Timed Obstruction Logic: A Timed Approach to Dynamic Game Reasoning

Jean Leneutre
Télécom Paris, Institut Polytechnique de Paris
Palaiseau, France
jean.leneutre@telecom-paris.fr

Vadim Malvone
Télécom Paris, Institut Polytechnique de Paris
Palaiseau, France
vadim.malvone@telecom-paris.fr

James Ortiz
Télécom Paris, Institut Polytechnique de Paris
Palaiseau, France
james.ortizvega@telecom-paris.fr

ABSTRACT

Real-time cybersecurity and privacy applications require reliable verification methods and system design tools to ensure their correctness. Recently, a growing literature has recognized Timed Game Theory as a sound theoretical foundation for modeling strategic interactions between attackers and defenders. This paper proposes Timed Obstruction Logic (TOL), a formalism for verifying specific timed games with real-time objectives unfolding in dynamic models. These timed games involve players whose discrete and continuous actions can impact the underlying timed game model. We show that TOL can be used to describe important timed properties of real-time cybersecurity games. Finally, we provide a verification procedure for TOL and show that its complexity is PSPACE-complete, meaning that it is not higher than that of classical timed temporal logics like TCTL. Thus, we increase the expressiveness of properties without incurring any additional complexity.

CCS CONCEPTS

• **Software and its engineering** → **Software testing and debugging**; • **Theory of computation** → *Verification by model checking*.

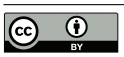
KEYWORDS

Dynamic Models; Temporal Logic; Model Checking

ACM Reference Format:
Jean Leneutre, Vadim Malvone, and James Ortiz. 2025. Timed Obstruction Logic: A Timed Approach to Dynamic Game Reasoning. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 10 pages.

1 INTRODUCTION

Multi-agent systems (MAS) are composed of interacting autonomous agents [3, 22, 31, 36, 39, 42] and have been successfully applied in cybersecurity and distributed systems. However, modeling security and heterogeneous distributed systems is inherently error-prone. Thus, computer scientists typically address the issue of verifying that a system actually behaves as intended, especially for complex systems. Some techniques have been developed to accomplish this task: testing is the most common, but in many cases, a formal proof of correctness is required. Formal verification techniques include theorem proving and model checking [21]. Model checking, in particular, has been successfully applied to formally verify security and distributed systems, including hardware components and communication and security protocols. Unlike traditional distributed systems, formal verification techniques for Real-Time MAS (RT-MAS) [15, 16] are still in their infancy due to the more sophisticated nature of the agents, their autonomy, their real-time constraints, and the richness of the formalisms used to specify properties [20, 35]. RT-MAS combines game theory techniques with real-time behavior in a distributed environment [5, 15, 16]. Such real-time behavior can be verified using real-time modal logic [6, 45]. The development of methods and techniques with integrated features from both research areas undoubtedly leads to an increase in complexity and the need to adapt current techniques or, in some cases, to develop new formalisms, techniques, and tools [40, 47, 48]. Developing these formalisms correctly requires algorithms, procedures, and tools to produce reliable end results [10, 51]. Agents in RT-MAS are considered to be players in games played over real-time models (such as Timed Automata (TA) [2] and Timed Petri Nets [33]), and their goals are specified by real-time logic formulas [6, 33, 40, 44, 47]. For example, the fact that a coalition of players has a strategy to achieve a certain goal by acting cooperatively can be expressed using the syntax of logics such as Timed Alternating-time Temporal Logic (TATL) [30, 37] and Strategic Timed Computation Tree Logic (STCTL) [6]. However, STCTL with continuous semantics is more expressive than TATL, as shown in [6]. Moreover, in [6], was shown that the model checking problem for STCTL with continuous semantics and memoryless perfect information is of the same complexity as for TCTL, while for STCTL with continuous semantics and perfect recall it is undecidable. Model checking for TATL with continuous semantics is undecidable [6]. In all previous logics, the timed game model in which the players are playing is treated as a static game model, i.e., the actions of the players affect their position within the model, but do not affect the structure of the model itself. In this paper, we propose Timed Obstruction Logic (TOL), for reasoning about RT-MAS with real-time goals [6, 33, 40, 44, 47] played in a dynamic model. Dynamic game models [17, 43, 51] have been studied in a variety of contexts, including cybersecurity and planning. In our new logic (TOL), games are played over an extended TA (Weighted TA (WTA) [4]) by two players (Adversary and Demon). There is a cost ($W(e)$) associated with each edge of the automaton. This means that, given a location l of the automaton and a natural number n, the Demon deactivates an appropriate subset T of the set of edges incident to l such that the sum of the deactivation costs of the edges contained in T is less than n. Then, the Adversary selects a location l' such that l is adjacent to l' and the edge from l to l'



Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

does not belong to the set of edges selected by the Demon in the previous round. The edges deactivated in the previous round are restored, and a new round starts at the last node selected by the opponent. The Demon wins the timed game if the infinite sequence of nodes subsequently selected by the opponent satisfies a certain property φ expressed by a TOL formula. Furthermore, in addition to the introduction of our new logic and its adaptation to the specification of properties in the context of real-time cybersecurity, we provide a verification procedure for TOL and show that its complexity is PSPACE-complete, i.e., not higher than that of classical timed temporal logics such as TCTL. Thus, we increase the expressiveness of properties without incurring a cost in terms of complexity.

Structure of the work. Theoretical background is presented in Section 2. In Section 3, we present the syntax and the semantics of our new logic, called Timed Obstruction Logic (TOL). In Section 4, we show our model checking algorithm and prove that the model checking problem for TOL is PSPACE-Complete. In Section 5, we compare TOL with other timed and modal logics. In Section 6, we present our case study in the cybersecurity context. In Section 7, we compare our approach to related work. Finally, Section 8 concludes and presents possible future directions.

2 BACKGROUND

Let $\mathbb{N}$ be the set of natural numbers, $\mathbb{N}_{\geq 0}$ the set including 0, $\mathbb{R}_{\geq 0}$ the set of non-negative reals, and $\mathbb{Z}$ the set of integers. For sets X and Y, $|X|$ denote the cardinality of X. The set operations of intersection, union, complementation, set difference, Cartesian product and empty set are $X \cap Y$, $X \cup Y$, $\overline{X}$, $X \setminus Y$, $X \times Y$, and $\emptyset$ respectively. Inclusion and strict inclusion are denoted $X \subseteq Y$ and $X \subset Y$. Let $\pi = \pi_1, \ldots, \pi_n$ be a countable sequence, we denote by π_i its i-th element, by $\pi_{\leq_i}$ the finite prefix $\pi_1, \ldots, \pi_i$ of π and by $\pi_{\geq i}$ the (possibly infinite) suffix of π starting at π_i. If π is a finite sequence, $last(\pi)$ denotes the last element π_n of π.

2.1 Attack Graphs

An attack is an attempt by an attacker to gain unauthorized access to resources or compromise system integrity. In this context, the Attack Graph (AG) [17, 32] is a widely used model for representing interactions between an attacker and a defender employing Moving Target Defense (MTD) mechanisms [19]. MTD techniques, such as Address Space Layout Randomization (ASLR) [41], dynamically reconfigure system components to modify the attack surface, thereby reducing the attacker's success probability. However, activating an MTD countermeasure incurs performance costs, as system services may become partially or completely unavailable during reconfiguration. Therefore, optimizing MTD deployment is essential to balance security risk reduction and system performance.

2.2 Weighted Transition Systems

Weighted Transition Systems (WTS) are an extension of Labeled Transition Systems (LTS) [46]. They introduce operational semantics in reactive systems. A AG can be defined as a WTS.

Definition 1 (Weighted Transition Systems (WTS)). *Let* AP *be a finite set of atomic propositions (atoms). A WTS is a tuple* $\mathcal{M} = (S, s_0, \Sigma, E, W, K, F)$ *where:*

- S *is a finite set of states,*
- $s_0 \in S$ *is an initial state,*
- Σ *is a finite set of actions,*
- $E \subseteq S \times \Sigma \times S$ *is a transition relation,*
- $W : E \to \mathbb{N}_{\geq 0}$ *is a function that labels the elements of* E*,*
- $K : S \to 2^{\mathsf{AP}}$ *is a labeling function for the states,*
- $F \subseteq S$ *is a set of goal states.*

The transitions of a WTS are noted in the following way: we write $s \xrightarrow[w]{a} s'$ whenever $a \in \Sigma$, $(s, a, s') \in E$ and $W(s, a, s') = w$ where $w \in \mathbb{N}_{\geq 0}$. We use non-negative real-valued variables called as *clocks* to represent the continuous time domain. Clocks advance synchronously at a uniform rate and are the basis of TA [2]. Here, we work with an extension of TA known as Weighted TA (WTA) [4].

2.3 Weighted Timed Automata

We now explore the relation between WTS and Weighted Timed Automata (WTA) [4]. A WTA is an extension of a TA [2] with weight/cost information at both locations and edges, and it can be used to address several interesting questions [4, 12].

Definition 2 (Clock constraints and invariants). *Let* X *be a finite set of variables ranging over* $\mathbb{R}_{\geq 0}$*, called clocks. The set* $\Phi^+(X)$ *of clock constraints over the set of clocks* X *is given by the following grammar:*

$$\phi := true \mid x \sim c \mid x - y \sim c \mid \phi_1 \ \wedge \ \phi_2$$

where $x, y \in X$*,* $c \in \mathbb{N}$*, and* $\sim \in \ \{<, >, \leq, \geq, =\}$.

The clock constraints of the form $true$, $x \sim c$ are called non-diagonal constraints and those of the form $x - y \sim c$ are called diagonal constraints. The set of non-diagonal constraints over X is denoted by $\Phi(X)$. In this paper, we use the non-diagonal constraints as in [2] where the comparison between two clocks is not allowed [13][1]. Clock invariants $\Delta(X)$ are clock constraints in which $\sim \in \{<, \leq\}$.

