapsto 2^{S_q}$ is called a decision graph's *marking*, mapping each node (Φ, s) to the set of states of the process q by the rules:

(a) $\phi(P,s) = I_q^s(P) \rightleftharpoons \{r \in S_q \mid (s,r) \in I_{p\times q}(P)\}$.

Let the node v be $\vee$-node, if $v = (\Phi_1 \vee \Phi_2, s)$ or $v = (\langle a\rangle \Phi, s)$, and $\wedge$-node, if $v = (\Phi_1 \wedge \Phi_2, s)$ or $v = ([a]\Phi, s)$. Then

(b) for $\vee$-node $\phi(v) = \bigcup_i \phi(v_i)$,

(c) for $\wedge$-node $\phi(v) = S_q \cap \bigcap_i \phi(v_i)$,

for all v_i, such that $(v, v_i) \in E$. For the rest of nodes with only one outcoming edge

(d) $\phi(v) = \phi(v')$, $(v,v') \in E$.

The value of $\phi(X,s)$ is determined as the least (the greatest) possible set with respect to (d), if X is bounded by the constructor of the least (the greatets) fixed point. Minima and maxima of such sets have priorities corresponding to the order of the nested fixed points.

The original problem $p \times q \models F$ can be shown to be equivalent to $s_q \in \phi(F, s_p)$. We will not try to find an algorithm of constructing ϕ directly. Instead, we will build a formula F/p from the extended μ-calculus by the graph $Gr(F,p)$, such that $I_q(F/p) = \phi(F, s_p)$, where $I_q \rightleftharpoons I_q^{s_p}$.

1. For every node (P, s) take a new propositional constant P/s: $I_q(P/s) = I_q^s(P)$.
2. For every node (X, s) take a new variable X/s.
3. Nodes $(\Phi_1 \vee \Phi_2, s)$ and $(\Phi_1 \wedge \Phi_2, s)$ map to the formulas $\Phi_1/s \vee \Phi_2/s$ and $\Phi_1/s \wedge \Phi_2/s$ respectively, where Φ_i/s corresponds to a node (Φ_i, s).
4. A node $(\langle a \rangle \Phi, s)$ maps to the formula $\langle c \rangle (\mathrm{False} \vee \bigvee_{s'} \Phi/s')$, where Φ/s' corresponds to (Φ, s'), to which edges are coming from $(\langle a \rangle \Phi, s)$, and $a = bc$. Similarly $([a]\Phi, s)$ maps to the formula $[c](\mathrm{True} \wedge \bigwedge_{s'} \Phi/s')$.
5. Beginning with the deepest nested fixed point $\sigma\vec{\Phi}.(\vec{X} = \vec{P})$ in the original formula, build an expression $\sigma\vec{\vec{\Phi}}.(\vec{\vec{X}} = \vec{\vec{P}})$, where the vector $\vec{\vec{A}} = (A_1/s_1, \ldots, A_m/s_n)$, and nodes (A_i, s_j) appear in the decision graph. Indexes in $\vec{\vec{X}}$ and $\vec{\vec{P}}$ have the same order, so that $\vec{\vec{X}} = \vec{\vec{P}}$ corresponds to the right system of equations. It is possible, since all nodes (P_i, s_j) are produced in the graph only by nodes (X_i, s_j), and hence, there are the same number of them.

Finally, as a result we take the formula F/s_p, corresponding to the original F in the state s_p, and denote it by $F/p \rightleftharpoons F/s_p$. It is easy to show that it is the formula for which $p \times q \models F \iff q \models F/p$. Obviously, the size of the formula is $|F/p| \leq (|G_p| \cdot |F|)$.

4.4. One more efficient MC algorithm

Approach II can be easily used in the description of another efficient MC algorithm, which in some sence can be considered as the refinement of the algorithm of Andersen [2].

