Verifying Real-Time Properties of Multi-agent Systems via SMT-Based Bounded Model Checking

Agnieszka M. Zbrzezny(✉) and Andrzej Zbrzezny

IMCS, Jan Długosz University, Al. Armii Krajowej 13/15, 42-200 Częstochowa, Poland
{agnieszka.zbrzezny,a.zbrzezny}@ajd.czest.pl

Abstract. We present a satisfiability modulo theories based bounded model checking (SMT-based BMC) method for timed interpreted systems ($\mathbb{TIS}$) and for properties expressible in the existential fragment of a Real-Time Computation Tree Logic with epistemic components (RTECTLK). We implemented the standard BMC algorithm and evaluated it for two multi-agent systems: a timed train controller system and a timed generic pipeline paradigm. We used the Z3 solver.

1 Introduction

The formalism of *interpreted systems* (IS) was introduced in [8] to model multi-agent systems (MAS) [19], which are intended for reasoning about the agents' epistemic and temporal properties. The formalism of timed interpreted systems ($\mathbb{TIS}$) [22] extends IS to make the reasoning possible about not only temporal and epistemic properties, but also about real-time aspects of MASs. The previous ten years in the area of MASs have seen significant research in verification procedures, which automatically evaluate whether a MAS reaches its intended specifications.

Model checking [4] is an automatic verification technique for concurrent systems such as: digital systems, distributed systems, real time systems, multi-agent systems, communication protocols, cryptographic protocols, concurrent programs, and many others. To be able to check automatically whether the system satisfies a given property, one must first create a model of the system, and then describe in a formal language both the created model and the property.

One of the main technique here is *symbolic model checking* [4]. Unfortunately, because of the agents' intricate nature, the practical applicability of model checking is firmly limited by the "state-space explosion problem" (i.e., an exponential growth of the system state space with the number of agents). To reduce this issue, various techniques, including the SAT- and BDD-based bounded model checking (binary decision diagrams based BMC) [10,15,18], have been advanced.

Partly supported by National Science Centre under the grant No. 2014/15/N/ST6/05079.

M. Baldoni et al. (Eds.): PRIMA 2016, LNAI 9862, pp. 149–167, 2016.
DOI: 10.1007/978-3-319-44832-9_9

These have been effective in permitting users to handle bigger MASs, however it is still hard to check MASs with numerous agents. The point of this paper is to help beat this inadequacy by employing SMT-solvers (i.e., *satisfiability modulo theories* tools for deciding the satisfiability of formulae in a number of theories) [2].

Bounded model checking for multi-agent systems is a symbolic model checking method designed for finding counterexamples, and whose main idea is to consider a model curtailed to a specific depth to search for an execution (or a set of executions) of a system under consideration of some length k, which constitutes a counterexample for a tested property. It uses a reduction of the problem of truth of a temporal formula [6] (an epistemic formula [8], doxastic formula [12], and deontic formula [14]) in a model of MAS to the problem of satisfiability of formulae. The reduction is achieved by a translation of the transition relation and a translation of a given property to a quantifier-free first-order formula. It should be emphasised that for a given temporal logic, bounded model checking is mainly used to disprove safety properties and to prove liveness properties.

A version of the SAT-based BMC method for specifications expressed in RTECTLK, and MASs modelled by interleaved interpreted systems (the interpreted system with the asynchronous semantics (interleaving semantics)), in which agents have time-limits or other explicit timing constraints to accomplish intended goals, has been published in [21].

The original contributions of the paper are as follows. First, we propose a SMT-based BMC technique for $\mathbb{TIS}$ and for RTECTLK. Second, we report on the implementation of the proposed BMC method as a new module of a verification system, and evaluate it experimentally by means of a modified *generic pipeline paradigm* [17] and a modified *train controller system*.

We do not compare our results with other model checkers for MASs, e.g. MCMAS [13] or MCK [9], simply because they do not support the RTECTLK language and the timed interpreted systems.

Scheme of the Paper. The rest of the paper is organised as follows. In the next section we briefly present the theory of the timed interpreted systems and the RTECTLK language. In Sect. 3 we present our SMT-based BMC method and an example of translation to SMT. In Sect. 4 we experimentally evaluate the performance of our SMT-based BMC encoding. We conclude with a brief discussion in Sect. 5.

2 Preliminaries

Timed Interpreted Systems ($\mathbb{TIS}$) were proposed in [22] to extend interpreted systems (ISs) in order to make possible reasoning about real-time aspects of MASs. In the formalism of interpreted systems, each agent is characterised by a set of local states and by a set of local actions that are performed following a local protocol. Given a set of initial states, the system evolves in compliance with an evolution function that determines how the local state of an agent changes as a function of its local state and of the other agents actions. The evolution of all the agents local states describes a set of runs and a set of reachable states. These can be used to interpret formulae involving temporal operators, epistemic operators to reason about what agents know.

2.1 Timed Interpreted Systems

Let $\mathbb{N}$ be a set of natural numbers, and $\mathbb{N}_+ = \mathbb{N}\setminus\{0\}$. We assume a finite set $\mathbb{X}$ of variables, called *clocks*. Each clock is a variable ranging over a set of non-negative natural numbers. For $x \in \mathbb{X}$, $\bowtie \in \{<, \leq, =, >, \geq\}$, $c \in \mathbb{N}$ we define a set of clock constraints over $\mathbb{X}$, denoted by $\mathcal{C}(\mathbb{X})$, The constraints are conjunctions of comparisons of a clock with a time constant c from the set of natural numbers $\mathbb{N}$, generated by the following grammar:

$$\mathfrak{cc} := \mathbf{true} \mid x \bowtie c \mid \mathfrak{cc} \wedge \mathfrak{cc}.$$

A clock valuation v of $\mathbb{X}$ is a total function from $\mathbb{X}$ into the set of natural numbers. The set of all the clock valuations is denoted by $\mathbb{N}^{\mathbb{X}}$. For $\mathbb{X}' \subseteq \mathbb{X}$, the valuation which assigns the value 0 to all clocks is defined as: $\forall_{x\in\mathbb{X}'} v'(x) = 0$ and $\forall_{x\in\mathbb{X}\setminus\mathbb{X}'} v'(x) = v(x)$. For $v \in \mathbb{N}^{\mathbb{X}}$, $succ(v)$ is the clock valuation of $\mathbb{X}$ that assigns the value $v(x) + 1$ to each clock x. A clock valuation v satisfies a clock constraint $\mathfrak{cc}$, written as $v \models \mathfrak{cc}$, iff $\mathfrak{cc}$ evaluates to true using the clock values given by v.

Let $\mathcal{A} = \{1, \ldots, n\}$ denote a non-empty and finite set of agents, and Ev be a special agent that is used to model the environment in which the agents operate, and $\mathcal{AP} = \bigcup_{i\in\mathcal{A}\cup\{Ev\}} \mathcal{AP}_i$ be a set of atomic formulae, such that $\mathcal{AP}_{i_1} \bigcap \mathcal{AP}_{i_2} = \emptyset$ for all $i_1, i_2 \in \mathcal{A} \cup \{Ev\}$.

A *timed interpreted system* is a tuple

$$\mathbb{TIS} = (\{L_i, Act_i, \mathbb{X}_i, P_i, \mathcal{V}_i, \mathcal{I}_i, \iota_i\}_{i\in\mathcal{A}\cup\{Ev\}}, \{t_i\}_{i\in\mathcal{A}}, \{t_{Ev}\}),$$

where:

- L_i is a non-empty set of *locations* of the agent i,
- $\iota_i \subseteq L_i$ is a non-empty set of initial locations,
- Act_i is a non-empty set of *possible actions* of the agent i, $Act = Act_1 \times \ldots \times Act_n \times Act_{Ev}$ is the set of *joint actions*,
- $\mathbb{X}_i$ is a non-empty set of *clocks*,
- $P_i : L_i \to 2^{Act_i}$ is a *protocol function*,
- $t_i : L_i \times L_{Ev} \times \mathcal{C}(\mathbb{X}_i) \times 2^{\mathbb{X}_i} \times Act \to L_i$ is a (partial) *evolution function* for agents,
- $t_{Ev} : L_{Ev} \times \mathcal{C}(\mathbb{X}_{Ev}) \times 2^{\mathbb{X}_{Ev}} \times Act \to L_{Ev}$ is a (partial) *evolution function* for environment,
- $\mathcal{V}_i : L_i \to 2^{\mathcal{AP}_i}$ is a *valuation function* assigning to each location a set of atomic formulae that are assumed to be true at that location,
- $\mathcal{I}_i$: $L_i \to \mathcal{C}(\mathbb{X}_i)$ is an *invariant function*, that specifies the amount of time the agent i may spend in a given location.

It is assumed that locations, actions and clocks for the environment are "public", which means that all the agents know a current location, an action, and a clock valuation of the environment.

We also assume that if $\epsilon_i \in P_i(\ell_i)$, then $t_i(\ell_i, \ell_{Ev}, \mathfrak{cc}_i, \mathbb{X}, (a_1, \ldots, a_n, a_{Ev})) = \ell_i$ for $a_i = \epsilon_i$, any $\mathfrak{cc}_i \in \mathcal{C}(\mathbb{X}_i)$, and any $\mathbb{X} \subseteq \mathbb{X}_i$. Each element t of t_i is denoted by $<\ell_i, \ell_{Ev}, \mathfrak{cc}_i, \mathbb{X}', a, \ell_i'>$, where ℓ_i is the source location, ℓ_i' is the target location, a is an action, $\mathfrak{cc}$ is the enabling condition for t_i, and $\mathbb{X}' \subseteq \mathbb{X}_i$ is the set of clocks to be reset after performing t. An invariant condition allows the $\mathbb{TIS}$ to stay at the location ℓ as long as the constraint $\mathcal{I}_i(\ell_i)$ is satisfied. The guard $\mathfrak{cc}$ has to be satisfied to enable the transition.