Definition 3 (Clock valuations). *Given a finite set of clocks* X*, a clock valuation function,* $v : X \to \mathbb{R}_{\geq 0}$ *assigning to each clock* $x \in X$ *a non-negative value* $v(x)$*. We denote* $\mathbb{R}^X_{\geq 0}$ *the set of all valuations. For a clock valuation* $v \in \mathbb{R}^X_{\geq 0}$ *and a time value* $d \in \mathbb{R}_{\geq 0}$*,* $v + d$ *is the valuation satisfied by* $(v + d)(x) = v(x) + d$ *for each* $x \in X$*. Given a clock subset* $Y \subseteq X$*, we denote* $v[Y \leftarrow 0]$ *the valuation defined as follows:* $v[Y \leftarrow 0](x) = 0$ *if* $x \in Y$ *and* $v[Y \leftarrow 0](x) = v(x)$ *otherwise.*

Here, we only consider the weight/cost in the edges (transitions) in our WTA. Formally, a WTA is defined as follows [4].

Definition 4 (Weighted Timed Automata (WTA)). *Let* X *be a finite set of clocks and* AP *a finite set of atoms. A WTA is a tuple* $\mathcal{A} = (L, l_0, X, \Sigma, T, I, W, K, F)$*, where:*

- L *is a finite set of locations,*
- $l_0 \in L$ *is an initial location,*
- X *is a finite set of clocks,*
- Σ *is a finite set of actions,*
- $T \subseteq L \times \Sigma \times \Phi(X) \times 2^X \times L$ *is a finite set of transitions,*
- $I : L \to \Delta(X)$ *is a function that associates to each location a clock invariant,*

[1] Here, we use this kind of clocks constraints to ensure the correctness of the construction of our symbolic representation of WTA

- $W : T \to \mathbb{N}_{\geq 0}$ *is a function that labels the elements of* T,
- $K : L \to 2^{\mathsf{AP}}$ *is a labeling function for the locations,*
- $F \subseteq L$ *is a set of goal locations.*

We write $l \xrightarrow[w]{a,\phi,Y} l'$ instead of $(l, a, \phi, Y, l')_w \in T$ for an edge from l to l' with guard $\phi \in \Phi(X)$, reset set $Y \subseteq X$ and $w \in \mathbb{N}_{\geq 0}$. The value $W(t)$ given to edge $t = (l, a, \phi, Y, l')_w$ where $t \in T$ represents the cost of taking that edge. The value $W(t)$ given to edge t represents the deactivation cost. Since cost information cannot be employed as constraints on edges, the undecidability of Hybrid Automata (HA) [28] is avoided in the case of WTA [12] (i.e., decidability results are preserved for WTA). In WTA, costs are explicitly defined in its syntax, however, they do not influence the discrete behavior of the system. Since there is no cost constraint, the semantics of a WTA is similar to that of a TA. It is thus given as a WTS.

Definition 5 (Semantics of WTA). *Let* $\mathcal{A} = (L, l_0, X, \Sigma, T, I, W, K, F)$ *be a WTA. The semantics of WTA* $\mathcal{A}$ *is given by a WTS*$(\mathcal{A}) = (S, s_0, \Sigma_\Delta, E, W', K', S_F)$ *where:*

- $S \subseteq L \times \mathbb{R}^X_{\geq 0}$ *is a set of states,*
- $s_0 = (l_0, v_0)$ *with* $v_0(x) = 0$ *for all* $x \in X$ *and* $v_0 \models I(l_0)$,
- $\Sigma_\Delta = \Sigma \uplus \mathbb{R}_{\geq 0}$,
- $E \subseteq S \times \Sigma_\Delta \times S$ *is a transition relation defined by the following two rules:*
  - ***Discrete transition:*** $(l, v) \xrightarrow[w]{a} (l', v')$ *for* $a \in \Sigma$ *and* $w \in \mathbb{N}_{\geq 0}$ *iff* $l \xrightarrow[w]{a,\phi,Y} l'$, $v \models \phi$, $v' = v[Y \leftarrow 0]$ *and* $v' \models I(l')$ *and,*
  - ***Delay transition:*** $(l, v) \xrightarrow{d} (l, v + d)$, *for some* $d \in \mathbb{R}_{\geq 0}$ *iff* $v + d \models I(l)$.
- $W' = W$,
- $K'((l, v)) = K(l) \cup \{\phi \in \Phi(X) \mid v \models \phi\}$,
- $S_F \subseteq F \times \mathbb{R}^X_{\geq 0}$ *is a set of states,*

2.4 Paths and n-strategy

A path ρ in WTS$(\mathcal{A})$ is an infinite sequence of consecutive delays and discrete transitions. A finite path fragment of $\mathcal{A}$ is a run in WTS$(\mathcal{A})$ starting from the initial state $s_0 = (l_0, v_0)$, with delay and discrete transitions alternating along the path: $\rho = s_0 \xrightarrow{d_0} s'_1 \xrightarrow[w_0]{a_0} s_1 \xrightarrow{d_1} s'_2 \xrightarrow[w_1]{a_1} s_2 \ldots s_{n-1} \xrightarrow{d_{n-1}} s'_n \xrightarrow[w_n]{a_n} s_n \ldots$ or more compactly $s_0 \xrightarrow[w_0]{d_0,a_0} s_1 \xrightarrow[w_1]{d_1,a_1} s_2 \xrightarrow[w_2]{d_2,a_2} s_3 \ldots s_{n-1} \xrightarrow[w_{n-1}]{d_{n-1},a_{n-1}} s_n \ldots$, where $v_0(x) = 0$ for every $x \in X$. A path of WTS$(\mathcal{A})$ is *initial* if $s_0 = (l_0, v_0) \in S$, where $l_0 \in L$, v_0 assigns 0 to each clock, and *maximal* if it ends in a goal location. We write ρ_i to denote the i-th element $s_i = (l_i, v_i)$ of ρ, $\rho_{\leq i}$ to denote the prefix $s_0, \ldots, s_i$ of ρ and $\rho_{\geq i}$ to denote the suffix $s_i, s_{i+1} \ldots$ of ρ. A *history* is any finite prefix of some path. We use H to denote the set of histories. Due to the infinite nature of WTSs, i.e. their continuous transitions, given a WTA $\mathcal{A}$ we will use here ind(T') to indicate the set of deactivated edges $T' \subset T$ in $\mathcal{A}$ induced in the WTS$(\mathcal{A})$.

Definition 6. *Let* $\mathcal{A}$ *be a WTA and* n *be a natural number. Given a model WTS*$(\mathcal{A})$*, a* n*-strategy is a function* $\mathfrak{S} : H \to 2^T$ *such that, given a history* h*, returns a subset* T' *such that: (i)* $T' \subset T(last(h))$*, (ii)* ind$(T') \subset E(last(h))$*, (iii)* $(\sum_{t \in T'} W(t)) \leq n$*. A memoryless* n*-strategy is a* n*-strategy* $\mathfrak{S}$ *such that for all histories* h *and* h' *if* $last(h) = last(h')$ *then* $\mathfrak{S}(h) = \mathfrak{S}(h')$.

A path ρ is compatible with a n-strategy if for all $i \geq 1$, $(\rho_i, \sigma, \rho_{i+1}) \notin \mathfrak{S}(\rho_{\leq i})$, where $\sigma \in \Sigma$. Given a state $s = (l, v)$ and a n-strategy $\mathfrak{S}$, $Out(s, \mathfrak{S})$ refers to the set of pathways starting from s that are consistent with $\mathfrak{S}$.Definition 6 represents the case where the demon's strategy can deactivate edges.

2.5 Predecessor operator and Zone Graph

Since the number of states in a WTA is infinite, it is impossible to build a finite state automaton. Thus, a symbolic semantics called zone graph was proposed for a finite representation of TA behaviors [11]. The zone graph representation of TA is not only an important implementation approach employed by most contemporary TA tools [11], but it also provides a theoretical foundation for demonstrating the decidability of semantic properties for a given TA. In a zone graph, clock zones represent sets of clock valuations symbolically. A clock zone $Z \in \mathbb{R}^X_{\geq 0}$ over a set of clocks X is a set of valuations that satisfy a conjunction of constraints. Formally, the clock zone for the constraint ϕ is $Z = \{v \mid v(x) \models \phi, x \in X\}$. Geometrically, a zone is a convex polyhedron. A symbolic state (or zone) is a pair $\mathcal{Z} = (l, Z)$, where l is a location and Z is a clock zone. A zone $\mathcal{Z} = (l, Z)$ represents all the states $z = (l', v) \in \mathcal{Z}$ if $l = l'$ and $v \in Z$, indicating that a state is contained in a zone. We can now define the symbolic discrete and delay predecessor operations on zones as follows:

Definition 7 (Discrete and Time Predecessor). *Let* $\mathcal{Z}$ *be a zone and* e *be an edge of a WTS*$(\mathcal{A})$*, then:*

$$\textit{disc-pred}(e, \mathcal{Z}) = \{z \mid \exists z' \in \mathcal{Z}, z \xrightarrow[w]{e} z'\}$$

$$\textit{time-pred}(\mathcal{Z}) = \{z \mid \exists z' \in \mathcal{Z}, z \xrightarrow[w]{d} z' \text{ and } d \in \mathbb{R}_{\geq 0}\}$$

That is, disc-pred$(t, \mathcal{Z})$ is the set of all e-predecessors of states in $\mathcal{Z}$ and time-pred$(\mathcal{Z})$ is the set of all time-predecessors of states in $\mathcal{Z}$. According to these definitions, if $\mathcal{Z}$ is a zone then time-pred$(\mathcal{Z})$ and disc-pred$(e, \mathcal{Z})$ are also zones, meaning that zones are preserved by the above predecessor operations and $w \in \mathbb{N}_{\geq 0}$.

Definition 8 (Predecessor). *Let* $\mathcal{Z}$ *be a zone and* e *be a edge of a WTS*$(\mathcal{A})$*, then :*

$$\textit{pred}(e, \mathcal{Z}) = \textit{disc-pred}(e, \textit{time-pred}(\mathcal{Z}))$$

That is, pred is the set of all states that can reach some state in $\mathcal{Z}$ by performing a e discrete transition and allowing some time (delay transition) to pass.

