Model Checking Timed ATL for Durational Concurrent Game Structures

François Laroussinie, Nicolas Markey, Ghassan Oreiby

▶ **To cite this version:**

François Laroussinie, Nicolas Markey, Ghassan Oreiby. Model Checking Timed ATL for Durational Concurrent Game Structures. Proceedings of the 4th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'06), 2006, Paris, France. pp.245-259, 10.1007/11867340_18 . hal-01194613

HAL Id: hal-01194613
https://hal.archives-ouvertes.fr/hal-01194613

Submitted on 7 Sep 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Model-Checking Timed ATL for Durational Concurrent Game Structures[⋆]

F. Laroussinie, N. Markey, and G. Oreiby[⋆⋆]

Lab. Spécification & Vérification
ENS de Cachan & CNRS UMR 8643
61, av. Pdt. Wilson, 94235 Cachan Cedex, France
email: {fl,markey,oreiby}@lsv.ens-cachan.fr

Abstract. We extend the framework of ATL model-checking to "simply timed" concurrent game structures, *i.e.*, multi-agent structures where each transition carry an integral duration (or interval thereof). While the case of single durations is easily handled from the semantics point of view, intervals of durations raise several interesting questions. Moreover subtle algorithmic problems have to be handled when dealing with model checking. We propose a semantics for which we develop efficient (PTIME) algorithms for timed ATL without equality constraints, while the general case is shown to be EXPTIME-complete.

Introduction

Verification and model-checking. The development of embedded reactive systems is impressive (both in terms of their number and of their complexity), and their formal verification can't be ignored. Model-checking [12, 7] is a well-established technique for verifying that (an automaton representing) such a system satisfies a given property. Following [21, 11, 22], temporal logics have been used for specifying those properties: *Linear time* temporal logics (*e.g.* LTL) expresses properties on each single execution of the model, while *branching time* temporal logics (*e.g.* CTL) deal with the computation tree of the model.

The model-checking technique has been extended to also handle quantitative measurement of time. In that framework, automata are equipped with real-valued clocks [2], and temporal logics are extended to also express quantitative constraints on the flow of time [1]. Again, this framework is now well understood, but the algorithms are noticeably more complex.

In order to lower the complexity of those algorithms, less expressive models and logics have been developed [14, 9, 17]. Those models are less expressive, but can be handled very efficiently, especially through symbolic model-checking algorithms using BDD techniques [8, 20, 9, 19].

[⋆] This work is partly supported by ACI Sécurité & Informatique CORTOS, a program of the French ministry of research.

[⋆⋆] This author is supported by a PhD grant from Région Ile-de-France.

Verification and control. In the late 80's, a new framework has been developed in the field of verification: control (and controller synthesis) [23]. The goal is now to build a controller that should prevent the (model of the) system from having unwanted behaviors.

This problem is closely related to (multi-player) games: solving such a game amounts to compute a strategy (if it exists) for a player so that he surely reaches a state where he is declared the winner. In that case, the underlying model is not a simple automaton, but rather a "concurrent game structure" (CGSs) [5], in which several agents concurrently decide on the behavior of the system. In order to reason with strategies, a new flavor of temporal logics has been defined: *alternating time* temporal logics (ATL) [4, 5]. This logic allows to express, for instance, that a coalition of agents has a strategy in order to always reach a winning location, or to always avoid reaching a bad locations. When the concurrent game structure is defined explicitly, ATL enjoys polynomial-time model-checking algorithms.

Our contribution. The goal of this paper is to extend the framework of ATL to (simply) timed systems. To that aim, we introduce *durational* CGSs (DCGSs), in which each transition is labeled with an interval of possible (integer) durations. Those durations are assumed to be atomic, *i.e.*, there are no intermediate state, and the complete duration elapses in one step.

We propose a semantics for DCGSs where we assume that each transition is associated with an extra agent, who is in charge of selecting the duration of that transition within the interval it is labeled with. We believe that this semantics is really interesting, as it allows to finely select which durations can be controlled by a coalition. Moreover, we show that it still enjoys polynomial-time quantitative model-checking algorithms in the case when no equality constraint is involved.