2.2 Timed Model

For a given $\mathbb{TIS}$ let the symbol $S = \prod_{i \in \mathcal{A} \cup \{Ev\}} (L_i \times \mathbb{N}^{\mathbb{X}_i})$ denote the non-empty set of all *global states*. Moreover, for a given global state $s = ((\ell_1, v_1), \ldots, (\ell_n, v_n), (\ell_{Ev}, v_{Ev})) \in S$, let the symbols $l_i(s) = \ell_i$ and $v_i(s) = v_i$ denote, respectively, the local component and the clock valuation of agent $i \in \mathcal{A} \cup \{Ev\}$ in s. Now, for a given $\mathbb{TIS}$ we define a *timed model* (or a *model*) as a tuple $\mathcal{M} = (Act, S, \iota, T, \mathcal{V})$, where:

- $Act = Act_1 \times \ldots \times Act_n \times Act_{Ev}$ is the set of all the joint actions,
- $S = \prod_{i \in \mathcal{A} \cup \{Ev\}} (L_i \times \mathbb{N}^{\mathbb{X}_i})$ is the set of all the *global states*,
- $\iota = \prod_{i \in \mathcal{A} \cup \{Ev\}} (\iota_i \times \{0\}^{\mathbb{X}_i})$ is the set of all the *initial* global states,
- $\mathcal{V} : S \to 2^{\mathcal{AP}}$ is the valuation function defined as $\mathcal{V}(s) = \bigcup_{i \in \mathcal{A} \cup \{Ev\}} \mathcal{V}_i(l_i(s))$,
- $T \subseteq S \times (Act \cup \{\tau\}) \times S$ is a transition relation defined by action and time transitions. For $\widetilde{a} \in Act$:

1. action transition: $(s, \widetilde{a}, s') \in T$ (or $s \xrightarrow{\widetilde{a}} s'$) iff for all $i \in \mathcal{A} \cup \{Ev\}$, there exists a local transition $t_i(l_i(s), \mathfrak{cc}_i, \mathbb{X}', \widetilde{a}) = l_i(s')$ such that $v_i(s) \models \mathfrak{cc}_i \wedge \mathcal{I}(l_i(s))$ and $v_i'(s') = v_i(s)[\mathbb{X}' := 0]$ and $v_i'(s') \models \mathcal{I}(l_i(s'))$ ($v_i(s)[\mathbb{X}' := 0]$ denotes the clock valuation which assigns 0 to each clock in $\mathbb{X}'$ and agrees with $v_i(s)$ over the rest of the clocks.
2. time transition $(s, \tau, s') \in T$ iff for all $i \in \mathcal{A} \cup \{Ev\}$, $l_i(s) = l_i(s')$ and $v_i'(s') = v_i(s) + 1$ and $v_i'(s') \models \mathcal{I}(l_i(s'))$.

A *path* π in $\mathcal{M}$ is a sequence $\pi = (s_0, s_1, \ldots)$ of states such that $(s_0, \tau, s_1) \in T$ holds and for each $i > 0$, either $(s_i, \widetilde{a}_i, s_{i+1}) \in T$ or $(s_i, \tau, s_{i+1}) \in T$, and if $(s_i, \widetilde{a}_i, s_{i+1}) \in T$ holds, then $(s_{i+1}, \tau, s_{i+2}) \in T$ holds.

Observe that the above definition of the path ensures that the first transition is the time one, and between each two action transitions at least one time transition appears.

The set of all the paths starting at $s \in S$ is denoted by $\Pi(s)$, and the set of all the paths starting at an initial state is denoted by $\Pi = \bigcup_{s^0 \in \iota} \Pi(s^0)$. Moreover, for $a \in Act \cup \{\tau\}$, we sometimes write $s \xrightarrow{a} s'$ instead of $(s, a, s') \in T$. Eventually, for $s \in S$ and $a \in Act \cup \{\tau\}$, the set of direct *a-successors* of s is defined as: $Post(s,a) = \{s' \in S | s \xrightarrow{a} s'\}$, and the set of direct successors of s is defined as $Post(s) = \bigcup_{a \in Act \cup \{\tau\}} Post(s,a)$.

Given a $\mathbb{TIS}$, one can define for any agent i the *indistinguishability* relation $\sim_i \subseteq S \times S$ as follows: $s \sim_i s'$ iff $l_i(s') = l_i(s)$ and $v_i(s') = v_i(s)$.

We assume the following definitions of epistemic relations: $\sim_\Gamma^E \overset{def}{=} \bigcup_{i \in \Gamma} \sim_i$, $\sim_\Gamma^C \overset{def}{=} (\sim_\Gamma^E)^+$ (the transitive closure of $\sim_\Gamma^E$), $\sim_\Gamma^D \overset{def}{=} \bigcap_{i \in \Gamma} \sim_i$, where $\Gamma \subseteq \mathcal{A}$.

2.3 Abstract Model

The set of all the clock valuations is infinite which means that a model has an infinite set of states. We need to abstract the proposed model before we can apply the bounded model checking technique. Let c_i be the largest constant appearing in any enabling condition or state invariants of agent i, and $v, v' \in \mathbb{N}^{|\mathbb{X}|}$ be two clock valuations. We say that $v \simeq_i v'$ iff the following condition holds for each $x \in \mathbb{X}_i$:

$$v(x) > c_i \text{ and } v'(x) > c_i \text{ or } v(x) \leq c_i \text{ and } v'(x) \leq c_i \text{ and } v(x) = v'(x).$$

Next, we define the relation $\simeq$ as follows: $v \simeq v'$ iff $v \simeq_i v'$, for every $i \in \mathcal{A} \cup \{Ev\}$. Obviously, $\simeq$ is an equivalence relation. It is easy to see that equivalent clock valuations satisfy the same clock constraints that occur in $\mathbb{TIS}$. Basing on this observation one can define the *abstract model* for $\mathbb{TIS}$. Namely, let $\mathbb{D}_i = \{0, \ldots, c_i + 1\}$, and $\mathbb{D} = \bigcup_{i \in \mathcal{A} \cup \{Ev\}} \mathbb{D}_i^{\mathbb{X}_i}$. For any $v \in \mathbb{D}$ let us define the successor $succ(v)$ of v as follows: for each $x \in \mathbb{X}$,

$$succ(v)(x) = \begin{cases} v(x) + 1, \text{ if } x \in \mathbb{X}_i \text{ and } v(x) \leq c_i, \\ \\ v(x), \qquad \text{if } x \in \mathbb{X}_i \text{ and } v(x) > c_i. \end{cases}$$

Now, one can define the *abstract model* as a tuple $\widehat{\mathcal{M}} = (Act, \widehat{S}, \widehat{\iota}, \widehat{T}, \widehat{\mathcal{V}})$, where $\widehat{S} = \prod_{i \in \mathcal{A} \cup \{Ev\}} (L_i \times \mathbb{D}_i^{\mathbb{X}_i})$ $\widehat{\iota} = \iota$, $\widehat{\mathcal{V}} = \mathcal{V}|_{\widehat{S}}$, and $\widehat{T} \subseteq \widehat{S} \times (Act \cup \{\tau\}) \times \widehat{S}$ is a transition relation defined by action and time transitions. For $\widetilde{a} \in Act$:

1. action transition: $(\widehat{s}, \widetilde{a}, \widehat{s}') \in \widehat{T}$ iff $\forall_{i \in \mathcal{A}} \exists_{\phi_i \in \mathcal{C}(\mathbb{X}_i)} \exists_{\mathbb{X}_i' \subseteq \mathbb{X}_i} (t_i(l_i(\widehat{s}), \phi_i, \mathbb{X}_i', \widetilde{a}) = l_i(\widehat{s}')$ and $v_i \models \phi_i \wedge \mathcal{I}(l_i(\widehat{s}))$ and $v_i'(\widehat{s}') = v_i(\widehat{s})[\mathbb{X}_i' := 0]$ and $v_i'(\widehat{s}') \models \mathcal{I}(l_i(\widehat{s}')))$
2. time transition: $(\widehat{s}, \tau, \widehat{s}') \in \widehat{T}$ iff $\forall_{i \in \mathcal{A} \cup \{Ev\}} (l_i(\widehat{s}) = l_i(\widehat{s}'))$ and $v_i(\widehat{s}) \models \mathcal{I}(l_i(\widehat{s}))$ and $succ(v_i(\widehat{s})) \models \mathcal{I}(l_i(\widehat{s})))$ and $\forall_{i \in \mathcal{A}} (v_i'(\widehat{s}') = succ(v_i(\widehat{s}')))$ and $(v_{Ev}'(\widehat{s}') = succ(v_{Ev}(\widehat{s})))$.

Given the abstract model one can define for any agent i the indistinguishability relation $\sim_i \subseteq \widehat{S} \times \widehat{S}$ as follows: $\widehat{s} \sim_i \widehat{s}'$ iff $l_i(\widehat{s}') = l_i(\widehat{s})$ and $v_i(\widehat{s}') = v_i(\widehat{s})$.

In the following paragraph and in the following two lemmas we assume that $\mathcal{M}$ is the timed model for a timed interpreted system $\mathbb{TIS}$, and $\widehat{\mathcal{M}}$ is the abstract model for $\mathcal{M}$ such that for each $i \in \mathcal{A} \cup \{Ev\}$, $max(\mathbb{D}_i) = c_i$.