Definition 9 (Zone Graph). *Given a WTA* $\mathcal{A}$*, a zone graph is a transition system* $ZG(\mathcal{A}) = (Q, q_0, (\Sigma \cup \{\epsilon\}), \rightarrow_{ZG}, K_Z)$*, where:*

- Q *consists of pairs* $q = (l, Z)$ *where* $l \in L$*, and* $Z \in \Phi^+(X)$ *is a non-empty clock zone with* $Z \subseteq I(l)$,
- $q_0 = (l_0, Z_0) \in Q$ *is the initial zone with* $Z_0 = [\![\bigwedge_{x \in X} x = 0]\!]$,
- Σ *is the set of labels of* $\mathcal{A}$,
- $\rightarrow_{ZG} \subseteq Q \times (T \cup \{\epsilon\}) \times Q$ *is a set of transitions, where each transition in ZG*$(\mathcal{A})$ *is a labeled by a transition* $e = (l, a, \phi, Y, l')_w \in T$*, where* T *is in* $\mathcal{A}$ *(and* $\Sigma = T$*) and* ϵ *is a symbolic delay transition. For each* $e \in (\Sigma \cup \{\epsilon\})$*, transitions are defined as:*

– *For every e in $\mathcal{A}$ and zone q in Q, there exists a discrete transition (q, e, q'), where $q = (l, Z) \xrightarrow{e}_{ZG} q' = (l', post(e, (l, Z)))$ if $post(e, (l, Z)) \neq \emptyset$,*
– *For a clock zone Z, there exists a delay transition (q, ϵ, q'), where $q = (l, Z) \xrightarrow{\epsilon}_{ZG} q' = (l, Z')$ and $Z' = time\text{-}succ(l, Z) \cap I(l)$ where Z' is a time successor of Z.*

- $K_z(l, Z) = K(l) \cup \{\phi \in \Phi^+(X) \mid \nu \models \phi\}$

Every zone has a transitive ϵ delay transition to itself. An ϵ delay transition must be strict to maintain clock zones and is not reflexive between zones. Since an ϵ transition is reflexive, the time successor relation is also reflexive. The operations time-succ(l, Z) and post$(e, (l, Z)$ present the set of time successors of any state in (l, Z) and the successor of (l, Z) by the transition e, respectively (remember that $\mathcal{Z} = (l, Z)$).

3 TIMED OBSTRUCTION LOGIC

In this section, we define the syntax and semantics of our Timed Obstruction Logic (TOL). Our definitions are based on [17, 18].

Definition 10. *Let $\mathcal{A}$ be a WTA,* AP *a set of atomic propositions (or atoms), a set X of clocks of $\mathcal{A}$ and J a non-empty set of clocks of the formula, where $X \cap J = \emptyset$. Formulas of Timed Obstruction Logic (TOL) are defined by the following grammar:*

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \phi \mid \langle\!\!\downarrow_n\rangle(\varphi_1 \,\mathsf{U}\, \varphi_2) \mid \langle\!\!\downarrow_n\rangle(\varphi_1 \,\mathsf{R}\, \varphi_2) \mid j.\varphi$$

where $p \in$ AP is an atomic formula, $j \in J$, $n \in \mathbb{N}_{\geq 0}$ represents the grade *of the strategic operator, and $\phi \in \Phi(X \cup J)$.*

It is possible to compare a formula clock and an automata clock, for example, by using the clock constraint ϕ, which applies to both formula clocks and clocks of the TA. The boolean connectives $\bot$, $\vee$ and, $\rightarrow$ can be defined as usual. Clock j in $j.\varphi$ is called a freeze identifier and bounds the formula clock j in φ. The interpretation is that $j.\varphi$ is valid in a state s if φ holds in s where clock j starts with value 0 in s. This freeze identifier can be used in conjunction with temporal constructs to indicate common timeliness requirements such as punctuality, bounded response, and so on. As OL, we define $\langle\!\!\downarrow_n\rangle\mathsf{F}\,\varphi := \langle\!\!\downarrow_n\rangle(\top \,\mathsf{U}\, \varphi)$, $\langle\!\!\downarrow_n\rangle\mathsf{G}\,\varphi := \langle\!\!\downarrow_n\rangle(\bot \,\mathsf{R}\, \varphi)$ and $\langle\!\!\downarrow_n\rangle(\varphi \,\mathsf{W}\, \psi) := \langle\!\!\downarrow_n\rangle(\psi \,\mathsf{R}\, (\varphi \vee \psi))$. The size $|\varphi|$ of a formula φ is the number of its connectives. With the help of the freeze identifier operator of TOL, a time constraint can be added concisely. For instance, the formula $j.\langle\!\!\downarrow_n\rangle((\varphi_1 \wedge j \leq 7) \,\mathsf{U}\, \varphi_2)$ intuitively means that there is a demonic strategy such that all paths that are compatible with the strategy, the property φ_1 holds continuously until within 7 time units φ_2 becomes valid. From the above formula, it is clear that timing constraints are allowed. In this case, we will call the formulas with timing constraints, such as timed temporal formulas. The intuitive meaning of a formula $\langle\!\downarrow\rangle\varphi$ with φ timed temporal formula is: there is a demonic strategy such that all paths of the WTS that are compatible with the strategy satisfy φ. Unlike OL, our logical TOL does not use the next operator. This is because the time domain is continuous and there is no unique next time. The next operator could be added to our logic. However, it would only be used in temporal semantics and our logic is more oriented towards real-time semantics. Formulas of TOL will be interpreted over WTS. We can now precisely define the semantics of TOL formulas.

Definition 11 (TOL Semantics). *Let $\mathcal{A}$ be a WTA, a set X of clocks of $\mathcal{A}$ and J a non-empty set of clocks of the formula, $p \in$ AP, $\phi \in \Phi(X \cup J)$ and $\mathcal{M} = WTS(\mathcal{A})$. An extended state over S is a triple (l, ν, μ), where $s = (l, \nu) \in S$ is a WTS state and μ a valuation for the formula clocks in J. The satisfaction relation between a WTS $\mathcal{M}$, TOL formulas φ and ψ and an extended state $s_\mu = (l, \nu, \mu)$[2] of the formula, is given inductively as follows:*

- $\mathcal{M}, s \models \top$ *for all state s,*
- $\mathcal{M}, s \models p$ *iff* $p \in K(s)$,
- $\mathcal{M}, s \models \neg\varphi$ *iff not* $\mathcal{M}, s \models \varphi$ *(notation $\mathcal{M}, s \not\models \varphi$),*
- $\mathcal{M}, s \models \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{M}, s \models \varphi_1$ *and* $\mathcal{M}, s \models \varphi_2$,
- $\mathcal{M}, s \models \phi$ *iff* $\nu \models \phi$,
- $\mathcal{M}, s \models \langle\!\!\downarrow_n\rangle(\varphi \,\mathsf{U}\, \psi)$ *iff there is a n-strategy $\mathfrak{S}$ such that for all $\rho \in Out(s, \mathfrak{S})$ there is a $j \in \mathbb{N}$ such that $\mathcal{M}, \rho_j \models \psi$ and for all $0 \leq k < j$, $\mathcal{M}, \rho_k \models \varphi$,*
- $\mathcal{M}, s \models \langle\!\!\downarrow_n\rangle(\varphi \,\mathsf{R}\, \psi)$ *iff there is a n-strategy $\mathfrak{S}$ such that for all, $\rho \in Out(s, \mathfrak{S})$ we have that either $\mathcal{M}, \rho_i \models \psi$ for all $i \in \mathbb{N}$ or there is a $k \in \mathbb{N}$ such that $\mathcal{M}, \rho_k \models \varphi$ and $\mathcal{M}, \rho_i \models \psi$ for all $0 \leq i \leq k$,*
- $\mathcal{M}, s \models j.\varphi$ *iff* $\mathcal{M}, (l, \nu[j \leftarrow 0]) \models \varphi$.

Two formulas φ and ψ are semantically equivalent (denoted by $\varphi \equiv \psi$) iff for any model $\mathcal{M}$ and extended state s of $\mathcal{M}$, $\mathcal{M}, s \models \varphi$ iff $\mathcal{M}, s \models \psi$.

The relationship between WTA and WTS is defined as follows.

Definition 12. *Let $\mathcal{A}$ be a WTA and $\varphi \in$ TOL, then $\mathcal{A} \models \varphi$ iff $WTS(\mathcal{A}) \models \varphi$.*

Let φ be a formula, the set of extended states satisfying φ is independent of the valuation μ for the formula clocks. Thus, for any state $s = (l, \nu)$ in a WTS and valuations μ, μ' for the formula clocks, we can get that $\mathcal{M}, (l, \nu, \mu) \models \varphi$ iff $\mathcal{M}, (l, \nu, \mu') \models \varphi$. Therefore, when φ is closed, it makes sense to talk about a state s that satisfies φ.

Let φ be any formula, $(X \cup J)$ a set of clocks (formula and automaton) and $\mathcal{A}$ be a WTA, then Sat(φ) denotes the set of extended states of $\mathcal{M} = \mathsf{WTS}(\mathcal{A})$ verifying, φ, i.e., Sat$(\varphi) = \{s \in S \mid \mathcal{M}, s \models \varphi\}$. Next, we can establish the model checking problem.

Definition 13. *Given $\mathcal{M}$, an extended state s, and φ, the model checking concerns determining whether $\mathcal{M}, s \models \varphi$.*

4 MODEL CHECKING

Here, we present our model checking algorithm for TOL. Furthermore, we show that the model checking problem for TOL is decidable in PSPACE-complete. The general structure of the algorithm shown here is similar to OL model checking algorithm [18]. TOL model checking algorithm is based on the computation of the set Sat(φ) of all states satisfying a TOL formula φ, followed by checking whether the initial state is included in this set. A WTA $\mathcal{A}$ satisfies TOL state formula φ if and only if φ holds in the initial state of WTA: $\mathcal{A} \models \varphi$ iff $l_0 \in L$ such that $(l_0, \nu_0, \mu_0) \in$ Sat(φ), where $\nu_0(x) = 0$ for all $x \in X$ and $\mu_0(x) = 0$ for all $j \in J$. A TOL formula $\langle\!\!\downarrow_n\rangle((\varphi \,\mathsf{U}\, \psi)$ holds in a state s iff $\langle\!\!\downarrow_n\rangle\mathsf{U}(\text{Sat}(\psi_1), \text{Sat}(\psi_2))$ with U being an TOL operator. As mentioned in subsection 2.5, build a WTS$(\mathcal{A})$ for some

[2]To facilitate reading, from this point onward we will use only the symbol s for an extended state.