Assume given the problem $p \models F$. Applying the algorithm (2) to all products in the process p consequently, we will finally obtain an equivalent problem $q \models F'$, where q is an atomic process. Composing q with $\mathbf{1} = (\{\bullet\}, \to_1, \{*\})$ – the process with only one state $\bullet$ and action $*$, we will have an equivalent problem $(\mathbf{1} \times q)\{* \cdot \alpha \mapsto \alpha\} \models F'$. Now we again apply the algorithm (2) to the relabelling and q. As a result, we will have the problem $\mathbf{1} \models \Phi$ equivalent to the original one, and $|\Phi| \leq |G_p| \cdot |F|$. All $\langle\rangle$- and $[]$-subformulas are of the form $\langle * \rangle G_1$ and $[*]G_2$, and hence, are equivalent to G_1 and G_2 respectively. Therefore, we can consider Φ as a $\langle\rangle$- and $[]$-free formula.

In [2] an algorithm of evaluating of such formula in the model M_1 is presented with the complexity $O(|\Phi|) = O(|G_p| \cdot |F|)$ for $\mathrm{ad}(F) \leq 1$, and hence, for $\mathrm{ad}(F) > 1$ the complexity is $O(|G_p| \cdot |F|(|S_p| \cdot |F|)^{\mathrm{ad}(F)-1})$ by Theorem 5.1 from [2].

Thus, there exists an efficient model checking algorithm for μ-calculus like in [2], but for a model given as a system of parallel processes and for

the variant of the logic with more concise fixed points. In general, the size of formulas in μ-calculus with the constructor $(\sigma\vec{X}.\vec{\Phi})_i$ grows exponentially on the size of equivalent formulas with the constructor $\sigma\vec{\Phi}.(\vec{X} = \vec{P})$. Note also, that our approach does not need to translate fixed points to *simple* fixed points beforehand. It is done directly while checking the formula Φ and is a part of the algorithm [2].

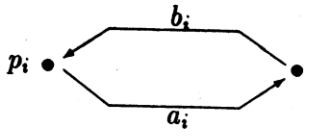
4.5. Efficient PCMC is the unreachable dream of expressive logics

In this section we will clarify some rather serious restrictions of pure compositional MC methods (PCMC) which do not use any additional information about a model but hierarchy. The idea is the following. Assume we are given an arbitrary atomic process p with the number of states N. Then we can build a compositional process q with the sum of the number of states of all atomic subprocesses $2\lceil\log_2 N\rceil$, which is strongly bisimilar to p. Further, if we have a MC algorithm for some fragment of the μ-calculus with the complexity $O((N \cdot |F|)^k)$ and a method of formula decomposition with no more than polynomial growth and the complexity no greater than $O(|F|^n \cdot \log_2 N)$, where n is a constant, then we immediately have a MC algorithm for the logic with the complexity $O(k \log_2 N \cdot |F|^{\max\{n,k\}})$. Since the problem of MC for μ-calculus is known to be at least PTIME-complete and there are algorithms with the complexity $O((N \cdot |F|)^{\mathrm{ad}(F)} | \to |)$, then there exists no efficient decomposition procedure for it. All the same is concerned with CTL. The MC problem for it is PTIME-complete, and the best existing algorithm is linear. Therefore, we can not decrease the complexity to $\log_2 N$. Now we formulate this as a theorem.

Theorem 2. *Given a fragment of μ-calculus with at least linear MC procedure on the size of a model. Then, if there exists an algorithm of constructing formulas F' and F'' from p, q and F, such that $p \times q \models F \iff p \models F'$ (and/or $q \models F''$), then its complexity is at least exponential on the size of p (and/or q).*

Proof. It suffices to build a process with the sum of the number of states of all its atomic subprocesses no greater than $2\lceil\log_2 N\rceil$ from the original LTS with the number of states N.