It is easy to see that for each $v \in \mathbb{N}^{\mathbb{X}}$ there exist unique $u \in \mathbb{D}$ such that $u \simeq v$. Indeed, for $x \in \mathbb{X}_i$ let $u(x) = v(x)$ if $v(x) \leq c_i$, and $u(x) = c_i + 1$

otherwise. Clearly, $u \simeq v$. It is also easy to see that for each $v \in \mathbb{N}^{\mathbb{X}}$ and $u \in \mathbb{D}$, $v \simeq u$ implies $succ(v) \simeq succ(u)$.

The following two lemmas state that the timed and the abstract model for $\mathbb{TIS}$ are trace-equivalent. Both the lemmas can be proven by straightforward induction on j.

Lemma 1. *For each path π in $\mathcal{M}$ there exist a path $\widehat{\pi}$ in $\widehat{\mathcal{M}}$ such that $\forall j \geq 0\ \widehat{\pi}(j) \simeq \pi(j)$.*

Lemma 2. *For each path $\widehat{\pi}$ in $\widehat{\mathcal{M}}$ there exist a path π in $\mathcal{M}$ such that $\forall j \geq 0\ \pi(j) \simeq \widehat{\pi}(j)$.*

In fact it is easy to prove that the timed model and the abstract model for $\mathbb{TIS}$ are bisimulation-equivalent. Let us recall from [1] the definition of a bisimulation. Let $\mathcal{M}_i = (Act_i, S_i, \iota_i, T_i, \mathcal{V}_i)$, $i = 1, 2$ be timed models.

Definition 1. *Let $\mathcal{M}_i = (Act_i, S_i, \iota_i, T_i, \mathcal{V}_i)$, $i = 1, 2$ be timed models. A bisimulation for $(\mathcal{M}_1, \mathcal{M}_2)$ is a binary relation $\mathcal{R} \subseteq S_1 \times S_2$ such that*

- *for every $s \in \iota_1$, there exists $s_2 \in \iota_2$ such that $(s_1, s_2) \in \mathcal{R}$, and for every $s_2 \in \iota_2$, there exists $s_1 \in \iota$ such that $(s_1, s_2) \in \mathcal{R}$*
- *for all $(s_1, s_2) \in \mathcal{R}$ it holds:*
  1. *$\mathcal{V}(s_1) = \mathcal{V}(s_2)$*
  2. *if $s_1' \in Post(s_1)$ then there exists $s_2' \in Post(s_2)$ with $(s_1', s_2') \in \mathcal{R}$*
  3. *if $s_2' \in Post(s_2)$ then there exists $s_1' \in Post(s_1)$ with $(s_1', s_2') \in \mathcal{R}$.*

$\mathcal{M}_1$ and $\mathcal{M}_2$ are bisimulation-equivalent *(*bisimilar *for short), if there exists a bisimulation $\mathcal{R}$ for $(\mathcal{M}_1, \mathcal{M}_2)$.*

For a given timed model $\mathcal{M}$ and the abstract model $\widehat{\mathcal{M}}$ of $\mathcal{M}$ let us define a binary relation $\mathcal{R} \subseteq S \times \widehat{S}$ in the following way: $(s_1, s_2) \in \mathcal{R}$ iff $s_1 \simeq s_2$. Obviously, $\mathcal{R}$ is a bisimulation for $(\mathcal{M}, \widehat{\mathcal{M}})$. As a result from this fact we obtain the following lemma.

Lemma 3. *Let $\mathcal{M}$ be a timed model and $\widehat{\mathcal{M}}$ be the abstract model of $\mathcal{M}$. Then, $\mathcal{M}$ and $\widehat{\mathcal{M}}$ are bisimilar.*

2.4 RTECTLK

Multi-agent Systems (MAS) formalisms are typically built on extensions of computational tree logic (CTL). For the purposes of this paper we consider specifications given in the RTECTLK language built from a set of propositional formulae $p \in \mathcal{AP}$, and a set of agents $i \in \mathcal{A}$. An existential fragment of the soft real-time CTL (RTECTL) [7] is a propositional branching-time temporal logic with bounded operators, which was introduced to permit specification and reasoning about time-critical correctness properties. The RTECTLK [21] language is an epistemic soft real-time computation tree logic that is the fusion [3] of the two underlying languages: RTECTL and $S5_n$ for the knowledge operators [8].

Syntax of RTECTLK. Let $\mathcal{AP}$ be a set of atomic formulae, $\mathcal{A}$ a set of agents, and I be an interval in $\mathbb{N}$ of the form: $[a, b)$ and $[a, \infty)$, for $a, b \in \mathbb{N}$. Hereafter by $\mathbf{left}(\mathrm{I})$ we denote the left end of the interval I, i.e., $\mathbf{left}(\mathrm{I}) = a$, and by $\mathbf{right}(\mathrm{I})$ the right end of the interval I, i.e., $\mathbf{right}([a, b)) = b - 1$ and $\mathbf{right}([a, \infty)) = \infty$.

Let $p \in \mathcal{AP}$, $i \in \mathcal{A}$, and $\Gamma \subseteq \mathcal{A}$. The set of RTECTLK formulae is defined by the following grammar:

$$\varphi := \mathbf{true} \mid \mathbf{false} \mid p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{EX}\varphi \mid \mathbf{E}(\varphi \mathbf{U}_{\mathrm{I}} \varphi) \mid \mathbf{EG}_{\mathrm{I}}\varphi \mid$$
$$\overline{\mathrm{K}}_i \varphi \mid \overline{\mathrm{D}}_\Gamma \varphi \mid \overline{\mathrm{E}}_\Gamma \varphi \mid \overline{\mathrm{C}}_\Gamma \varphi$$

$\mathbf{U}_{\mathrm{I}}$ and $\mathbf{G}_{\mathrm{I}}$ are the operators, resp., for bounded "Until" and "Always". The formula $\mathbf{EG}_{\mathrm{I}}\alpha$ is read as "there exists a computation such that α always holds in the interval I" and the formula $\mathbf{E}(\alpha \mathbf{U}_{\mathrm{I}} \beta)$ is read as "there exists a computation such that β holds in the interval I at least in one state and always earlier α holds". The other basic bounded temporal operators can be introduced as usual:

$$\mathbf{E}(\alpha\, \mathbf{R}_{\mathrm{I}}\, \beta) \overset{def}{=} \mathbf{E}(\beta\, \mathbf{U}_{\mathrm{I}}\, (\alpha \wedge \beta)) \vee \mathbf{EG}_{\mathrm{I}}\beta, \qquad \mathbf{EF}_{\mathrm{I}}\alpha \overset{def}{=} \mathbf{E}(\mathbf{true}\mathbf{U}_{\mathrm{I}}\alpha).$$

$\overline{\mathrm{K}}_i$ is the operator dual for the standard epistemic modality K_i ("agent i knows"), so $\overline{\mathrm{K}}_i\alpha$ is read as "agent i does not know whether or not α holds". Similarly, the modalities $\overline{\mathrm{D}}_\Gamma, \overline{\mathrm{E}}_\Gamma, \overline{\mathrm{C}}_\Gamma$ are the diamonds for $\mathrm{D}_\Gamma, \mathrm{E}_\Gamma, \mathrm{C}_\Gamma$ representing distributed knowledge in the group Γ, "everyone in Γ knows", and common knowledge among agents in the group Γ.

Semantics of RTECTLK. As the semantics for the timed model and the semantics for the abstract model are identical we shall present only the semantics for the abstract model. Let $\widehat{\mathcal{M}} = (Act, \widehat{S}, \widehat{\iota}, \widehat{T}, \widehat{\mathcal{V}})$ denote the abstract model. An abstract path $\widehat{\pi}$ in the abstract model is a sequence $\widehat{s}_0 \xrightarrow{b_1} \widehat{s}_1 \xrightarrow{b_2} \widehat{s}_2 \xrightarrow{b_3} \ldots$ of transitions such that for each $i > 1$, $b_i \in Act \cup \{\tau\}$ and $b_1 = \tau$ and for each two consecutive transitions at least one of them is a time transition. The set of all the abstract paths starting at $\widehat{s} \in \widehat{S}$ is denoted by $\widehat{\Pi}(\widehat{s})$, and the set of all the abstract paths starting at an abstract initial state is denoted by $\widehat{\Pi} = \bigcup_{\widehat{s}^0 \in \widehat{\iota}} \widehat{\Pi}(\widehat{s}^0)$.

For the group of epistemic modalities we also define the following. If $\Gamma \subseteq \mathcal{A}$, then $\sim^E_\Gamma \overset{def}{=} \bigcup_{i \in \Gamma} \sim_i$, $\sim^C_\Gamma \overset{def}{=} (\sim^E_\Gamma)^+$ (the transitive closure of $\sim^E_\Gamma$), and $\sim^D_\Gamma \overset{def}{=} \bigcap_{i \in \Gamma} \sim_i$.