WTA $\mathcal{A}$ is therefore not practicable. Instead, the basic idea is to construct a zone graph [11], which is built from the WTA $\mathcal{A}$ and the TOL formula φ (i.e., ZG($\mathcal{A}$, φ)). Since ZG($\mathcal{A}$, φ) depends on φ, the definition of K_z (in Definition 9) is modified as follows: $K_z(l, Z)$ = $K(l) \cup \{\phi \in \Phi^+(X) \mid \nu \models \phi\} \cup \{\phi \in \Phi^+_\varphi(J) \mid \nu \models \phi\}$ (here $\Phi^+_\varphi(J)$ denotes the set of clock constraints of φ). In short, Algorithm 1 begins with a WTA $\mathcal{A}$ and a formula φ used to construct the zone graph ZG($\mathcal{A}$, φ) and returns the set of symbolic states of $\mathcal{A}$ satisfying φ. The Algorithm 1 works as follows: it first constructs the zone graph ZG($\mathcal{A}$, φ), then it recursively computes, for all subformulas ψ, the sets of symbolic states Sat(ψ) for which ψ is satisfied. The computation of Sat(ψ) for ψ being true, a proposition p, or a clock constraint ϕ is explicit. The negation and conjunction computations are straightforward. The computation of the TOL formula $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)$ and $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{R}\, \psi_2)$ are defined under the computation of predecessor sets. However, the notion of predecessors is different for the quantifiers in TCTL [29][2]. The computation of the TOL formula $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)$ can be reduced to the computation of an OL formula. The computation of $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)$ is a fixed-point iteration that starts at Sat(ψ_2) and iteratively adds all predecessor symbolic states that are in Sat(ψ_1). We need to define a new predecessor operator to compute all predecessors with the obstruction operator. We will now use our zone graph to compute predecessors. The predecessor computation is done by the operator $\blacktriangledown(n, \mathcal{Z})$ for a symbolic state $\mathcal{Z}$ (zones) and a number n, computes the set of all predecessor symbolic states (likewise, for the operator R).

Definition 14. *Given a symbolic state $\mathcal{Z}$ and e an edge, we define Pred($\mathcal{Z}$) = $\bigcup_{e \in E}$ pred(e, $\mathcal{Z}$).*

It is well known that the union of zones could create non-convex zones. However, the problem related to the union of zones has been addressed in different kinds of literature [27, 49, 50], yielding excellent results and algorithms for generating convex zones. We could use the algorithm presented in [49] to obtain convex zones again. Now, the obstruction predecessor of a zone $\mathcal{Z}$, denoted $\blacktriangledown(n, \mathcal{Z})$, is defined as the set of symbolic states that characterizes all predecessors of the symbolic state $\mathcal{Z}$, where each state z satisfies $\blacktriangleright(z, n, \overline{\mathcal{Z}})$, that is, it can transition to a state not in the set $\mathcal{Z}$ where the sum of all successors of z is less than or equal to n.

Definition 15. *Let $n \in \mathbb{N}$ and $\mathcal{Z}$ a symbolic state, we write:*

$$\blacktriangleright(z, n, \mathcal{Z}) = \Big(\sum_{z' \in \mathcal{Z} \wedge \sigma \in \Sigma_\Delta} \mathsf{W}(z, \sigma, z')\Big) \leq n$$

$$\blacktriangledown(n, \mathcal{Z}) = \{z \in \mathit{Pred}(\mathcal{Z}) \mid \blacktriangleright(z, n, \overline{\mathcal{Z}})\}$$

Proposition 1. *Let z a state, n a natural number, and $\mathcal{Z}$ a symbolic state (or zone), then $z \in \blacktriangledown(n, \mathcal{Z})$ iff $z \in \mathcal{Z}$.*

Proof. (Sketch) A proof of this proposition may be obtained from TA [2]. □

For the U and R operators, auxiliary methods are defined. These methods are listed in Algorithm 2 and Algorithm 3. Algorithm 2 shows the backward search for computing the method $\langle\!\dashv_n\rangle$U (Sat(ψ_1), Sat(ψ_2)) in line 15 of Algorithm 1. Algorithm 3 shows the backward search for computing the method $\langle\!\dashv_n\rangle$R(Sat(ψ), Sat(ψ_2)) in line 17 of Algorithm 1. Termination of the Algorithm 1 intuitively follows, as the number of states in the zone graph is finite. The following proposition establishes the termination and the correctness of our model checking algorithm.

Algorithm 1 TOL model checking
Input: A model $\mathcal{M}$ = ZG($\mathcal{A}$, φ) where $\mathcal{A}$ is a WTA and φ is a TOL formula
Output: Sat(φ) ← $\{q \in Q \mid \mathcal{M}, q \models \varphi\}$

```
1:  for all i ≤ |φ| do
2:      for all ψ ∈ Sub(φ) with |ψ| = i do
3:          switch (ψ) do
4:              case ψ = ⊤
5:                  Sat(ψ) ← Q
6:              case ψ = p
7:                  Sat(ψ) ← {q ∈ Q | p ∈ K_z(q)}
8:              case ψ = ¬ψ_1
9:                  Sat(ψ) ← Q \ Sat(ψ)
10:             case ψ = φ
11:                 Sat(ψ) ← φ
12:             case ψ = ψ_1 ∧ ψ_2
13:                 Sat(ψ) ← Sat(ψ_1) ∩ Sat(ψ_2)
14:             case ψ = ⟨⊣_n⟩(ψ_1 U ψ_2)
15:                 Sat(ψ) ← ⊣_n U(Sat(ψ_1), Sat(ψ_2))
16:             case ψ = ⟨⊣_n⟩(ψ_1 R ψ_2)
17:                 Sat(ψ) ← ⊣_n R(Sat(ψ_1), Sat(ψ_2))
18:             case ψ = j.ψ_1
19:                 Sat(ψ) ← Sat(ψ_1)
```

Algorithm 2 Backward search for computing $\dashv_n$U
Input: A TOL formula $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)$
Output: Sat($\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)$) ← $\{q \in Q \mid \mathcal{M}, q \models \langle\!\dashv_n\rangle(\psi_1 \,\mathsf{U}\, \psi_2)\}$

```
1: X ← ∅
2: Y ← ψ_2
3: while Y ≠ X do
4:     X ← Y
5:     Y ← ψ_2 ∪ (ψ_1 ∩ ▼(n, X))
6: return Y
```

Algorithm 3 Backward search for computing $\dashv_n$R
Input: A formula $\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{R}\, \psi_2)$
Output: Sat($\langle\!\dashv_n\rangle(\psi_1 \,\mathsf{R}\, \psi_2)$) ← $\{q \in Q \mid \mathcal{M}, q \models \langle\!\dashv_n\rangle(\psi_1 \,\mathsf{R}\, \psi_2)\}$

```
1: X ← ⊤
2: Y ← ψ_2
3: while Y ≠ X do
4:     X ← Y
5:     Y ← ψ_2 ∩ (ψ_1 ∪ ▼(n, X))
6: return Y
```

Proposition 2 (Termination). *Let $\mathcal{A}$ be a WTA and φ be a formula. Algorithm 1 always terminates on input ZG($\mathcal{A}$, φ).*

Proof. (Sketch) Algorithm 1 computes the zone graph in a finite time. Since finite sets bound the number of iterations, the computation of the subformulas $\mathsf{Sub}(\psi)$ and the updating of the labeling function K_z are also bounded. Thus, Algorithm 1 terminates. □

Let us now prove the correctness of the model checking algorithm. The following lemma is first defined.

Lemma 1. *Let φ be a formula, and $\mathcal{A}$ be a WTA. The extended state $s = (l, v, \mu)$ of the corresponding WTS($\mathcal{A}$) satisfies φ iff, the symbolic state $q = (l, Z)$ of the corresponding zone graph ZG($\mathcal{A}$, φ) satisfies the formula φ.*

Proposition 3. *Let $\mathcal{A}$ be a WTA, φ be aTOL formula and $\mathcal{M}$ = WTS($\mathcal{A}$) be a WTS. Then, $p \in K_z(q)$ iff, $\mathcal{M}, s \models \varphi$ where $q = (l, Z) \in Q$ and $s = (l, v, \mu) \in S$.*

Proof. (Sketch) We show by induction over the structure of φ that, for every $\psi \in \mathsf{Sub}(\varphi)$ and $q = (l, Z) \in Q$, $p \in K_z(q)$ holds iff, $\mathcal{M}, s \models \psi$.

(**Soundness**) For every $\psi \in \mathsf{Sub}(\varphi)$ and $q = (l, Z) \in Q$, $p \in K_z(q)$ implies $\mathcal{M}, s \models \psi$. We prove this by induction over the structure of ψ as follows. The base cases, $\psi = \top$ and $\psi = p$ ($p \in \mathsf{AP}$), are obvious. For the induction step, the cases of boolean combinations, $\psi = \neg\psi$ and $\psi = \psi_1 \wedge \psi_2$, of maximal state formulas is trivial. The induction step for the remaining obstruction operators is as follows: If $\psi = \dashv_n \mathsf{U}(\mathsf{Sat}(\psi_1), \mathsf{Sat}(\psi_2))$. Let Y be the set of symbolic states of Q that is returned by algorithm 2 at line 6. We need to show that $Y = \mathsf{Sat}(\psi_2)$ provided that $X = \mathsf{Sat}(\psi_1)$. We first show that $\mathsf{Sat}(\psi) \subseteq Y$. Suppose that $q \in \mathsf{Sat}(\psi)$. By the definition of satisfaction, this means that there is a strategy $\mathfrak{S}$ such that given any $\rho = q_1, q_2, \ldots$ in $Out(q, \mathfrak{S})$. Note that since the cardinality of a zone graph $\mathcal{M}$ = ZG($\mathcal{A}$, φ) is finite (i.e., finite zones) and we can suppose that $\mathfrak{S}$ is memoryless, we can focus on the finite prefix $q_1, \ldots q_m$ of ρ in which all the q_i are distinct. Let A_i (for $i < |\mathcal{M}|$) be the value of the variable A before the first i-th iteration of the algorithm. We show that if $C \subseteq A_i$ then $C \subseteq A_{i+1}$. Firstly, note that $A_i \subseteq \mathsf{Sat}(\psi_1)$ for all i. In algorithm 2 at line 5, we have that $A_{i+1} = \blacktriangledown(n, A_i) \cap \mathsf{Sat}(\psi_1)$, i.e., A_{i+1} is computed by taking all the element of $\mathsf{Sat}(\psi)$ that have at most n successors that are not in A_i. Hence, $p \in K_z(q)$ implies $\mathcal{M}, s \models \psi$. If $\psi = \dashv_n \mathsf{R}(\mathsf{Sat}(\psi_1), \mathsf{Sat}(\psi_2))$ then the proof is similar to the above.