Let the initial LTS be defined as an atomic process p. We introduce $n = \lceil\log_2 N\rceil$ new atomic processes $p_1, \ldots, p_n$ with two states and two transitions a_i and b_i (and of course, the idling action$*$):



Enumerate all states of the process p by bit vectors of the length n, such that $s_p = (0,\ldots,0)$. The desired process will be $p' = (p_1 \times \ldots \times p_n)\,\restriction \Lambda\{\Xi\}$, where $c_1 \cdot \ldots \cdot c_n \in \Lambda \iff$ in p there exists an action α from the state $(i_1,\ldots,i_n)$ to $(j_1,\ldots,j_n)$ and

$$c_k = \begin{cases} a_k, & j_k > i_k, \\ b_k, & j_k < i_k, \\ *\ , & j_k = i_k, \end{cases}$$

and $\Xi(c_1 \cdot \ldots \cdot c_n) = \alpha$ for the α and c_k from the definition of Λ. Obviously, p' will be strongly bisimilar to the process p. Note that the process p' has exactly $2n$ states of atomic subprocesses by the construction. □

The result has also a positive corollary. For instance, for the μ-calculus without negation and disjunction, with proper modalities and for a proper class of models we have the algorithm (1) of decomposition with the complexity $O(|F|)$, not depending on the size of a model. Therefore, for the fragment there exists a MC algorithm with the complexity $O(|F|^{\mathrm{ad}(F)} \cdot \mathrm{ad}(F) \cdot \log_2 |S|)$. The restriction on models can be, for instance, the requirement that all different transitions in the initial process be labeled by different action symbols.

5. Conclusion

We offered a universal Finite Process Algebra allowing to describe finite parallel systems as conveniently as in CCS and SCCS. Moreover, this algebra can be easily extended by multiple clocks, what significantly increases the expressive power of the language, and one does not need to rebuild the algorithms.

The semantics of a compound action firing is a semantics of 'dynamic boundaries', when whether an action ab may occur is determined by the simultaneous possibility of firing of a and b in their subprocesses. This semantics properly reflects the real situation, when a user defines different processes with their internal clocks and delays, and does not care about the delays of compound actions.

Another important result is the theorem on the minimal complexity for the worst case of a decomposition algorithm for rather expressive logics like CTL. As a corollary, the MC algorithm with the logarithmic complexity on the number of states was found for a restricted fragment of the modal μ-calculus.

Acknowledgements. I would like to thank Dr. Shilov N.V. and Dr. Nepomnyaschy V.A. from the Institute of Informatics Systems for many fruitful discussions and helpful advices about the results presented in the paper. I

am also very grateful to Dilian Gurov from the University of Victoria for his discussions on the earlier versions of the paper, and to Dr. Henrik R. Andersen from Danish Technical University for careful reading and looking for subtle errors.

References

[1] H.R. Andersen, *Verification of Temporal Properties of Concurrent Systems*, Ph.D. thesis, Dept. of Comp. Sci., Aàrhus University, Denmark, June 1993.

[2] H.R. Andersen, *Model checking and boolean graphs*, Theoretical Computer Science 126(1), 1994, 3–30.

[3] H.R. Andersen, C. Stirling, G. Winskel, *A Compositional Proof System for the Modal μ-Calculus*, To appear in: Proceedings of LICS94, IEEE Computer Society Press.

[4] Adnan Aziz, Thomas R. Shiple, Vigyan Singhal, Alberto L. Sangiovanni-Vincentelli, *Formula-Dependent Equivalence for Compositional CTL Model Checking*, Proceedings of CAV'94, July 1994, in LNCS, Vol. 818, 324–337.

[5] S.A. Berezin, N.V. Shilov, P.V. Shneider, *An effective model checking for Mu-calculus: from finite systems towards systems with real time*, Bulletin of the Novosibirsk Computing Center, Series: Computer Science, issue 1, 1993.

[6] S.A. Berezin, N.V. Shilov, *An Approach to Effective Model-Checking of Real-Time Finite-State Machines in Mu-Calculus*, Proceedings of LFCS'94, in LNCS, St. Petersburg, Russia, July, 1994. Vol. 813, 47–55,

[7] S.A. Berezine, *Model checking in μ-calculus for distributed systems*, To appear in: Specification, Verification, and Net Models of Concurrent Systems, Institute of Informatics Systems, Novosibirsk, Russia, 1994.