A RTECTLK formula φ is *true* in the abstract model $\widehat{\mathcal{M}}$ (in symbols $\widehat{\mathcal{M}} \models \varphi$) iff $\widehat{\mathcal{M}}, \widehat{s}^0 \models \varphi$ for some $\widehat{s}^0 \in \widehat{\iota}$ (i.e., φ is true at some abstract initial state of the abstract model $\widehat{\mathcal{M}}$). For every $\widehat{s} \in \widehat{S}$ the relation $\models$ is defined inductively as follows:

- $\widehat{\mathcal{M}}, \widehat{s} \models p$ iff $p \in \widehat{\mathcal{V}}(\widehat{s})$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \neg p$ iff $p \notin \widehat{\mathcal{V}}(\widehat{s})$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \alpha \wedge \beta$ iff $\widehat{\mathcal{M}}, \widehat{s} \models \alpha$ and $\widehat{\mathcal{M}}, \widehat{s} \models \beta$,

- $\widehat{\mathcal{M}}, \widehat{s} \models \alpha \vee \beta$ iff $\widehat{\mathcal{M}}, \widehat{s} \models \alpha$ or $\widehat{\mathcal{M}}, \widehat{s} \models \beta$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \mathbf{EX}\alpha$ iff $(\exists \widehat{\pi} \in \widehat{\Pi}(\widehat{s}))(\widehat{\mathcal{M}}, \widehat{\pi}(1) \models \alpha)$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \mathbf{E}(\alpha \mathbf{U}_{\mathrm{I}} \beta)$ iff $(\exists \widehat{\pi} \in \widehat{\Pi}(\widehat{s}))(\exists m \in \mathrm{I})[\widehat{\mathcal{M}}, \widehat{\pi}(m) \models \beta$ and $(\forall j < m)\widehat{\mathcal{M}}, \widehat{\pi}(j) \models \alpha]$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \mathbf{EG}_{\mathrm{I}}\alpha$ iff $(\exists \widehat{\pi} \in \widehat{\Pi}(\widehat{s}))$ such that $(\forall m \in \mathrm{I})[\widehat{\mathcal{M}}, \widehat{\pi}(m) \models \alpha]$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \overline{\mathrm{K}}_i\alpha$ iff $(\exists \widehat{s}' \in \widehat{S})(\widehat{s} \sim \widehat{s}'$ and $\widehat{\mathcal{M}}, \widehat{s}' \models \alpha)$,
- $\widehat{\mathcal{M}}, \widehat{s} \models \overline{Y}\alpha$ iff $(\exists \widehat{s}' \in \widehat{S})(\widehat{s} \sim \widehat{s}'$ and $\widehat{\mathcal{M}}, \widehat{s}' \models \alpha)$, where $\overline{Y} \in \{\overline{\mathrm{D}}_\Gamma, \overline{\mathrm{E}}_\Gamma, \overline{\mathrm{C}}_\Gamma\}$, and $\sim \in \{\sim^D_\Gamma, \sim^E_\Gamma, \sim^C_\Gamma\}$.

We end the section by defining the notions of validity and the model checking problem. Namely, a RTECTLK formula φ is *valid* in $\widehat{\mathcal{M}}$ (denoted $\widehat{\mathcal{M}} \models \varphi$) iff $\widehat{\mathcal{M}}, \widehat{\iota} \models \varphi$, i.e., φ is true at the abstract initial state of the abstract model $\widehat{\mathcal{M}}$. The *model checking problem* asks whether $\widehat{\mathcal{M}} \models \varphi$.

From the fact that the timed model and the abstract model for $\mathbb{TIS}$ are bisimulation-equivalent it follows that the same formulae are true in both the models.

We state the following theorem:

Theorem 1. *Let $\mathcal{M}$ be the timed model, φ an RTECTLK formula, and $\widehat{\mathcal{M}}$ be the abstract model of $\mathcal{M}$. Then, $\mathcal{M} \models \varphi$ iff $\widehat{\mathcal{M}} \models \varphi$.*

3 SMT-based Bounded Model Checking

In this section we present an outline of the bounded semantics for RTECTLK and define a SMT-based BMC for RTECTLK, which is based on the BMC encoding presented in [21]. The main difference between the SAT-based encoding and the SMT-base encoding is the representation of symbolic states, and symbolic actions. In effect, the SMT-based encoding is the generalisation of the propositional encoding.

RTECTLK formulae can be checked by BMC on an abstract model instead of on the original model. The model checking for this class of formulae is decidable. The complexity of standard SMT-based BMC is double exponential [5].

The SMT-based BMC is based on the notion of the bounded semantics, the definition of which requires the concept of k-paths and loops.

3.1 Bounded Semantics

Let $\widehat{\mathcal{M}} = (Act, \widehat{S}, \widehat{\iota}, \widehat{T}, \widehat{\mathcal{V}})$ be an abstract model and $k \geq 0$. A *k-path* $\widehat{\pi}_k$ in $\widehat{\mathcal{M}}$ is a finite sequence of abstract states $(\widehat{s}_0, \ldots, \widehat{s}_k)$ such that $(\widehat{s}_j, \widehat{s}_{j+1}) \in \widehat{T}$ for each $0 \leq j < k$. By $\widehat{\Pi}_k(\widehat{s})$ we denote the set of all the k-paths starting at $\widehat{s}$ in $\widehat{\mathcal{M}}$, and $\widehat{\Pi}_k = \bigcup_{\widehat{s} \in \widehat{S}} \widehat{\Pi}_k(\widehat{s})$. A k-path $\widehat{\pi}_k$ is a (k,l)-*loop* iff $\widehat{\pi}_k(l) = \widehat{\pi}_k(k)$ for some $0 \leq l < k$; note that (k,l)-loop $\widehat{\pi}$ generates the infinite path of the following form: $\zeta \cdot \theta^\omega$ with $\zeta = (\widehat{\pi}(0), \ldots, \widehat{\pi}(l-1))$ and $\theta = (\widehat{\pi}(l), \ldots, \widehat{\pi}(k-1))$. Since in the bounded semantics we consider finite prefixes of paths only, the satisfiability of all the temporal operators depends on whether a considered k-path is a loop. Thus, as customary, we introduce a function $loop : \widehat{\Pi}_k \to 2^{\mathbb{N}}$, which identifies these k-paths that are loops. The function is defined as: $loop(\widehat{\pi}_k) = \{l \mid 0 \leq l < k$ and $\widehat{\pi}_k(l) = \widehat{\pi}_k(k)\}$.

Definition 2. *Given are a bound $k \in \mathbb{N}$, an abstract model $\widehat{\mathcal{M}}$, and RTECTLK formulae α, β. $\widehat{\mathcal{M}}, \widehat{s} \models_k \alpha$ denotes that α is $k-$true at the abstract state $\widehat{s}$ of $\widehat{\mathcal{M}}$. The relation $\models_k$ is defined inductively as follows:*

- $\widehat{\mathcal{M}}, \widehat{s} \models_k \mathbf{true}$, $\widehat{\mathcal{M}}, \widehat{s} \not\models_k \mathbf{false}$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k p$ *iff* $p \in \widehat{\mathcal{V}}(\widehat{s})$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \neg p$ *iff* $p \notin \widehat{\mathcal{V}}(\widehat{s})$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \alpha \vee \beta$ *iff* $\widehat{\mathcal{M}}, \widehat{s} \models_k \alpha$ *or* $\widehat{\mathcal{M}}, \widehat{s} \models_k \beta$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \alpha \wedge \beta$ *iff* $\widehat{\mathcal{M}}, \widehat{s} \models_k \alpha$ *and* $\widehat{\mathcal{M}}, \widehat{s} \models_k \beta$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \boldsymbol{EX}\alpha$ *iff* $k > 0$ *and* $(\exists \widehat{\pi} \in \widehat{\Pi}_k(\widehat{s}))\widehat{\mathcal{M}}, \widehat{\pi}(1) \models_k \alpha$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \boldsymbol{E}(\alpha \boldsymbol{U}_I \beta)$ *iff* $(\exists \widehat{\pi} \in \widehat{\Pi}_k(\widehat{s}))(\exists 0 \leq m \leq k)(m \in \mathrm{I}$ *and* $\widehat{\mathcal{M}}, \pi(m) \models_k \beta$ *and* $(\forall 0 \leq j < m)\widehat{\mathcal{M}}, \widehat{\pi}(j) \models_k \alpha)$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \boldsymbol{EG}_I\alpha$ *iff* $(\exists \widehat{\pi} \in \widehat{\Pi}_k(\widehat{s}))((k \geq \mathbf{right}(\mathrm{I})$ *and* $(\forall j \in \mathrm{I})\ \widehat{\mathcal{M}}, \widehat{\pi}(j) \models_k \alpha)$ *or* $(k < \mathbf{right}(\mathrm{I})$ *and* $(\exists l \in loop(\widehat{\pi}))(\forall min(\mathbf{left}(\mathrm{I}), l) \leq j < k)\ \widehat{\mathcal{M}}, \widehat{\pi}(j) \models_k \alpha))$,
- $\widehat{\mathcal{M}}, \widehat{s} \models_k \overline{Y}\alpha$ *iff* $(\exists \widehat{\pi} \in \widehat{\Pi}_k(\widehat{\iota}))(\exists 0 \leq j \leq k)(\widehat{\mathcal{M}}, \pi(j) \models_k \alpha$ *and* $\widehat{s} \sim \widehat{\pi}(j))$, *where* $\overline{Y} \in \{\overline{\mathrm{K}}_i, \overline{\mathrm{D}}_\Gamma, \overline{\mathrm{E}}_\Gamma, \overline{\mathrm{C}}_\Gamma\}$ *and* $\sim \in \{\sim_i, \sim^D_\Gamma, \sim^E_\Gamma, \sim^C_\Gamma\}$.

A RTECTLK formula φ is *valid in an abstract model $\widehat{\mathcal{M}}$ with a bound k* (denoted $\widehat{\mathcal{M}} \models_k \varphi$) iff $\widehat{\mathcal{M}}, \widehat{\iota} \models_k \varphi$, i.e., φ is $k-$true at the abstract initial state of the abstract model $\widehat{\mathcal{M}}$. The *bounded model checking problem* asks whether $\widehat{\mathcal{M}} \models_k \varphi$.

By straightforward induction on the length of a RTECTLK formula φ we can show that the following lemmas hold.