(**Completeness**) We prove that if $\mathcal{M}, s \models \varphi$ then $s \in \mathsf{Sat}(\varphi)$. So, we can prove whether for every $\psi \in \mathsf{Sub}(\varphi)$ and $s \in S$, implies $\mathcal{M}, s \not\models \psi$ as follows. We prove this over the structure of ψ. The base cases, $\psi = \top$ and $\psi = p$ ($p \in \mathsf{AP}$), are obvious. For the induction step, the cases of boolean combinations, $\psi = \neg\psi$, then ψ was model checked and it was found to be true. Thus, $\mathcal{M}, s \not\models \psi$. For $\psi = \psi_1 \wedge \psi_2$, then ψ_1 and ψ_2 were model checked and at least one of them was found to be false. Therefore, $\mathcal{M}, s \not\models \psi$. The proof for $\psi = \dashv_n \mathsf{U}(\mathsf{Sat}(\psi_1), \mathsf{Sat}(\psi_2))$ then $\mathcal{M}, s \not\models \psi$ is similar to the above case (similar for R). □

The following theorem establishes the complexity of our model checking algorithm 1.

Theorem 1. *The model checking problem of TOL on WTA is PSPACE-complete.*

Proof. PSPACE-hardness: (Sketch) The proof follows from the PSPA CE-hardness of the model checking of the logic TCTL over TA [2], since WTA [4] are an extension of TA and TOL is the corresponding extension of TCTL and OL [18]: If we take the 0-fragment of TOL to be the set of TOL formulas in which the grade of any strategic operator is 0 (i.e., TOL^0) then TOL^0 = TCTL and WTA = TA.

PSPACE-membership: (Sketch) To prove PSPACE-membership, we use the idea suggested in [2]. Let $\mathcal{A}$ be a WTA, $\varphi \in$ TOL, D the number of clocks of the automaton $\mathcal{A}$, C_x the maximal constant associated with of clocks $\mathcal{A}$ and φ, m the nesting depth of the largest fixed-point quantifier in φ. We consider the zone graph $\mathsf{ZG}(\mathcal{A}, \varphi)$ [11] associated with $\mathcal{A}$ and the formula φ with clocks X. The zone graph depends on the maximum constants with which the clocks in $\mathcal{A}$ and φ are compared. Using the zone graph $\mathsf{ZG}(\mathcal{A}, \varphi)$, model checking of TOL formulas can be done in polynomial time in the number of D, C, and m. This can be shown as in [2]. According to [1], $\mathcal{A} \models \varphi$ iff $\mathcal{A}' \models \varphi$, where $\mathcal{A}' = untimed(\mathcal{A})$ is the untimed automaton associated with $\mathcal{A}$ and φ (the zone graph $\mathsf{ZG}(\mathcal{A}, \varphi)$ [11]). The size of $\mathcal{A}'$ is polynomial in the length of the timing constraints of the given WTA automaton and in the length of the formula φ (assuming binary encoding of the constants), that is, $|\mathcal{A}'| = O(|\varphi| \cdot (|L| + |T|) \cdot D! \cdot \prod_{x \in X} C_x)$. The zone graph $\mathcal{A}'$ can be constructed in linear time, which is also bounded by $O(\varphi \cdot (|L| + |T|) \cdot D! \cdot \prod_{x \in X} C_x)$ [2]. On the zone graph, untimed model checking can be done in time $O((|\varphi| \cdot |\mathcal{A}'|)$. Obviously, we get an algorithm of time complexity $O(|\varphi| \cdot (|L| + |T|))$. □

5 TOL VS. OTHER LOGICS

In this section, we establish relative relation between TOL with the Timed Computation Tree Logic (TCTL) [29], Timed μ-Calculus (T_μ) [29] and Timed Alternating-Time Temporal Logic (TATL) [30].

5.1 TOL and TCTL

Here, we show that TOL extends TCTL[29] with a reduction to a fragment of our logic. We define the 0-fragment of TOL to be the set of TOL formulas in which the grade of any strategic operator is 0. We denote by TOL^0 such a fragment. Let $(-)^\bullet$ be the mapping from TOL^0 to TCTL formulas that translate each strategic operator $\langle\dashv_0\rangle$ with the universal path operator A of TCTL, i.e., the function recursively defined as follows. Let φ be a TOL formula. Then TCTL fragment formula $(\varphi)^\bullet$ is defined as follows, where $p \in \mathsf{AP}$

$$
\begin{array}{lcl}
(\top)^\bullet & = & \top \\
(p)^\bullet & = & p \\
(\neg\varphi)^\bullet & = & \neg(\varphi)^\bullet \\
(\varphi_1 \wedge \varphi_2)^\bullet & = & (\varphi_1)^\bullet \wedge (\varphi_2)^\bullet \\
(\phi)^\bullet & = & \phi \\
(\mathsf{A}(\varphi_1 \,\mathsf{U}\, \varphi_2))^\bullet & = & \langle\dashv_0\rangle((\varphi_1)^\bullet \,\mathsf{U}\, (\varphi_2)^\bullet) \\
(\mathsf{A}(\varphi_1 \,\mathsf{R}\, \varphi_2))^\bullet & = & \langle\dashv_0\rangle((\varphi_1)^\bullet \,\mathsf{R}\, (\varphi_2)^\bullet) \\
(j.\varphi)^\bullet & = & j.(\varphi)^\bullet
\end{array}
$$

The translation from TOL to TCTL can be reversed. Our TOL logic makes two important contributions to TCTL. We extend TCTL with the same model checking complexity. Thus, we gain expressiveness without sacrificing tractability.

Theorem 2. *Let $\mathcal{A}$ be a WTA. For every model $\mathcal{M}$ = WTS($\mathcal{A}$), state s, and formula $\varphi \in TOL^0$, we have that $\mathcal{M}, s \models \varphi$ if and only if $\mathcal{M}, s \models_{TCTL} (\varphi)^\bullet$, where $\models_{TCTL}$ is the TCTL satisfaction relation.*

Proof. The result follows by observing that, for any state s, the set of paths compatible with a 0-strategy $\mathfrak{S}$ starting at s is equal to the set of paths starting at s and that given any two 0-strategies $\mathfrak{S}_1$ and $\mathfrak{S}_2$ we have that $Out(s, \mathfrak{S}_1) = Out(s, \mathfrak{S}_2)$. □

5.2 TOL and T_μ

T_μ is an extension of the modal μ-calculus [34] with clocks. Let AP be a non-empty at most countable set of atomic propositions, and $\mathcal{V}$ a non-empty at most countable set of formula variables. The formal definition of the formulas is as follows.

$$\varphi ::= \top \mid p \mid Y \mid \phi \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \rhd \varphi_2 \mid j.\varphi \mid \mu Y.\varphi$$

where $p \in \mathsf{AP}$, $Y \in \mathcal{V}$, $\phi \in \Phi(X)$ and $j \in X_\varphi$, where X is the set of clocks of the automaton and X_φ is a set of clocks of the formula. In $\mu Y.\varphi$ it is required that the variable Y occurs in the scope of an even number of negations in φ. The greatest fix-point operator can be defined by $\nu Y.\varphi$ = The $\rhd$ operator can be considered a (timed) next operator, where a state satisfies $\varphi_1 \rhd \varphi_2$ if one of its time successors has an action transition whose destination state satisfies φ_2, and every intermediate time successor (including this one) fulfills φ_1 or φ_2. Here, we show that TOL can be encoded into T_μ with a reduction to a fragment of our logic. Specifically, we show how to translate each TOL formula φ to a T_μ formula $(\varphi)^{T\mu}$ and that given a WTA $\mathcal{A}$ such that $\mathcal{M} = \mathsf{WTS}(\mathcal{A})$. Now, let $(-)^{T\mu}$ be the function from TOL formulas to T_μ formulas, defined as follows:

$$\begin{array}{lcl}
(\top)^{T\mu} & = & \top \\
(p)^{T\mu} & = & p \\
(\phi)^{T\mu} & = & \phi \\
(\neg\varphi)^{T\mu} & = & \neg(\varphi)^{T\mu} \\
(\varphi_1 \wedge \varphi_2)^{T\mu} & = & (\varphi_1)^{T\mu} \wedge (\varphi_2)^{T\mu} \\
(j.\varphi)^{T\mu} & = & j.(\varphi)^{T\mu} \\
(\langle\!\!\downarrow_0\rangle(\varphi_1 \,\mathsf{U}\, \varphi_2))^{T\mu} & = & \mu Y.((\varphi_2)^{T\mu} \vee ((\varphi_1)^{T\mu} \rhd Y)) \\
(\langle\!\!\downarrow_0\rangle(\varphi_1 \,\mathsf{R}\, \varphi_2)))^{T\mu} & = & \nu Y((\varphi_2)^{T\mu} \wedge ((\varphi_1)^{T\mu} \rhd Y))
\end{array}$$

Note that $(\varphi)^{T\mu}$ is a closed T_μ formula for every formula φ. Let us call *unary* a TOL model $\mathcal{M}$ such that $W(t) = 1$ for all $t \in T$.