Related work. Our discrete-time extension of CGSs to DCGSs is inspired by that of [17], where efficient quantitative model-checking algorithms are proposed. Several other extensions of games with time have been proposed in the recent literature, *e.g.* [18, 3, 6, 10]. The semantics assumed there uses dense-time where players choose either to wait for a delay or to fire an action-transition. In [13], another dense-time semantics is proposed, working (roughly) as follows: each player chooses a (strictly positive) delay and a transition, and the game follows the player with the shortest delay. With this semantics, each player can take the others by surprise.

Those papers only deal with qualitative control objectives. In [24], Schobbens proposes a quantitative extension of ATL over timed CGSs (with a semantics of time similar to that of [13]). The resulting logic, ARTL*, a mixture of ATL and MITL, is shown decidable.

1 Definitions

1.1 Tight Durational CGS (TDCGS)

We extend the model of CGSs, see [5, 16].

Definition 1. *A TDCGS is a* 6*-tuple* $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg} \rangle$ *where:*

- Loc *is the (finite) set of* locations*;*
- $\mathsf{Agt} = \{a_1, \ldots, a_k\}$ *is a (finite) set of* agents *(or* players*);*
- AP *the set of* atomic propositions*;*
- $\mathsf{Lab}\colon \mathsf{Loc} \to 2^{\mathsf{AP}}$ *the labelling function;*
- $\mathsf{Mv}\colon \mathsf{Loc} \times \mathsf{Agt} \to \mathcal{P}(\mathbb{N}) \smallsetminus \{\varnothing\}$ *gives the set of possible moves at a given location for a given agent;*
- $\mathsf{Edg}\colon \mathsf{Loc} \times \mathbb{N}^k \to \mathsf{Loc} \times \mathbb{N}^{>0}$ *is* the transition table*, that is a partial function assigning a successor location and a duration for a move of all agents.*

The difference with classical CGSs is that each transition of a TDCGS carries a positive[1] integer, representing the duration (or cost) of that transition. Given a transition $\mathsf{Edg}(q, c_1, \ldots, c_k) = (q', t)$, we use $\mathsf{Edg}_\ell(q, c_1, \ldots, c_k)$ (resp. $\mathsf{Edg}_\tau(q, c_1, \ldots, c_k)$) to denote the location q' (resp. the duration t).

The semantics of a TDCGS is similar to that of classical CGSs: a *move* of agent a in location q is an integer c such that $c \in \mathsf{Mv}(q, a)$. Once each agent a_i has selected a move $c_i \in \mathsf{Mv}(q, a_i)$, the transition table Edg indicates the transition to be fired, namely $\mathsf{Edg}(q, c_1, \ldots, c_k)$.

Definition 2. *An* execution *of a TDCGS* $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg} \rangle$ *from a location* $q_0 \in \mathsf{Loc}$ *is an infinite sequence* $\rho = (q_0, d_0) \ldots (q_i, d_i) \ldots$ *such that:*

- $d_0 = 0$*;*
- *for each* i*, there exists a set of moves* $c_1^i, \ldots, c_k^i$ *such that*

$$(q_{i+1}, d_{i+1} - d_i) = \mathsf{Edg}(q_i, c_1^i, \ldots, c_k^i).$$

For an execution $\rho = (q_0, d_0) \ldots (q_i, d_i) \ldots$, the integer d_i is the *date* when arriving in q_i.