Lemma 4. *Given are a bound $k \geq 0$, an abstract model $\widehat{\mathcal{M}}$, and a RTECTLK formula φ. Then, the following implication holds: $\widehat{\mathcal{M}}, \widehat{s} \models_k \varphi$ implies $\widehat{\mathcal{M}}, \widehat{s} \models \varphi$, for each $\widehat{s}$ in $\widehat{\mathcal{M}}$.*

Lemma 5. *Given are an abstract model $\widehat{\mathcal{M}}$, a bound $k = |\widehat{\mathcal{M}}|$ (where $|\widehat{\mathcal{M}}|$ denotes the number of states in the abstract model $\widehat{\mathcal{M}}$), and a RTECTLK formula φ. Then, the following implication holds: for each $\widehat{s}$ in $\widehat{\mathcal{M}}$, if $\widehat{\mathcal{M}}, \widehat{s} \models \varphi$, then there exists $k \geq 0$ such that $\widehat{\mathcal{M}}, \widehat{s} \models_k \varphi$.*

The following theorem states that there exists a bound such that bounded semantics is equivalent to the unbounded one, which means that the model checking problem ($\widehat{\mathcal{M}} \models \varphi$) can be reduced to the bounded model checking problem ($\widehat{\mathcal{M}} \models_k \varphi$). Its proof follows from Lemmas 4 and 5.

Theorem 2. *Let $\widehat{\mathcal{M}}$ be an abstract model and φ a RTECTLK formula. Then, the following equivalence holds: $\widehat{\mathcal{M}} \models \varphi$ iff there exists $k \geq 0$ such that $\widehat{\mathcal{M}} \models_k \varphi$.*

The reduction of RTECTLK to the quantifier-free first-order formula allows us to use efficient SMT solvers to perform model checking. A function $\widehat{f}_k$ that gives a bound on the number of k-paths of $\widehat{\mathcal{M}}$, which are sufficient to validate a given RTECTLK formula is defined in [21].

By straightforward induction on the length of a RTECTLK formula φ we can show that φ is $k-$true in $\widehat{\mathcal{M}}$ if and only if φ is $k-$true in $\widehat{\mathcal{M}}$ with a number of $k-$paths reduced to $\widehat{f}_k(\varphi)$.

3.2 The Translation of RTECTLK to the Quantifier-Free First-Order Formulae

Now we present our translation of a RTECTLK formula into a quantifier-free first-order formula. Given are a model $\widehat{\mathcal{M}} = (Act, \widehat{S}, \widehat{\iota}, \widehat{T}, \widehat{\mathcal{V}})$, a RTECTLK formula φ, and a bound $k \geq 0$. It is well known that the main idea of the BMC method consists in translating the bounded model checking problem, i.e., $\widehat{\mathcal{M}} \models {}_k\varphi$, to the problem of checking the satisfiability of the following propositional formula:

$$[\widehat{\mathcal{M}}, \varphi]_k := [\widehat{\mathcal{M}}^{\varphi, \widehat{\iota}}]_k \wedge [\varphi]_{\widehat{\mathcal{M}},k}$$

The formula $[\widehat{\mathcal{M}}^{\varphi, \widehat{\iota}}]_k$ constrains the $f_k(\varphi)$ symbolic k-paths to be valid k-paths of $\widehat{\mathcal{M}}$, while the formula $[\varphi]_{\widehat{\mathcal{M}},k}$ encodes a number of constraints that must be satisfied on these sets of k-paths for φ to be satisfied. Once this translation is defined, checking satisfiability of a RTECTLK formula can be done by means of a SMT-solver.

Let $i \in \mathcal{A} \cup \{Ev\}$. In order to define the formula $[\widehat{\mathcal{M}}, \varphi]_k$ we proceed as follows. We assume that each abstract global state $\widehat{s} \in \widehat{S}$ of $\widehat{\mathcal{M}}$ is represented by a valuation of a *symbolic global state* $\overline{\mathbf{w}} = ((u_1, v_1), \ldots, (u_n, v_n), (u_{Ev}, v_{Ev}))$ that consists of *symbolic local states* and each symbolic local state w_i is a pair (u_i, v_i) of individual variables ranging over the natural numbers, in which the first element represents a location of the agent i, and the second represents the clocks valuation. Each joint action $a \in Act$ is represented by a valuation of a *symbolic action* $\overline{\mathbf{a}} = (a_1, \ldots, a_n, a_{Ev})$ that consists of *symbolic local actions* and each symbolic local action $\widetilde{a}_i$ is an individual variable ranging over the natural numbers.

In order to define the formula $[\widehat{\mathcal{M}}, \varphi]_k$ we proceed as follows. A finite sequence $(\overline{\mathbf{w}}_0, \ldots, \overline{\mathbf{w}}_k)$ of *symbolic states* is called a *symbolic k-path*. Since, in general, we may need to consider more than one symbolic k-path, we introduce a notion of the j-th symbolic k-path, which is denoted by $(\overline{\mathbf{w}}_{0,j}, \ldots, \overline{\mathbf{w}}_{k,j})$, where $\overline{\mathbf{w}}_{i,j}$ are *symbolic states* for $0 \leq j < f_k(\varphi)$ and $0 \leq i \leq k$. Note that the exact number of necessary symbolic k-paths depends on the checked formula φ, and it can be calculated by means of the function f_k [21]. We define the following quantifier-free first-order formulae:

- $I_{\widehat{s}}(\overline{\mathbf{w}})$ - it encodes the abstract global state $\widehat{s}$ of the abstract model $\widehat{\mathcal{M}}$;
- $H_i(w_i, w_i')$ - it encodes equality of two local states, such that $w_i = w_i'$ for $i \in \mathcal{A} \cup \{Ev\}$;
- $\mathcal{T}_i(w_i, (\widetilde{a}, \delta), w'_i)$ - it encodes the local evolution function of agent i;

– $\mathcal{A}(\overline{\mathbf{a}})$ - it encodes that each symbolic local action a_i of $\overline{\mathbf{a}}$ has to be executed by each agent in which it appears;
– $\mathcal{T}(\overline{\mathbf{w}}, (\overline{\mathbf{a}}, \delta), \overline{\mathbf{w}}') := \mathcal{A}(\overline{\mathbf{a}}) \wedge \bigwedge_{i \in \mathcal{A} \cup \{Ev\}} \mathcal{T}_i(w_i, (\overline{\mathbf{a}}, \delta), w_i')$;
– Let $\boldsymbol{\pi}_j$ denote the j-th *symbolic k-path*, i.e. the sequence of symbolic transitions: $\overline{\mathbf{w}}_{0,j} \xrightarrow{\overline{\mathbf{a}}_{1,j}, \delta_{1,j}} \overline{\mathbf{w}}_{1,j} \xrightarrow{\overline{\mathbf{a}}_{2,j}, \delta_{2,j}} \dots \xrightarrow{\overline{\mathbf{a}}_{k,j}, \delta_{k,j}} \overline{\mathbf{w}}_{k,j}$.

Thus, given the above, we can define the formula $[\widehat{\mathcal{M}}^{\varphi,\widehat{\iota}}]_k$ as follows:

$$[\widehat{\mathcal{M}}^{\varphi,\widehat{\iota}}]_k := \bigvee_{s \in \widehat{\iota}} I_s(\overline{\mathbf{w}}_{0,0}) \wedge \bigvee_{j=1}^{\widehat{f_k}(\varphi)} \overline{\mathbf{w}}_{0,0} = \overline{\mathbf{w}}_{0,j} \wedge \bigwedge_{j=1}^{\widehat{f_k}(\varphi)} \bigwedge_{i=0}^{k-1} \mathcal{T}(\overline{\mathbf{w}}_{i,j}, (\overline{\mathbf{a}}_{i,j}, \delta_{i,j}), \overline{\mathbf{w}}_{i+1,j})$$

where $\overline{\mathbf{w}}_{i,j}$ and $\overline{\mathbf{a}}_{i,j}$ are, respectively, symbolic states, symbolic actions for $0 \leq i \leq k$ and $1 \leq j \leq f_k(\varphi)$.

The formula $[\varphi]_{\mathcal{M},k}$ encodes the bounded semantics of the RTECTLK formula φ, and it is defined on the same sets of individual variables as the formula $[\mathcal{M}^{\varphi,\iota}]_k$. Moreover, it uses the auxiliary quantifier-free first-order formulae defined in [20].

Furthermore, following [20], our formula $[\varphi]_{\mathcal{M},k}$ uses the following auxiliary functions g_l, g_r, g_μ, h_k^U, h_k^G which were introduced in [23], and which allow us to divide the set $A \subseteq F_k(\varphi) = \{j \in \mathbb{N} \mid 1 \leq j \leq f_k(\varphi)\}$ into subsets necessary for translating the sub-formulae of φ.