[8] Orna Bernholtz, Moshe Y. Vardi and Pierre Wolper, *An Automata-Theoretic Approach to Branching-Time Model Checking*, Proceedings of CAV'94, in LNCS, July 1994, Vol. 818, 142–155.

[9] B. Boiglot and P. Wolper, *Symbolic Verification with Periodic Sets*, Proceedings of CAV'94, in LNCS, July 1994, Vol. 818, 55–67.

[10] V.S. Brace, R.L. Rudell and R.E. Bryant, *Efficient Implementation of a BDD Package*, Proc. of 27th ACM/IEEE Design Automation Conf., June, 1990.

[11] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill and J. Hwag, *Simbolic model checking: 10^{20} states and beyond*, In: Proc. 5th Ann. Symp. on Logic in Computer Science, IEEE Comp. Sci. Press, June, 1990.

[12] E. Clarke, T. Filkorn, S. Jha, *Exploiting Symmetry in Temporal Logic Model Checking*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 450–462.

[13] R. Cleaveland, *Tableau-based model checking in the Propositional Mu-Calculus*, Acta Informatica, Vol. 27, No. 8, 1990, 725–748.

[14] R. Cleaveland, M. Klein, B. Steffen, *Faster Model Checking for the Modal Mu-Calculus*, Proc. of CAV-92, Montreal, Canada, in LNCS, Vol. 663, 410–422.

[15] D. Dams, O. Grumberg, R. Gerth, *Generation of Reduced Models for Checking Fragments of CTL*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 479–490.

[16] D. Dams, R. Gerth, G. Döhmen, R. Herrmann, P. Kelb and H. Paragmann, *Model Checking Using Adaptive State and Data Abstraction*, Proceedings of CAV'94, July 1994, in LNCS, Vol. 818, 455–467.

[17] E.A. Emerson, C.S. Julta, A.P. Sistla, *On model-checking for fragments of Mu-calculus*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 385–396.

[18] E. Emerson and A. Sistla, *Symmetry and Model Checking*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 463–478.

[19] Scott Hazelhurst and Carl-Johan H. Seger, *Composing Symbolic Trajectory Evaluation Results*, Proceedings of CAV'94, July 1994, in LNCS, Vol. 818, 273–285.

[20] Scott Hazelhurst and Carl-Johan H. Seger, *A Simple Theorem Prover Based on Symbolic Trajectory Evaluation and BDDs*, (to appear).

[21] H. Hungar, *Combining model checking and theorem proving to verify parallel processes*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 154–165.

[22] H. Hüttel, *SnS can be modally characterized*, Theoretical Computer Science, Vol. 74, No. 2, August, 1990, 239–248.

[23] D. Kozen, *Results on the propositional μ-calculus*, Theoretical Computer Science, Vol. 27, No. 3, December, 1983, 333–354.

[24] David E. Long, Anca Browne, Edmund M. Clarke, Somesh Jha, Wilfredo R. Marrero, *An Improved Algorithm for the Evaluation of Fixpoint Expressions*, Proceedings of CAV'94, July 1994, in LNCS, Vol. 818, 338–350.

[25] R. Milner, *Calculi for synchrony and asynchrony*, Theoretical Computer Science, Vol. 25, No. 3, 1983, 267–310.

[26] J.S. Ostroff, *Automated verification of timed transition models*, Lect. Notes in Comput. Sci., Vol. 407, 247–256.

[27] Oleg V. Sokolsky, Scott A. Smolka, *Incremental model checking in the modal Mu-Calculus*, Proceedings of CAV'94, July 1994, in LNCS, Vol. 818, 351–363.

[28] C. Stirling, *Modal logics for communicating systems*, Theoretical Computer Science, 49, July, 1987, 311–348.

[29] Igor Walukiewicz, *Completeness of Kozen's axiomatization of the propositional μ-calculus*, Manuscript, November, 1994.

[30] T. Yoneda and A. Shibayama, B.-H. Schlingloff and E. Clarke, *Efficient verification of parallel real-time systems*, Proceedings of CAV'93, June/July 1993, in LNCS, Vol. 697, 321–332.