The interesting point with ATL, compared to standard temporal logics, is that it allows quantifications on *strategies* of (coalitions of) agents. A coalition is a subset of the set of agents. Now we introduce the notions of strategy and outcome:

Definition 3. *Let* $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg} \rangle$ *be a TDCGS.*

- *Let* $a \in \mathsf{Agt}$*. A* strategy σ_a *for* a *is a mapping that associates, with any finite prefix* $(q_0, d_0) \ldots (q_i, d_i)$ *of any execution, a possible move for agent* a *in* q_i*, i.e.* $\sigma_a((q_0, d_0) \ldots (q_i, d_i)) \in \mathsf{Mv}(q_i, a)$*.*
- *Let* $A \subseteq \mathsf{Agt}$ *be a coalition. A move for* A *from a location* q *is a family* $(c_a)_{a \in A}$ *: one move for each agent in* A*. We write* $\mathsf{Mv}(q, A)$ *to represent the set of all possible moves for* A *from* q*. Moreover a strategy* σ_A *for* A *is a family* $(\sigma_a)_{a \in A}$*.*
  We write $\overline{A}$ *for* $\mathsf{Agt} \smallsetminus A$*. Given a move* $c \in \mathsf{Mv}(q, A)$ *and* $\overline{c} \in \mathsf{Mv}(q, \overline{A})$*, we write* $\mathsf{Edg}(q, c \cdot \overline{c})$ *for the transition corresponding to these choices.*

[1] We require in this paper that the durations be non-zero. The case of zero-durations makes some of our algorithms slightly more difficult, and will be handled in a long version of this paper.

- *Let $A \subseteq \mathsf{Agt}$ be a coalition and σ_A be a strategy for A. An execution $\rho = (q_0, d_0) \ldots (q_i, d_i) \ldots$ is an* outcome *of σ_A from q_0 if, for any i, writing $c^i = \sigma_A((q_0, d_0) \ldots (q_i, d_i))$, there exists $\overline{c}^i \in \mathsf{Mv}(q_i, \overline{A})$ s.t.*

$$(q_{i+1}, d_{i+1} - d_i) = \mathsf{Edg}(q_i, c \cdot \overline{c}).$$

  We denote by $Out^{\mathcal{A}}(q, \sigma_A)$ the set of all outcomes of σ_A from q (we omit the superscript $\mathcal{A}$ when it is clear from the context).

Size of a TDCGS. The size $|\mathcal{A}|$ of $\mathcal{A}$ is the sum of the sizes of Loc and Edg. The size of Edg is defined as follows: $|\mathsf{Edg}| = \sum_{q \in \mathsf{Loc}} \sum_{c \in \mathsf{Mv}(q,\mathsf{Agt})} (1 + \lfloor \log(\mathsf{Edg}_\tau(q, c)) \rfloor)$.

1.2 Timed ATL (TATL)

Definition 4. *The syntax of* TATL *is defined by the following grammar:*

$$\mathsf{TATL} \ni \varphi_s, \psi_s ::= \top \mid P \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \langle\!\langle A \rangle\!\rangle\, \varphi_p$$
$$\varphi_p ::= \mathbf{X}\, \varphi_s \mid \varphi_s\, \mathbf{U}_{\sim\zeta}\, \psi_s \mid \varphi_s\, \mathbf{R}_{\sim\zeta}\, \psi_s$$

with $P \in \mathsf{AP}$, $A \subseteq \mathsf{Agt}$, $\sim \in \{<, \leq, =, \geq, >\}$, and $\zeta \in \mathbb{N}$.

We also define the usual shorthands, such as $\bot \stackrel{\text{def}}{\equiv} \neg\top$, $\langle\!\langle A \rangle\!\rangle\, \mathbf{F}_{\sim\zeta}\, \varphi_s \stackrel{\text{def}}{\equiv} \langle\!\langle A \rangle\!\rangle\, \top\, \mathbf{U}_{\sim\zeta}\, \varphi_s$, and $\langle\!\langle A \rangle\!\rangle\, \mathbf{G}_{\sim\zeta}\, \varphi_s \stackrel{\text{def}}{\equiv} \langle\!\langle A \rangle\!\rangle\, \bot\, \mathbf{R}_{\sim\zeta}\, \varphi_s$.