Definition 3 (Translation of RTECTLK Formulae). *Let φ be a RTECTLK formula, and $k \geq 0$ a bound. We define inductively the translation of φ over path number $n \in F_k(\varphi)$ starting at symbolic state $\overline{\mathbf{w}}_{m,n}$ as shown below.*

- $[\mathbf{EX}\alpha]_k^{[m,n,A]} :=$ *(1)* $\overline{\mathbf{w}}_{m,n} = \overline{\mathbf{w}}_{0,min(A)} \wedge [\alpha]_k^{[1,min(A),g_s(A)]}$, if $k > 0$
  (2) **false**, otherwise,
- $[\mathbf{E}(\alpha \mathbf{U}_{\mathrm{I}} \beta)]_k^{[m,n,A]} := \overline{\mathbf{w}}_{m,n} = \overline{\mathbf{w}}_{0,min(A)} \wedge \bigvee_{i=0}^{k}([\beta]_k^{[i,min(A),h_{\mathbf{U}}(A,k,f_k(\beta))(k)]}$
  $\wedge In(i,\mathrm{I}) \wedge \bigwedge_{j=0}^{i-1}[\alpha]_k^{[j,min(A),h_{\mathbf{U}}(A,k,f_k(\beta))(j)]})$,
- $[\mathbf{EG}_{\mathrm{I}}\alpha]_k^{[m,n,A]} :=$ $\overline{\mathbf{w}}_{m,n} = \overline{\mathbf{w}}_{0,min(A)} \wedge$
  (1) $\bigwedge_{j=\mathbf{left}(\mathrm{I})}^{\mathbf{right}(\mathrm{I})}[\alpha]_k^{[j,min(A),h_{\mathbf{G}}(A,k)(j)]}$, if $\mathbf{right}(\mathrm{I}) \leq k$
  (2) $\bigvee_{l=0}^{k-1}(\overline{\mathbf{w}}_{k,min(A)} = \overline{\mathbf{w}}_{l,min(A)}$
  $\wedge \bigwedge_{j=min(\mathbf{left}(\mathrm{I}),l)}^{k-1}[\alpha]_k^{[j,min(A),h_{\mathbf{G}}(A,k)(j)]})$, otherwise,
- $[\overline{\mathrm{K}}_i\alpha]_k^{[m,n,A]} :=$ $I_\iota(\overline{\mathbf{w}}_{0,min(A)}) \wedge \bigvee_{j=0}^{k}([\alpha]_k^{[j,min(A),g_s(A)]}$
  $\wedge H_i(\overline{\mathbf{w}}_{m,n}, \overline{\mathbf{w}}_{j,min(A)}))$
- $[\overline{\mathrm{D}}_\Gamma\alpha]_k^{[m,n,A]} :=$ $I_\iota(\overline{\mathbf{w}}_{0,min(A)}) \wedge \bigvee_{j=0}^{k}([\alpha]_k^{[j,min(A),g_s(A)]}$
  $\wedge \bigwedge_{i \in \Gamma} H_i(\overline{\mathbf{w}}_{m,n}, \overline{\mathbf{w}}_{j,min(A)}))$,
- $[\overline{\mathrm{E}}_\Gamma\alpha]_k^{[m,n,A]} :=$ $I_\iota(\overline{\mathbf{w}}_{0,min(A)}) \wedge \bigvee_{j=0}^{k}([\alpha]_k^{[j,min(A),g_s(A)]}$
  $\wedge \bigvee_{i \in \Gamma} H_i(\overline{\mathbf{w}}_{m,n}, \overline{\mathbf{w}}_{j,min(A)}))$,
- $[\overline{\mathrm{C}}_\Gamma\alpha]_k^{[m,n,A]} :=$ $[\bigvee_{j=1}^{k}(\overline{\mathrm{E}}_\Gamma)^j\alpha]_k^{[m,n,A]}$.

The theorem below states the correctness and the completeness of the presented translation. It can be proven by induction on the structure of the given RTECTLK formula.

Theorem 3. *Let $\widehat{\mathcal{M}}$ be an abstract model, and φ a RTECTLK formula. For every $k \in \mathbb{N}$, $\widehat{\mathcal{M}} \models {}_k\varphi$ if, and only if, the quantifier-free first-order formula $[\widehat{\mathcal{M}}, \varphi]_k$ is satisfiable.*

4 Experimental Results

In this section we experimentally evaluate the performance of our SMT-based BMC encoding for RTECTLK over the $\mathbb{TIS}$ semantics. We have conducted the experiments using two benchmarks that are no yet widely used in the multi-agent community: the timed generic pipeline paradigm (TGPP) $\mathbb{TIS}$ model [22] and the timed train controller system (TTCS) $\mathbb{TIS}$ model [22]. We would like to point out that both benchmarks are very useful and scalable examples.

TGPP. The abstract model of TGPP involves $n+2$ agents: Producer producing data within the certain time interval ($[a, b]$) or being inactive, Consumer receiving data within the certain time interval ($[c, d]$) or being inactive within the certain time interval ($[g, h]$), a chain of n intermediate Nodes which can be ready for receiving data within the certain time interval ($[c, d]$), processing data within the certain time interval ($[e, f]$) or sending data, and the environment Ev. The weights are used to adjust the cost properties of Producer, Consumer, and of the intermediate Nodes.

Each agent of the scenario can be modelled by considering its local states, the local actions, the local protocol, the local evolution function, the local weight function, the local clocks, the clock constraints, the invariants, and the local valuation function. Figure 1 shows the local states, the possible actions, and the protocol, the clock constraints, invariants and weights for each agent. Null actions are omitted in the figure. For environment, we shall consider just one local state: $L_{Ev} = \{\cdot\}$. The set of actions for Ev is $Act_{Ev} = \{\epsilon_{Ev}\}$. The local protocols of Ev is the following: $P_{Ev}(\cdot) = Act_{Ev}$. The set of clocks of Ev is empty, and the invariant function is $\mathcal{I}_{Ev}(\cdot) = \{\emptyset\}$.

Given Fig. 1, the local evolution functions of TGPP are straightforward to infer. Moreover, we assume the following set of propositional variables: $\mathcal{AP} = \{ProdReady, ProdSend, ConsReady, ConsFree\}$ with the following definitions of local valuation functions: $\widehat{\mathcal{V}}_P(ProdReady) = \{ProdReady\}$, $\widehat{\mathcal{V}}_P(ProdSend) = \{ProdSend\}$; $\widehat{\mathcal{V}}_C(ConsReady) = \{ConsReady\}$, $\widehat{\mathcal{V}}_C(ConsFree) = \{ConsFree\}$.

Let $Act = Act_P \times \prod_{i=1}^{n} Act_{N_i} \times Act_C \times Act_{Ev}$, with $Act_P = \{Produce, Send_1, \epsilon_P\}$, $Act_C = \{Start_{n+1}, Consume, Send_{n+1}, \epsilon_C\}$, $Act_{N_i} = \{Start_i, Send_i, Send_{i+1}, Proc_i, \epsilon_{N_i}\}$, and $Act_{Ev} = \{\epsilon_{Ev}\}$ defines the set of joint actions for the scenario. For $\widetilde{a} \in Act$ let $act_P(\widetilde{a})$ denotes an action of Producer, $act_C(\widetilde{a})$ denotes an action of Consumer, $act_{N_i}(\widetilde{a})$ denotes an action of Node i,

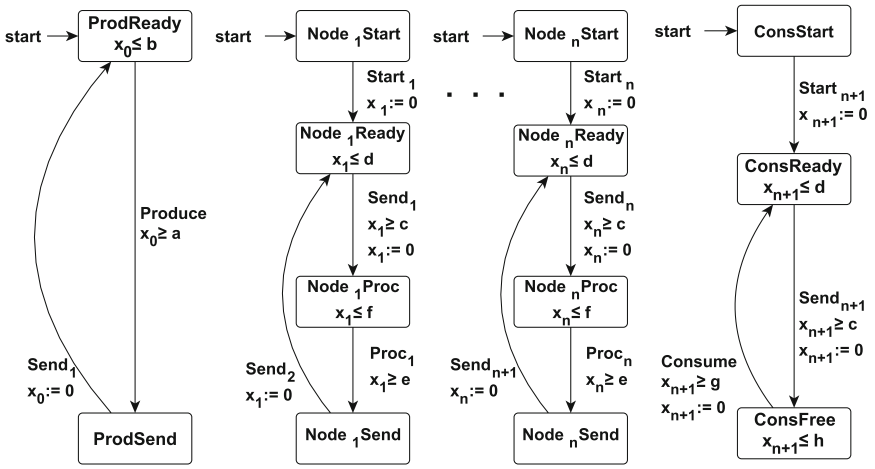


Fig. 1. The TGPP system

and $act_{Ev}(\widetilde{a})$ denotes an action of environment Ev. We assume the following local evolution functions:

- $t_P(ProdReady, \cdot, x_0 \geq a, \emptyset, \widetilde{a}) = ProdSend$, if $act_P(\widetilde{a}) = Produce$;
- $t_P(ProdSend, \cdot, true, \{x_0\}, \widetilde{a}) = ProdReady$, if $act_P(\widetilde{a}) = Send_1$ and $act_{N_i}(\widetilde{a}) = Send_1$;
- $t_C(ConsStart, \cdot, true, \{x_{n+1}\}, \widetilde{a}) = ConsReady$, if $act_C(\widetilde{a}) = Start_{n+1}$;
- $t_C(ConsReady, \cdot, x_{n+1} \geq c, \{x_{n+1}\}, \widetilde{a}) = ConsFree$, if $act_C(\widetilde{a}) = Send_{n+1}$ and $act_{N_n}(\widetilde{a}) = Send_{n+1}$;
- $t_C(ConsFree, \cdot, x_{n+1} \geq g, \{x_{n+1}\}, \widetilde{a}) = ConsReady$, if $act_C(\widetilde{a}) = Consume$.

The set of all the global states $\widehat{S}$ for the scenario is defined as the product $(L_P \times \mathbb{D}_P^{|\mathbb{X}_P|}) \times \prod_{i=1}^{n}(L_{N_i} \times \mathbb{D}_{N_i}^{|\mathbb{X}_{N_i}|}) \times (L_C \times \mathbb{D}_{L_C}^{|\mathbb{X}_{L_C}|}) \times L_{Ev}$. The set of the initial states is defined as $\widehat{\iota} = \{s^0\}$, where $s^0 = ((ProdReady, 0), (Node_1Start, 0), \ldots, (Node_nStart, 0), (ConsStart, 0), (\cdot))$.