Theorem 3. *If $\mathcal{M}$ is a unary WTS then for every TOL formula φ and state s we have that $\mathcal{M}, s \models \varphi$ iff $\mathcal{M}, s \models (\varphi)^{T\mu}$.*

5.3 TOL and TATL

Here, we compare our TOL with TATL [30]. We show that given a TOL formula φ and a WTA $\mathcal{A}$ ($\mathcal{M} = \mathsf{WTS}(\mathcal{A})$) that satisfies it, there is a Timed Concurrent Game Structure (TCGS)[14] that satisfies a TATL translation of φ. First, define a rooted TOL as a pair $\langle \mathcal{M}, s\rangle$ where $\mathcal{M}$ is a $\mathsf{WTS}(\mathcal{A})$ and s is one of its states. Given a natural number n, let $S^{\leq n}$ be the subset of $S \times 2^E$ defined by $(s, E) \in S^{\leq n}$ iff either $E = \emptyset$ or each $e \in E$ has s as source and $(\sum_{e\in E} W(e)) \leq n$. If $\langle \mathcal{M}, s\rangle$ is a rooted TOL model and n is a natural number, then $\mathbb{G}^n_{\mathcal{M}} = \langle Q, q_i, \mathsf{AP}, \mathsf{Ag}, X, I, act_D, act_T, P, \delta, \mathcal{V}\rangle$ is the TCGS, where: (a) $Q = Q_D \cup Q_T$ is a set of states, where $Q_D = S$ and $Q_T = S^{\leq n}$. Moreover, $q_I = s$ is the initial state. The set Q_D is the set of states where is the Demon's turn to move, while Q_T is the set of states in which is the Traveler's turn to move. (b) AP is a set of atomic formulas labeling states of $\mathcal{M}$. (c) is a set of clocks. (d) is the invariant of each state.(e) $\mathsf{Ag} = \{D, T\}$ where D is the Demon and T is the Traveler. (f) The set of actions act_D of the Demon is equal to the set of subset of R appearing in $S^{\leq n}$ plus the idle action $\star$. More precisely $act_D = \{E \in 2^R : \exists q \in S^{\leq n} \wedge q = \langle s, E\rangle\} \cup \{\star\}$. (g) The set of actions act_T of the Traveler is $R \cup \{\star\}$. We denote by $act = act_D \cup act_T$. (h) The protocol function $P : Q \times \mathsf{Ag} \to 2^{act} \setminus \emptyset$ is defined as follows. For every $q \in Q_D$, we have that $P(q, i)$ is equal to $Y_q = \{E \in 2^R : \langle q, E\rangle \in S^{\leq n}\}$ if $i = D$, and $\{\star\}$ otherwise. For every $q \in Q_T$, we have that if $q = \langle s, E\rangle$ then $P(q, i)$ is equal to $\{e \in R : e \notin E \wedge s' \in S\}$ if $i = T$ and it is equal to $\{\star\}$ otherwise. (i) The transition function $\delta : Q \times (act_D \uplus \mathbb{R}_{\geq 0}) \times (act_T \uplus \mathbb{R}_{\geq 0}) \to Q$ is defined as follows: $\delta(q, E, \star) = \langle q, E\rangle$ iff $q \in Q_D$ and $\delta(q, \langle s, s'\rangle, \star) = s'$ iff $q = \langle s, E\rangle \in Q_T$ and $\langle s, s'\rangle \notin E$. (j) The labeling function $V : S \to 2^{\mathsf{AP}}$ is defined by $V(q) = K(q)$ for any $q \in Q_D$ and $V(q) = \emptyset$ for any $q \in Q_T$.

Note that given a $\mathcal{M}$ and a natural number n, the TCGS $\mathbb{G}^n_{\mathcal{M}}$ can have a number of states that is **exponential** in the number of states of $\mathcal{M}$. Consider the function from TOL formulas to TATL formulas, inductively defined by:

$$\begin{array}{lcl}
(\top)^{\mathsf{A}} & = & \top \\
(p)^{\mathsf{A}} & = & p \\
(\phi)^{\mathsf{A}} & = & \phi \\
(\neg\varphi)^{\mathsf{A}} & = & \neg(\varphi)^{\mathsf{A}} \\
(\varphi_1 \wedge \varphi_2)^{\mathsf{A}} & = & (\varphi_1)^{\mathsf{A}} \wedge (\varphi_2)^{\mathsf{A}} \\
(j.\varphi)^{\mathsf{A}} & = & j.(\varphi)^{\mathsf{A}} \\
(\langle\!\!\downarrow_n\rangle(\varphi_1 \,\mathsf{U}\, \varphi_2))^{\mathsf{A}} & = & \langle\!\langle D\rangle\!\rangle^{\downarrow}(\varphi)^{\mathsf{A}} \,\mathsf{U}\, (\psi)^{\mathsf{A}} \\
(\langle\!\!\downarrow_n\rangle(\varphi_1 \,\mathsf{R}\, \varphi_2)))^{\mathsf{A}} & = & \langle\!\langle D\rangle\!\rangle^{\downarrow}(\varphi)^{\mathsf{A}} \,\mathsf{R}\, (\psi)^{\mathsf{A}}
\end{array}$$

Given a TCGS $\mathbb{G}^n_{\mathcal{M}}$ as the one defined above, and a path ρ of the TCGS, we write ρ^D for the subsequence of ρ containing only states that are in Q_D. If Δ is a TATL strategy and $q \in Q_D$ is a state, then $Out^D(q, \Delta)$ denotes the set of sequences $\{\rho \in Q_D^\omega : \rho = \pi^D$ for some $\pi \in Out(q, \Delta)\}$.

Theorem 4. *Let φ be any TOL formula that contains at most a strategic operator $\langle\!\!\downarrow_n\rangle$, we have that $\mathcal{M}, s \models \varphi$ iff $\mathbb{G}^n_{\mathcal{M}}, s \models_D (\varphi)^{\mathsf{A}}$.*

6 CASE STUDY

Based on the concepts of AG presented in Subsection 2.1, we would like to check whether there are MTD response strategies to satisfy some security objectives. To achieve this, we assume that: (1) The defender always knows the AG state reached by the attacker (called *attacker current state*). (2) At every moment, there is a unique attacker current state in the AG. (3) When detecting the attacker current state, the defender can activate a (or a subset of) MTD(s) temporarily removing an (a subset of) outgoing edge(s). The defender cannot remove edges that are not outgoing from the attacker current state. (4) The sum of the costs associated to the subset of deactivated is less than a given threshold. (5) When the attacker launches an attack from its current state, if the corresponding edge has not been removed by the defender, then the attack always succeeds (i.e. the attacker reaches the next state). (6) When the attacker launches an attack from its current state, if the corresponding edge has been removed by the defender, then the attack always fails (i.e. the attacker stays in its current state). Consider the model in Fig. 1. We can assume that when reaching state s_1, s_3, or s_5 the attacker has root privilege on a given critical server s. In addition, if the attacker completes attack steps a_6 or a_7 (that is, it reaches state s_5), then the defender will obtain information on the identity

of the attacker. Let a be an atomic proposition that expresses the fact that the identity of the attacker is known. Let r_s be an atomic proposition expressing the fact that the attacker has root privilege on the server s. We can express, via TOL formulas, the following security objectives:

- *The attacker will never be able to obtain root privileges on server s unless the defender can obtain information about his identity within* 3 *time units*: that is, either we want the attacker to never reach a state satisfying r_s *or* if the attacker reaches such a state, the defender wants to be able to identify it within 3 time units (a). By using t_1 as a variable, the following TOL formula captures the objective: $\varphi_1 := j.\langle\!\downarrow_{t_1}\rangle \mathsf{G}\,(r_s \vee (r_s \rightarrow \langle\!\downarrow_{t_1}\rangle \mathsf{F}(\mathrm{j} \leq 3 \wedge a)))$.
- *While the defender has not obtained information about the attacker identity within* 5 *time units, the attacker has not root privilege on the server s*: that is, we want r_s to be false *until* we have identified the attacker (a) within 5 time units, if such an identification ever happens. Thus, by using t_2 as a variable for a given threshold, we can write our objective by using the until connective: $\varphi_2 := j.\langle\!\downarrow_{t_2}\rangle(\neg r_s \ \wedge j \leq 5 \,\mathsf{U}\, a)$.

Suppose that t_1 and t_2 are respectively 3 and 4. Let $\mathcal{M} = \mathsf{WTS}(\mathcal{A})$, we have that $\mathcal{M}, s_0 \models \varphi_1 \wedge \varphi_2$. To satisfy φ_1 consider the 3-memoryless strategy $\mathfrak{S}_1$ that associates $\{\langle s_1, s_2\rangle\}$ to s_1, $\{\langle s_3, s_4\rangle\}$ to s_3, and $\emptyset$ to any other state of $\mathcal{M}$. Remark that for any path $\pi \in Out(s_0, \mathfrak{S}_1)$ and any $i \in \mathbb{N}$ we have that $\mathcal{M}, \pi_i \models r_s$ iff $\pi_i \in \{s_1, s_3, s_5\}$. Thus, we must establish that $\mathcal{M}$ satisfies $\langle\!\downarrow_3\rangle \mathsf{F}(j \leq 3 \ \wedge \ a)$ on s_1 (resp. s_3 and s_5). To do so, we remark that $Out(s_1, \mathfrak{S}_1)$ (resp. $Out(s_3, \mathfrak{S}_1)$ and $Out(s_5, \mathfrak{S}_1)$) only contains the path s_1, s_3, s_5^ω (resp. s_3, s_5^ω and s_5^ω) and that $\mathcal{M}, s_5 \models a$. Thus, we have obtained that there is a strategy (i.e. $\mathfrak{S}_1$) such that for all $\pi \in Out(s_0, \mathfrak{S}_1)$ and all $i \in \mathbb{N}$ either $\mathcal{M}, \pi_i \models \neg r_s$ or if $\mathcal{M}, \pi_i \models r_s$ then there is a strategy ($\mathfrak{S}_1$ itself) such that $\mathcal{M}, \rho_j \models a$ for some $j \geq 1$ and for all $\rho \in Out(\pi_i, \mathfrak{S}_1)$, as we wanted. Remark that if $t_1 < 3$ then it is impossible to satisfy φ_1 in $\mathcal{M}$ at s_0. For the specification $\varphi_2 = j.\langle\!\downarrow_4\rangle(\neg r_s \ \wedge \ j \leq 5 \,\mathsf{U}\, a)$, consider the 4-memoryless strategy $\mathfrak{S}_2$ that associates $\{\langle s_0, s_1\rangle\}$ to s_0, $\{\langle s_2, s_1\rangle, \langle s_2, s_3\rangle\}$ to s_2, $\{\langle s_4, s_3\rangle\}$ to s_4 and $\emptyset$ to s_5. The only path in $Out(s_0, \mathfrak{S}^\star)$ is $s_0, s_2, s_4, s_5^\omega$ and since s_5 satisfies a and all the other s_i do not satisfy r_s we obtain the wanted result.