TATL formulae are interpreted over states of TDCGSs. Intuitively, the state-formula $\langle\!\langle A \rangle\!\rangle\, \varphi_p$ holds in q iff there exists a strategy for coalition A in order to enforce the path-formula φ_p along all the outcomes. Formally:

Definition 5. *The following clauses define when a location q (resp. an execution $\rho = (q_0, d_0)(q_1, d_1) \ldots$) of a TDCGS $\mathcal{A}$ satisfies a* TATL *formula φ_s (resp. a path-formula φ_p), written $q \models_{\mathcal{A}} \varphi_s$ (resp. $\rho \models_{\mathcal{A}} \varphi_p$), by induction over the formula (semantics of boolean operators and atomic propositions are omitted):*

$$\begin{aligned}
q \models_{\mathcal{A}} \langle\!\langle A \rangle\!\rangle\, \varphi_p \quad &\Leftrightarrow \quad \exists \sigma_A.\ \forall \rho \in Out(q, \sigma_A).\ \rho \models_{\mathcal{A}} \varphi_p \\
\rho \models_{\mathcal{A}} \mathbf{X}\, \varphi_s \quad &\Leftrightarrow \quad q_1 \models_{\mathcal{A}} \varphi_s \\
\rho \models_{\mathcal{A}} \varphi_s\, \mathbf{U}_{\sim\zeta}\, \psi_s \quad &\Leftrightarrow \quad \exists i \in \mathbb{N}.\ q_i \models_{\mathcal{A}} \psi_s,\ d_i \sim \zeta \\
&\qquad\quad \textit{and } q_j \models_{\mathcal{A}} \varphi_s \textit{ for any } 0 \leq j < i \\
\rho \models_{\mathcal{A}} \varphi_s\, \mathbf{R}_{\sim\zeta}\, \psi_s \quad &\Leftrightarrow \quad \rho \models_{\mathcal{A}} \neg(\neg\varphi_s\, \mathbf{U}_{\sim\zeta}\, \neg\psi_s)
\end{aligned}$$

When $\mathcal{A}$ is clear from the context, we just write $q \models \varphi$. Note that, contrary to usual definitions of ATL [4, 5], we include the "release" modality $\mathbf{R}$, as we proved in [16] that modality $\langle\!\langle A \rangle\!\rangle \mathbf{R}$ cannot be expressed using only $\langle\!\langle A \rangle\!\rangle \mathbf{U}$ and $\langle\!\langle A \rangle\!\rangle \mathbf{G}$. Intuitively, $\varphi_s\, \mathbf{R}_{\sim\zeta}\, \psi_s$ requires that ψ_s must hold when the condition "$\sim \zeta$" is fulfilled, but this global requirement is released as soon as φ_s holds. Formally:

$$\begin{aligned}
\rho \models_{\mathcal{A}} \varphi_s\, \mathbf{R}_{\sim\zeta}\, \psi_s \quad \Leftrightarrow \quad &\forall i \in \mathbb{N}.\ (d_i \sim \zeta \Rightarrow q_i \models_{\mathcal{A}} \psi_s) \\
&\text{or } \exists j < i.\ q_j \models_{\mathcal{A}} \varphi_s.
\end{aligned}$$

We use $\mathsf{TATL}_{\leq,\geq}$ to denote the fragment of TATL where subscripts "$= \zeta$" are not allowed in timing constraints for $\mathbf{U}$ and $\mathbf{R}$.

2 Model checking TATL

The complexity of model-checking an ATL formula over a CGS has been shown to be linear in both the size of the structure and the size of the formula [5]. On the other hand, TCTL model checking on DKSs is PTIME-complete when timing constraints contain no equality, while it is Δ_2^{P}-complete otherwise [17].

We present in this section our model-checking algorithm for TATL over TDCGSs. We explain how to handle modalities $\mathbf{U}_{\sim\zeta}$ and $\mathbf{R}_{\sim\zeta}$. We then gather up those algorithms in a global labeling algorithm, which is PTIME-complete when no equality constraint is involved, and EXPTIME-complete otherwise.