The specifications we consider are as follows:

- $\varphi_1 = \mathbf{EF}_{[0,\infty)}(ProdSend \wedge \mathbf{EG}_{[a,\infty)}\overline{\mathrm{K}}_C\overline{\mathrm{K}}_P(Received))$, where $a = 2n+1$ and $n \geq 1$ – *states that it is not true that if Producer produces a product, then ultimately in a or more steps, Consumer knows that Producer does not know that Consumer has the product.*
- $\varphi_2 = \mathbf{EF}_{[0,\infty)}\overline{\mathrm{K}}_P(ProdSend \wedge \mathbf{EF}_{[0,4)}(Received))$ – *expresses that it is not true that Producer knows that if he produces a product, then always within the next three steps later Consumer does not have the product.*
- $\varphi_3 = \mathbf{EF}_{[0,\infty)}\overline{\mathrm{K}}_P(ProdSend \wedge \mathbf{EF}_{[n,n+4)}(Received))$ – *states that it is not true that Producer knows that if he produces a product, then always within interval* $[n, n+4)$ *Consumer does not have the product.*

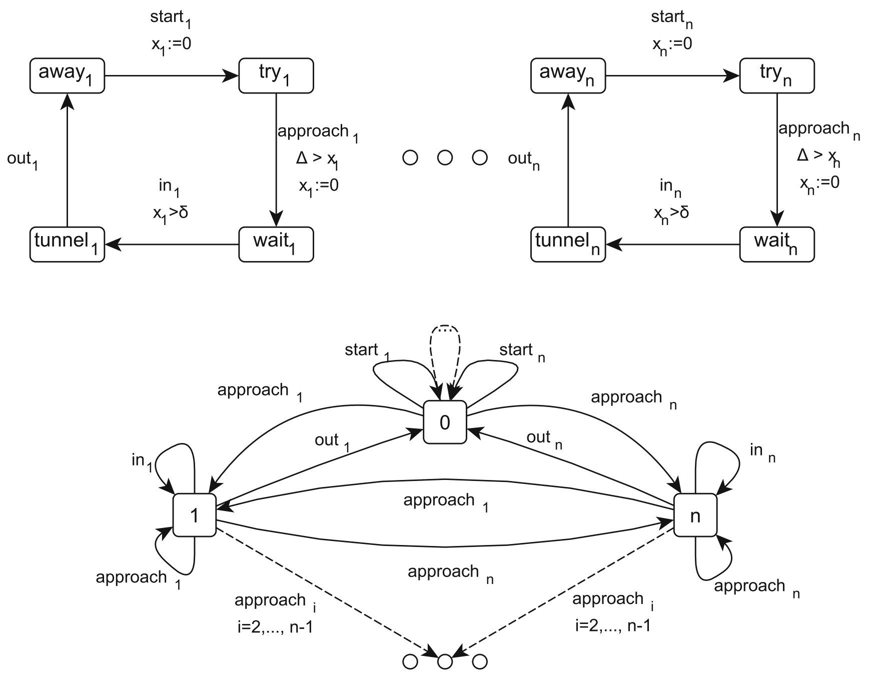


Fig. 2. The TTCS system

TTCS. The TTCS consists of n (for $n \geq 2$) trains $T_1, \ldots, T_n$, each one using its own circular track for travelling in one direction and containing its own clock x_i, together with controller C used to coordinate the access of trains to the tunnel through all trains have to pass at certain point, and the environment Ev. There is only one track in the tunnel, so trains arriving from each direction cannot use it in this same time. There are signals on both sides of the tunnel, which can be either red or green. All trains notify the controller when they request entry to the tunnel or when they leave the tunnel. The controller controls the colour of the displayed signal, and the behaviour of the scenario depends on the values δ and Δ ($\Delta > \delta+3$ makes it incorrect - the mutual exclusion does not hold) (Fig. 2).

Controller C has $n+1$ states, denoting that all trains are away (state 0), and the numbers of trains, i.e., $1, \ldots, n$. Controller C is initially at state 0. The action $Start_i$ of train T_i denotes the passage from state away to the state where the train wishes to obtain access to the tunnel. This is allowed only if controller C is in state 0. Similarly, train T_i synchronises with controller C on action $approach_i$, which denotes setting C to state i, as well as out_i, which denotes setting C to state 0. Finally, action in_i denotes the entering of train T_i into the tunnel. For environment, we shall consider just one local state: $L_{Ev} = \{\cdot\}$. The set of actions for Ev is $Act_{Ev} = \{\epsilon_{Ev}\}$. The local protocols of Ev is the following: $P_{Ev}(\cdot) = Act_{Ev}$. The set of clocks of Ev is empty, and the invariant function is $\mathcal{I}_{Ev}(\cdot) = \{\emptyset\}$.

The set of all the global states $\widehat{S}$ for the scenario is defined as the product $\prod_{i=1}^{n}(L_{T_i} \times \mathbb{D}_{T_i}^{|\mathbb{X}_{T_i}|}) \times (L_C \times \mathbb{D}_{L_C}^{|\mathbb{X}_{L_C}|}) \times L_{Ev}$. The set of the initial states is defined as $\widehat{\iota} = \{s^0\}$, where $s^0 = (away_1, 0), \ldots, (away_n, 0), (0, 0), (\cdot)$.

Moreover, we assume the following set of propositional variables: $\mathcal{AP} = \{tunnel_1, \ldots, tunnel_n\}$ with the following definition of local valuation functions for $i \in \{1, \ldots, n\}$: $\widehat{\mathcal{V}}_{T_i}(tunnel_i) = \{tunnel_i\}$.

Let $Act = \prod_{i=1}^{n} Act_{T_i} \times Act_C \times Act_{Ev}$, with $Act_C = \{start_1, \ldots, start_n, approach_1, \ldots, approach_n, in_1, \ldots, in_n, out_1, \ldots, out_n\}$, $Act_{T_i} = \{start_1, \ldots, start_n, approach_1, \ldots, approach_n, in_1, \ldots, in_n, out_1, \ldots, out_n\}$, and $Act_{Ev} = \{\epsilon_{Ev}\}$ defines the set of joint actions for the scenario. For $\widetilde{a} \in Act$ let $act_{T_i}(\widetilde{a})$ denotes an action of $Train_i$, $act_C(\widetilde{a})$ denotes an action of Controller, and $act_{Ev}(\widetilde{a})$ denotes an action of environment Ev.

We assume the following local evolution functions for $i \in \{1, \ldots, n\}$: $t_{T_i}(away_i, \cdot, true, \{x_i\}, \widetilde{a}) = try_i$, if $act_{T_i}(\widetilde{a}) = start_i$ and $act_C(\widetilde{a}) = start_i$; $t_{T_i}(try_i, \cdot, \Delta > x_i, \{x_i\}, \widetilde{a}) = wait_i$, if $act_{T_i}(\widetilde{a}) = approach_i$ and $act_C(\widetilde{a}) = approach_i$; $t_{T_i}(wait_i, \cdot, x_i > 6, \{\emptyset\}, \widetilde{a}) = tunnel_i$, if $act_{T_i}(\widetilde{a}) = in_i$ and $act_C(\widetilde{a}) = in_i$; $t_{T_i}(tunnel_i, \cdot, true, \{\emptyset\}, \widetilde{a}) = away_i$, if $act_{T_i}(\widetilde{a}) = out_i$ and $act_C(\widetilde{a}) = out_i$.

The specifications we consider are as follows:

- $\varphi_4 = \mathbf{EF}_{[0,\infty)}(\overline{\mathrm{K}}_{Train_1}(InTunnel_1 \wedge \mathbf{EG}_{[2,\infty)}(\neg InTunnel_1)))$ – *states that it is not true that it is always the case that agent Train 1 knows that whenever he is in the tunnel, it will be in the tunnel once again within a bounded period of steps, i.e., within n steps for* $n \geq 2$.
- $\varphi_5 = \mathbf{EF}_{[0,\infty)}(InTunnel_1 \wedge \overline{\mathrm{K}}_{Train_1}(\mathbf{EG}_{[1,n+2)}(\bigwedge_{i=1}^{n}(\neg InTunnel_i))))$ – *expresses that it is not true that it is always the case that if Train 1 is in the tunnel, then he knows that either he or other train will be in the tunnel during the next* $n+1$ *steps.*

All the above formulae are true in the model for TTCS.

4.1 Performance Evaluation

We have performed our experiments on a computer equipped with I7-3770 processor, 32 GB of RAM, and the operating system Linux with the kernel 4.5.1. Our SMT-based and SAT-based BMC algorithms are implemented as standalone programs written in the programming language `C++`. We used the state of the art SMT-solver Z3 [16] (https://github.com/Z3Prover). All the benchmarks together with instructions on how to reproduce our experimental results can be found at the web page http://tinyurl.com/smt4tis-rtectlk.

TGPP. The number of considered k-paths for the formula φ_1 is equal to 10 for $n = 1$ and $4 \cdot (n+1)$ for $n > 1$; for the formula φ_2 is equal to 6 for $n = 1$ and $4 \cdot n$ for $n > 1$; for the formula φ_3 is equal to 6 for $n = 1$, $3 \cdot n$, if n is an even number, and $3 \cdot n$ if n is an odd number.