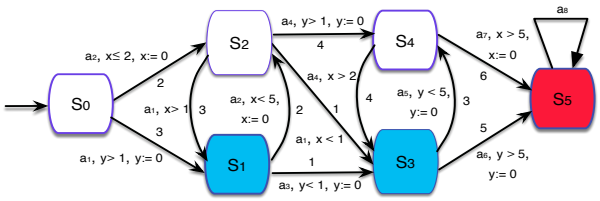


Figure 1: A WTA from [18] where states s_1, s_3 and s_5 represent the attacker's goals and blue states satisfy r_s, red state satisfies both a and r_s, and white states satisfy neither r_s nor a.

7 RELATED WORK

Several studies have explored the strategic capabilities of agents in dynamic game models.

Untimed Games and Strategic Logics [7, 38, 52] some research related to sabotage games have been introduced by van Benthem with the aim of studying the computational complexity of a special class of graph reachability problems in which an agent has the ability to delete edges. To reason about sabotage games, van Benthem introduced Sabotage Modal Logic (SML). The model checking problem for the sabotage modal logic is PSPACE-complete [38]. Our version of the games is not comparable to the sabotage games, because we provide the possibility to temporarily select subsets of edges, while in the sabotage games the saboteur can only delete one edge at a time. In this respect, our work is related to [17], where the authors use an extended version of sabotage modal logic, called Subset Sabotage Modal Logic (SSML), which allows for the deactivation of certain subsets of edges of a directed graph. The authors show that the model checking problems for such logics are decidable. Also, we recall that SSML is an extension of SML, but does not include temporal operators and quantitative information about the cost of edges, as we do. A Dynamic Escape Games (DEG) [51] is a variant of weighted two-player turn-based reachability games. In a DEG, an agent has the ability to inhibit edges. In [18] have been introduced an untimed Obstruction Logic (OL) which allows reasoning about two-player games played on a labeled and weighted directed graph. ATL [3] and SCTL [6] are extensions of CTL with the notion of strategic modality. These kinds of logics are used to express properties of agents as their possible actions. However, none of these logics includes quantitative information about cost edges, real time and time operators like our TOL. **Timed Games and Strategic Logics** [6, 24, 30, 33, 48] several research works have focused on extending games and logics to the real-time domain. The most established model in this respect is the Timed Game Automata (TGA)[12, 24]. A TGA is a TA whose set of transitions is divided among the different players. At each step, each player chooses one of her possible transitions, as well as some time she wants to wait before firing her chosen transition. The logics ATL and CTL has also been extended to TATL [30, 33] and STCTL [6], in which formula clocks are used to express the time requirements. It is exponentially decidable whether a TATL formula satisfies a TGA [30]. However, in [6] was shown that STCTL is more expressive than TATL and the model checking problem for STCTL with memoryless perfect information is of the same complexity as for TCTL. Model checking for TATL with continuous semantics is undecidable [6]. However, all these logics do not use dynamic models.

8 CONCLUSIONS

In this paper, we introduced TOL, a logic for reasoning about two-player games with real-time goals, where one player can locally and temporarily modify the timed game structure. We proved that its model-checking problem is PSPACE-complete and demonstrated its applicability to cybersecurity properties. For future work, we aim to explore several extensions. One direction is to consider timed games with multiple players, forming *coalitions* of travelers, similar to how TATL relates to TCTL. Another is integrating probability events to TOL. Lastly, we seek to introduce imperfect information, though this is generally undecidable [23]. To address this, we could approximate perfect information [8], use bounded memory [9], or apply hybrid techniques [25, 26]. These avenues would enhance the applicability of TOL to real-world problems.

REFERENCES

[1] Rajeev Alur. 1991. *Techniques for automatic verification of real-time systems.* Ph.D. Dissertation. Stanford University, USA. https://searchworks.stanford.edu/view/2112175

[2] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theor. Comput. Sci.* 126, 2 (1994), 183–235. https://doi.org/10.1016/0304-3975(94)90010-8

[3] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49, 5 (2002), 672–713. https://doi.org/10.1145/585265.585270

[4] Rajeev Alur, Salvatore La Torre, and George J. Pappas. 2001. Optimal Paths in Weighted Timed Automata. In *Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001, Proceedings (Lecture Notes in Computer Science, Vol. 2034)*, Maria Domenica Di Benedetto and Alberto L. Sangiovanni-Vincentelli (Eds.). Springer, Berlin, Heidelberg, 49–62. https://doi.org/10.1007/3-540-45351-2_8

[5] Francesco Alzetta, Paolo Giorgini, Amro Najjar, Michael Ignaz Schumacher, and Davide Calvaresi. 2020. In-Time Explainability in Multi-Agent Systems: Challenges, Opportunities, and Roadmap. In *Explainable, Transparent Autonomous Agents and Multi-Agent Systems - Second International Workshop, EXTRAAMAS 2020, Auckland, New Zealand, May 9-13, 2020*, Vol. 12175. Springer, Berlin, Heidelberg, 39–53. https://doi.org/10.1007/978-3-030-51924-7_3

[6] Jaime Arias, Wojciech Jamroga, Wojciech Penczek, Laure Petrucci, and Teofil Sidoruk. 2023. Strategic (Timed) Computation Tree Logic. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023*, Noa Agmon, Bo An, Alessandro Ricci, and William Yeoh (Eds.). ACM, Richland, SC, 382–390. https://doi.org/10.5555/3545946.3598661

[7] Guillaume Aucher, Johan van Benthem, and Davide Grossi. 2018. Modal logics of sabotage revisited. *J. Log. Comput.* 28, 2 (2018), 269–303. https://doi.org/10.1093/LOGCOM/EXX034

[8] Francesco Belardinelli, Angelo Ferrando, and Vadim Malvone. 2023. An abstraction-refinement framework for verifying strategic properties in multi-agent systems with imperfect information. *Artif. Intell.* 316 (2023), 103847. https://doi.org/10.1016/J.ARTINT.2022.103847

[9] F. Belardinelli, A. Lomuscio, V. Malvone, and E. Yu. 2022. Approximating Perfect Recall when Model Checking Strategic Abilities: Theory and Applications. *J. Artif. Intell. Res.* 73 (2022), 36.

[10] F. Belardinelli, V. Malvone, and A. Slimani. 2020. *A Tool for Verifying Strategic Properties in MAS with Imperfect Information.* Springer. https://github.com/VadimMalvone/

[11] Johan Bengtsson and Wang Yi. 2003. Timed Automata: Semantics, Algorithms and Tools. In *Lectures on Concurrency and Petri Nets, Advances in Petri Nets (Lecture Notes in Computer Science, Vol. 3098)*, Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg (Eds.). Springer, Berlin, Heidelberg, 87–124. https://doi.org/10.1007/978-3-540-27755-2_3

[12] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, and Nicolas Markey. 2011. Quantitative analysis of real-time systems using priced timed automata. *Commun. ACM* 54, 9 (2011), 78–87. https://doi.org/10.1145/1995376.1995396

[13] Patricia Bouyer, François Laroussinie, and Pierre-Alain Reynier. 2005. Diagonal Constraints in Timed Automata: Forward Analysis of Timed Systems. In *Formal Modeling and Analysis of Timed Systems, Third International Conference, FORMATS 2005, Uppsala, Sweden, September 26-28, 2005, Proceedings (Lecture Notes in Computer Science, Vol. 3829)*, Paul Pettersson and Wang Yi (Eds.). Springer, Berlin, Heidelberg, 112–126. https://doi.org/10.1007/11603009_10

[14] Thomas Brihaye, François Laroussinie, Nicolas Markey, and Ghassan Oreiby. 2007. Timed Concurrent Game Structures. In *CONCUR 2007 - Concurrency Theory, 18th International Conference, CONCUR 2007, Lisbon, Portugal, September 3-8, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4703)*, Luís Caires and Vasco Thudichum Vasconcelos (Eds.). Springer, Berlin, Heidelberg, 445–459. https://doi.org/10.1007/978-3-540-74407-8_30

[15] Davide Calvaresi, Yashin Dicente Cid, Mauro Marinoni, Aldo Franco Dragoni, Amro Najjar, and Michael Schumacher. 2021. Real-time multi-agent systems: rationality, formal model, and empirical results. *Auton. Agents Multi Agent Syst.* 35, 1 (2021), 12. https://doi.org/10.1007/S10458-020-09492-5

[16] Davide Calvaresi, Mauro Marinoni, Arnon Sturm, Michael Schumacher, and Giorgio C. Buttazzo. 2017. The challenge of real-time multi-agent systems for enabling IoT and CPS. In *Proceedings of the International Conference on Web Intelligence, Leipzig, Germany, August 23-26, 2017.* ACM, New York, NY, USA, 356–364. https://doi.org/10.1145/3106426.3106518

[17] Davide Catta, Jean Leneutre, and Vadim Malvone. 2023. Attack Graphs & Subset Sabotage Games. *Intelligenza Artificiale* 17, 1 (2023), 77–88. https://doi.org/10.3233/IA-221080

[18] Davide Catta, Jean Leneutre, and Vadim Malvone. 2023. Obstruction Logic: A Strategic Temporal Logic to Reason About Dynamic Game Models. In *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland (Frontiers in Artificial Intelligence and Applications, Vol. 372).* IOS Press, Berlin, Heidelberg, 365–372. https://doi.org/10.3233/FAIA230292

[19] Jin-Hee Cho, Dilli P Sharma, Hooman Alavizadeh, Seunghyun Yoon, Noam Ben-Asher, Terrence J Moore, Dong Seong Kim, Hyuk Lim, and Frederica F Nelson. 2020. Toward proactive, adaptive defense: A survey on moving target defense. *IEEE Communications Surveys & Tutorials* 22, 1 (2020), 709–745.