From Fig. 3 one can observe that the SMT-BMC is able to verify the formula φ_1 for TGPP with 20 nodes, the formula φ_2 for TGPP with 20 nodes, and the formula φ_3 for TGPP with 8 nodes,

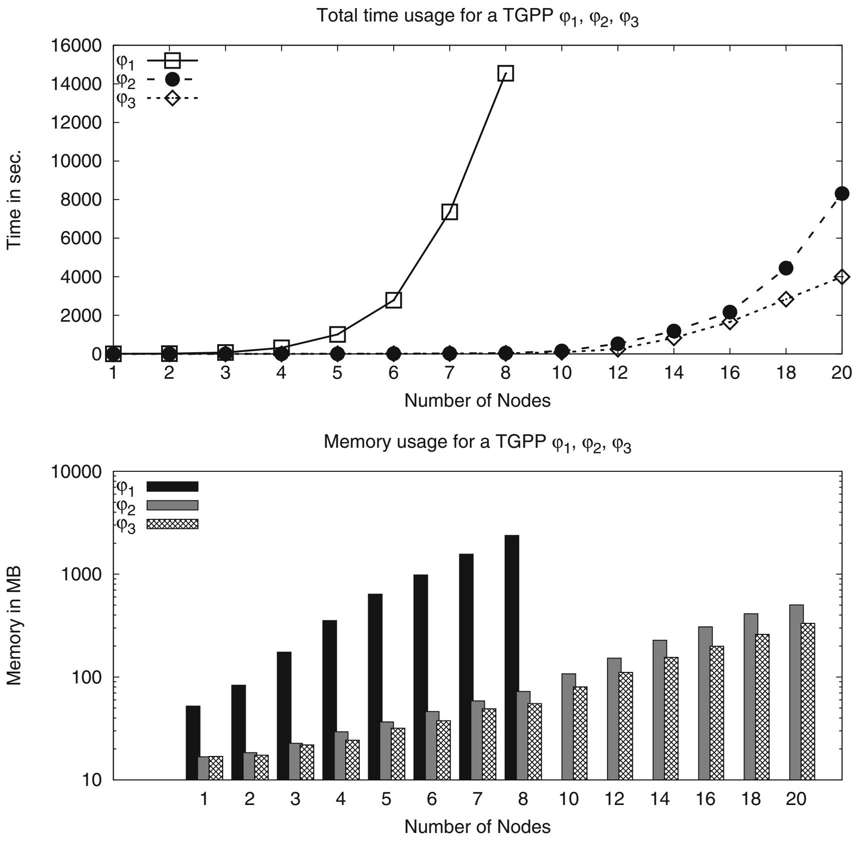


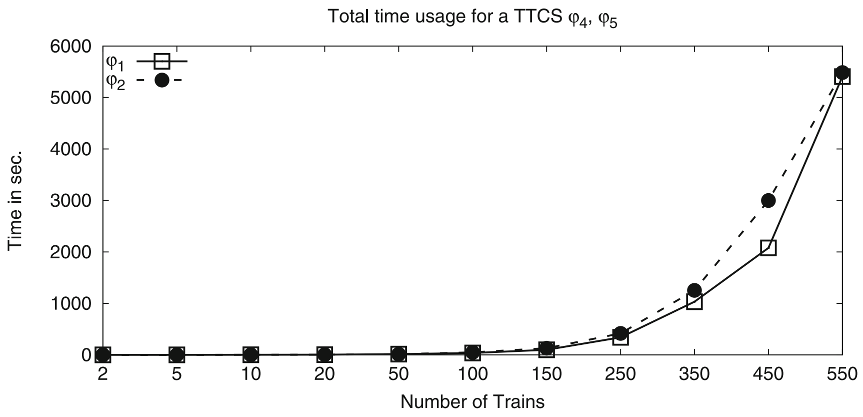
Fig. 3. TGPP with n nodes.

TTCS. The number of considered k-paths for both the formulae is equal to 7.

From Fig. 4 one can observe that the SMT-BMC is able to verify the formulae φ_4 and the formulae φ_5 for TTCS with 550 trains.

4.2 Performance Evaluation Summary

The experimental results show that the SMT-BMC is sensitive to scaling up the size of the benchmarks. As one can see from the line charts in Figs. 3 and 4 showing the total time and the memory consumption for all the tested properties, the experimental results confirm that our new SMT-based BMC for $\mathbb{TIS}$ and for RTECTLK is promising.



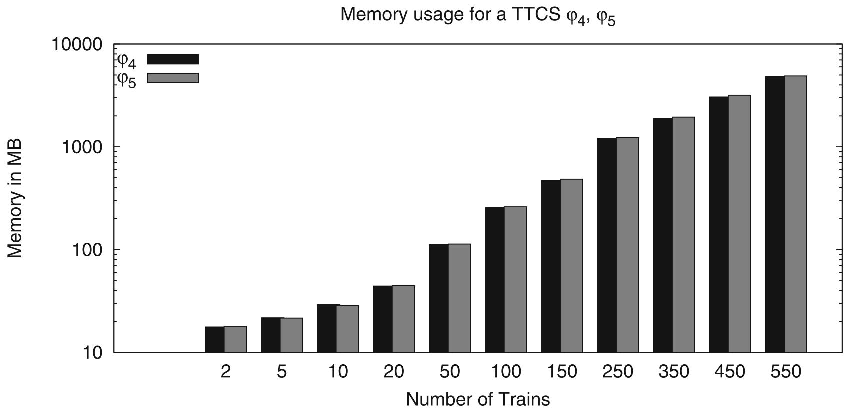


Fig. 4. TTCS with n trains.

5 Conclusions

We have proposed the SMT-based BMC verification method for model checking RTECTLK properties interpreted over the timed interpreted systems. We have provided a preliminary experimental results. The experimental results show that the SMT-based BMC method is worth of interest. We would like to use other SMT-solvers in our implementations and compare experimental results. The BMC for RTECTLK and for $\mathbb{TIS}$s may also be performed by means of Ordered Binary Diagrams (OBDD) and SAT. This will be explored in the future.

The module will be added to the model checker VerICS [11].

References

1. Baier, C., Katoen, J.-P.: Principles of Model Checking. MIT Press, Cambridge (2008)
2. Barrett, C., Sebastiani, R., Seshia, S., Tinelli, C.: Satisfiability modulo theories (chap. 26). In: Biere, A., Heule, M.J.H., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 825–885. IOS Press, Amsterdam (2009)
3. Blackburn, P., de Rijke, M., Venema, Y.: Modal logic. In: Cambridge Tracts in Theoretical Computer Science, vol. 53. Cambridge University Press (2001)
4. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge (1999)
5. Clarke, E., Kroning, D., Ouaknine, J., Strichman, O.: Completeness and complexity of bounded model checking. In: Steffen, B., Levi, G. (eds.) VMCAI 2004. LNCS, vol. 2937, pp. 85–96. Springer, Heidelberg (2004)
6. Emerson, E.A.: Temporal and modal logic (chap. 16). In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science, vol. B, pp. 996–1071. Elsevier Science Publishers, Amsterdam (1990)
7. Emerson, E.A., Mok, A.K., Sistla, A.P., Srinivasan, J.: Quantitative temporal reasoning. Real-Time Syst. **4**(4), 331–352 (1992)
8. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning About Knowledge. MIT Press, Cambridge (1995)
9. Gammie, P., van der Meyden, R.: MCK: model checking the logic of knowledge. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 479–483. Springer, Heidelberg (2004)
10. Jones, A.V., Lomuscio, A.: Distributed BDD-based BMC for the verification of multi-agent systems. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 675–682. IFAAMAS (2010)
11. Kacprzak, M., Nabialek, W., Niewiadomski, A., Penczek, W., Pólrola, A., Szreter, M., Woźna, B., Zbrzezny, A.: VerICS 2007 - a model checker for knowledge and real-time. Fundamenta Informaticae **85**(1–4), 313–328 (2008)
12. Levesque, H.: A Logic of implicit and explicit belief. In: Proceedings of the 6th National Conference of the AAAI, pp. 198–202. Morgan Kaufman, Palo Alto (1984)
13. Lomuscio, A., Qu, H., Raimondi, F.: MCMAS: a model checker for the verification of multi-agent systems. In: Bouajjani, A., Maler, O. (eds.) CAV 2009. LNCS, vol. 5643, pp. 682–688. Springer, Heidelberg (2009)
14. Lomuscio, A., Sergot, M.: Deontic interpreted systems. Stud. Logica. **75**(1), 63–92 (2003)
15. Męski, A., Penczek, W., Szreter, M., Woźna-Szcześniak, B., Zbrzezny, A.: BDD-versus SAT-based bounded model checking for the existential fragment of linear temporal logic with knowledge: algorithms and their performance. Auton. Agents and Multi-agent Syst. **28**(4), 558–604 (2014)
16. de Moura, L., Bjørner, N.S.: Z3: An efficient SMT solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 337–340. Springer, Heidelberg (2008)
17. Peled, D.: All from one, one for all: on model checking using representatives. In: Courcoubetis, C. (ed.) CAV 1993. LNCS, vol. 697, pp. 409–423. Springer, Heidelberg (1993)

18. Penczek, W., Lomuscio, A.: Verifying epistemic properties of multi-agent systems via bounded model checking. Fundamenta Informaticae **55**(2), 167–185 (2003)
19. Wooldridge, M.: An Introduction to Multi-agent Systems, 2nd edn. Wiley, Hoboken (2009)
20. Woźna-Szcześniak, B.: SAT-based bounded model checking for weighted deontic interpreted systems. In: Correia, L., Reis, L.P., Cascalho, J. (eds.) EPIA 2013. LNCS, vol. 8154, pp. 444–455. Springer, Heidelberg (2013)
21. Woźna-Szcześniak, B., Zbrzezny, A., Zbrzezny, A.: The BMC method for the existential part of RTCTLK and interleaved interpreted systems. In: Antunes, L., Pinto, H.S. (eds.) EPIA 2011. LNCS, vol. 7026, pp. 551–565. Springer, Heidelberg (2011)
22. Woźna-Szcześniak, B., Zbrzezny, A.: Checking EMTLK properties of timed interpreted systems via bounded model checking. Studia Logica, 1–38 (2015)
23. Zbrzezny, A.: Improving the translation from ECTL to SAT. Fundamenta Informaticae **85**(1–4), 513–531 (2008)