[20] Edmund M. Clarke and E. Allen Emerson. 2008. Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic. In *25 Years of Model Checking - History, Achievements, Perspectives (Lecture Notes in Computer Science, Vol. 5000)*, Orna Grumberg and Helmut Veith (Eds.). Springer, Berlin, Heidelberg, 196–215. https://doi.org/10.1007/978-3-540-69850-0_12

[21] Edmund M. Clarke, Orna Grumberg, Daniel Kroening, Doron A. Peled, and Helmut Veith. 2018. *Model checking, 2nd Edition.* MIT Press, Berlin, Heidelberg. https://mitpress.mit.edu/books/model-checking-second-edition

[22] Stephen Corley, Diego Magro, Fabio Malabocchia, Jens Meinköhn, Luisella Sisto, Sahin Albayrak, and Alexander Grosse. 1998. The Application of Intelligent and Mobile Agents to the Management of Software Problems in Telecommunications. In *Intelligent Agents for Telecommunication Applications, Second International Workshop, IATA '98, Paris, France, July 1998, Proceedings (Lecture Notes in Computer Science, Vol. 1437).* Springer, Berlin, Heidelberg, 118–128. https://doi.org/10.1007/BFB0053948

[23] Catalin Dima and Ferucio Laurentiu Tiplea. 2011. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. *CoRR* abs/1102.4225 (2011), 435–444. arXiv:1102.4225 http://arxiv.org/abs/1102.4225

[24] Marco Faella, Salvatore La Torre, and Aniello Murano. 2014. Automata-theoretic decision of timed games. *Theor. Comput. Sci.* 515 (2014), 46–63. https://doi.org/10.1016/J.TCS.2013.08.021

[25] Angelo Ferrando and Vadim Malvone. 2022. Towards the Combination of Model Checking and Runtime Verification on Multi-agent Systems. In *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. (Lecture Notes in Computer Science, Vol. 13616).* Springer, Berlin, Heidelberg, 140–152. https://doi.org/10.1007/978-3-031-18192-4_12

[26] Angelo Ferrando and Vadim Malvone. 2023. Towards the Verification of Strategic Properties in Multi-Agent Systems with Imperfect Information. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023.* ACM, Berlin, Heidelberg, 793–801. https://doi.org/10.5555/3545946.3598713

[27] R. Govind, Frédéric Herbreteau, B. Srivathsan, and Igor Walukiewicz. 2019. Revisiting Local Time Semantics for Networks of Timed Automata. In *30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands (LIPIcs, Vol. 140).* Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Berlin, Heidelberg, 16:1–16:15.

[28] Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. 1997. HYTECH: A Model Checker for Hybrid Systems. In *Computer Aided Verification, 9th International Conference, CAV '97, Haifa, Israel, June 22-25, 1997, Proceedings (Lecture Notes in Computer Science, Vol. 1254).* Springer, Berlin, Heidelberg, 460–463. https://doi.org/10.1007/3-540-63166-6_48

[29] Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. 1994. Symbolic Model Checking for Real-Time Systems. *Inf. Comput.* 111, 2 (1994), 193–244. https://doi.org/10.1006/INCO.1994.1045

[30] Thomas A. Henzinger and Vinayak S. Prabhu. 2006. Timed Alternating-Time Temporal Logic. In *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings (Lecture Notes in Computer Science, Vol. 4202)*, Eugene Asarin and Patricia Bouyer (Eds.). Springer, Berlin, Heidelberg, 1–17. https://doi.org/10.1007/11867340_1

[31] Wojciech Jamroga and Aniello Murano. 2015. Module Checking of Strategic Ability. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, Gerhard Weiss, Pinar Yolum, Rafael H. Bordini, and Edith Elkind (Eds.). ACM, Berlin, Heidelberg, 227–235. http://dl.acm.org/citation.cfm?id=2772911

[32] Kerem Kaynar. 2016. A taxonomy for attack graph generation and usage in network security. *J. Inf. Secur. Appl.* 29 (2016), 27–56. https://doi.org/10.1016/J.JISA.2016.02.001

[33] Michal Knapik, Étienne André, Laure Petrucci, Wojciech Jamroga, and Wojciech Penczek. 2019. Timed ATL: Forget Memory, Just Count. *J. Artif. Intell. Res.* 66 (2019), 197–223. https://doi.org/10.1613/JAIR.1.11612

[34] Orna Kupferman, Ulrike Sattler, and Moshe Y. Vardi. 2002. The Complexity of the Graded μ-Calculus. In *Automated Deduction - CADE-18, 18th International Conference on Automated Deduction, Copenhagen, Denmark, July 27-30, 2002, Proceedings (Lecture Notes in Computer Science, Vol. 2392)*, Andrei Voronkov (Ed.). Springer, Berlin, Heidelberg, 423–437. https://doi.org/10.1007/3-540-45620-1_34

[35] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. 2000. An automata-theoretic approach to branching-time model checking. *J. ACM* 47, 2 (2000), 312–360. https://doi.org/10.1145/333979.333987

[36] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. 2001. Module Checking. *Inf. Comput.* 164, 2 (2001), 322–344. https://doi.org/10.1006/INCO.2000.2893

[37] François Laroussinie, Nicolas Markey, and Ghassan Oreiby. 2006. Model-Checking Timed ATL for Durational Concurrent Game Structures. In *Proceedings of the 4th International Conferences on Formal Modelling and Analysis of Timed Systems (FORMATS'06) (Lecture Notes in Computer Science, Vol. 4202)*, Eugene Asarin

and Patricia Bouyer (Eds.). Springer-Verlag, Berlin, Heidelberg, 245–259. https://doi.org/10.1007/11867340_18

[38] Christof Löding and Philipp Rohde. 2003. Model Checking and Satisfiability for Sabotage Modal Logic. In *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings (Lecture Notes in Computer Science, Vol. 2914)*, Paritosh K. Pandya and Jaikumar Radhakrishnan (Eds.). Springer, Berlin, Heidelberg, 302–313. https://doi.org/10.1007/978-3-540-24597-1_26

[39] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. 2009. MCMAS: A Model Checker for the Verification of Multi-Agent Systems. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5643)*, Ahmed Bouajjani and Oded Maler (Eds.). Springer, Berlin, Heidelberg, 682–688. https://doi.org/10.1007/978-3-642-02658-4_55

[40] Alessio Lomuscio, Bozena Wozna, and Andrzej Zbrzezny. 2006. Bounded Model Checking Real-Time Multi-agent Systems with Clock Differences: Theory and Implementation. In *Model Checking and Artificial Intelligence, 4th Workshop, MoChArt IV, Riva del Garda, Italy, August 29, 2006, Revised Selected and Invited Papers (Lecture Notes in Computer Science, Vol. 4428)*, Stefan Edelkamp and Alessio Lomuscio (Eds.). Springer, Berlin, Heidelberg, 95–112. https://doi.org/10.1007/978-3-540-74128-2_7

[41] Hector Marco-Gisbert and Ismael Ripoll Ripoll. 2019. Address Space Layout Randomization Next Generation. *Applied Sciences* 9, 14 (2019), 39–53. https://www.mdpi.com/2076-3417/9/14/2928

[42] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2014. Reasoning About Strategies: On the Model-Checking Problem. *ACM Trans. Comput. Log.* 15, 4 (2014), 34:1–34:47. https://doi.org/10.1145/2631917

[43] Aniello Murano, Giuseppe Perelli, and Sasha Rubin. 2015. Multi-agent Path Planning in Known Dynamic Environments. In *PRIMA 2015: Principles and Practice of Multi-Agent Systems - 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings (Lecture Notes in Computer Science, Vol. 9387)*, Qingliang Chen, Paolo Torroni, Serena Villata, Jane Yung-jen Hsu, and Andrea Omicini (Eds.). Springer, Berlin, Heidelberg, 218–231. https://doi.org/10.1007/978-3-319-25524-8_14

[44] Hoang Nga Nguyen and Abdur Rakib. 2023. Formal Modelling and Verification of Probabilistic Resource Bounded Agents. *J. Log. Lang. Inf.* 32, 5 (2023), 829–859. https://doi.org/10.1007/S10849-023-09405-1

[45] James Jerson Ortiz, Moussa Amrani, and Pierre-Yves Schobbens. 2019. ML$_V$: A Distributed Real-Time Modal Logic. In *NASA Formal Methods - 11th International Symposium, NFM 2019, Houston, TX, USA, May 7-9, 2019, Proceedings (Lecture Notes in Computer Science, Vol. 11460)*, Julia M. Badger and Kristin Yvonne Rozier (Eds.). Springer, Berlin, Heidelberg, 19–35. https://doi.org/10.1007/978-3-030-20652-9_2

[46] Gordon D. Plotkin. 2004. A structural approach to operational semantics. *J. Log. Algebraic Methods Program.* 60-61 (2004), 17–139.

[47] A. Qasim, I. Fakhir, and S. Kazmi. 2015. Formal Specification and Verification of Real-Time Multi-Agent Systems using Timed-Arc Petri Nets. *Electrical and Computer Engineering* 12 (2015), 96–111.

[48] A. Qasim, S. Kanwal, A. Khalid, R. Kazmi S. Asad, and J. Hassan. 2019. Timed-Arc Petri-Nets based Agent Communication for Real-Time Multi-Agent Systems. *Journal of Advanced Computer Science and Applications,* 14 (2019), 139–153.

[49] Govind Rajanbabu. 2021. *Partial order reduction for timed systems. (Réduction d'ordre partiel pour les systèmes temporisés).* Ph.D. Dissertation. Chennai Mathematical Institute, Tamil Nadu, India. https://tel.archives-ouvertes.fr/tel-03346012

[50] Ramzi Ben Salah, Marius Bozga, and Oded Maler. 2006. On Interleaving in Timed Automata. In *CONCUR 2006 - Concurrency Theory, 17th International Conference, CONCUR 2006, Bonn, Germany, August 27-30, 2006, Proceedings (Lecture Notes in Computer Science, Vol. 4137)*, Christel Baier and Holger Hermanns (Eds.). Springer, Berlin, Heidelberg, 465–476. https://doi.org/10.1007/11817949_31

[51] Antonio Di Stasio, Paolo Domenico Lambiase, Vadim Malvone, and Aniello Murano. 2018. Dynamic Escape Game. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, Elisabeth André, Sven Koenig, Mehdi Dastani, and Gita Sukthankar (Eds.). International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, Berlin, Heidelberg, 1806–1808. http://dl.acm.org/citation.cfm?id=3237984

[52] Johan van Benthem. 2005. An Essay on Sabotage and Obstruction. In *Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday (Lecture Notes in Computer Science, Vol. 2605)*, Dieter Hutter and Werner Stephan (Eds.). Springer, Berlin, Heidelberg, 268–276. https://doi.org/10.1007/978-3-540-32254-2_16