2.1 Modalities $\mathbf{U}_{\leq\zeta}$ and $\mathbf{R}_{\leq\zeta}$

First we consider the case of formula $\langle\!\langle A\rangle\!\rangle\, \varphi_1 \,\mathbf{U}_{\leq\zeta}\, \varphi_2$, that is, when the coalition A aims at reaching φ_2 within ζ time units (and verifying φ_1 in the intermediate states). We assume that states have already been labeled with φ_1 and φ_2, which can therefore be seen as atomic propositions. We have:

Lemma 6. *Let $\mathcal{A}$ be a TDCGS, and $\varphi = \langle\!\langle A\rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq\zeta}\, P_2$ be a* TATL *formula (with $P_1, P_2 \in \mathsf{AP}$). Then we can compute in time $O(|\mathsf{Loc}|\cdot|\mathsf{Edg}|)$ the set of locations of $\mathcal{A}$ where φ holds.*

Proof. For this proof, we define the extra modality $\mathbf{U}^{\leq i}_{\leq n}$, with the following semantics:

$$\rho \models_{\mathcal{A}} P_1 \,\mathbf{U}^{\leq i}_{\leq n}\, P_2 \quad\Leftrightarrow\quad \exists j \leq i.\ q_j \models_{\mathcal{A}} P_1,\ d_j \leq n, \\ \text{and } q_k \models_{\mathcal{A}} P_2 \text{ for any } 0 \leq k < j$$

This modality requires that the right-hand side formula be satisfied within at most i steps. It is clear that, for any $n \in \mathbb{N}$,

$$q \models \langle\!\langle A\rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq n}\, P_2 \quad\Leftrightarrow\quad q \models \langle\!\langle A\rangle\!\rangle\, P_1 \,\mathbf{U}^{\leq|\mathsf{Loc}|}_{\leq n}\, P_2. \tag{1}$$

Indeed, if all the outcomes of a strategy satisfy $P_1 \,\mathbf{U}_{\leq n}\, P_2$, it is possible to adapt that strategy so that each location is visited at most once along each outcome.

We now define functions $v_i(q)$, for $i \in \mathbb{N}$ and $q \in \mathsf{Loc}$, by the following recursive rules:

$$\begin{cases} \text{if } q \models P_2: \quad v_0(q) = 0 \\ \text{if } q \models \neg P_2: v_0(q) = +\infty \end{cases}$$

$$\begin{cases} \text{if } q \models P_2: \ v_{i+1}(q) = 0 \\ \text{if } q \models \neg P_1 \wedge \neg P_2: \ v_{i+1}(q) = +\infty \\ \text{otherwise}: v_{i+1}(q) = \min\limits_{c \in \mathsf{Mv}(q,A)} \ \max\limits_{\overline{c} \in \mathsf{Mv}(q,\overline{A})} \Big(\mathsf{Edg}_\tau(q, c\cdot\overline{c}) + v_i(\mathsf{Edg}_\ell(q, c\cdot\overline{c}))\Big) \end{cases}$$

Our proof now amounts to showing the following lemma:

Lemma 7. *For any $i \in \mathbb{N}$, for any $n \in \mathbb{N}$ and any $q \in \mathsf{Loc}$, we have:*

$$n \geq v_i(q) \quad \Leftrightarrow \quad q \models \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}^{\leq i}_{\leq n}\, P_2.$$

The proof is by induction on i: The base case is straightforward, as well as the induction step when $q \models P_2$ and when $q \models \neg P_1 \wedge \neg P_2$. We thus only focus on the last case, when $q \models P_1 \wedge \neg P_2$: Assume the induction hypothesis holds up to level i. If $n \geq v_{i+1}(q)$, then (by definition of $v_{i+1}(q)$) there exists a move $c \in \mathsf{Mv}(q, A)$ such that, for any move $\overline{c} \in \mathsf{Mv}(q, \overline{A})$, we have

$$n \geq \mathsf{Edg}_\tau(q, c \cdot \overline{c}) + v_i(\mathsf{Edg}_\ell(q, c \cdot \overline{c})).$$

By i.h., for any $\overline{c} \in \mathsf{Mv}(q, \overline{A})$, we have

$$\mathsf{Edg}_\ell(q, c \cdot \overline{c}) \models \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}^{\leq i}_{\leq n - \mathsf{Edg}_\tau(q, c \cdot \overline{c})}\, P_2.$$

The strategy σ_A witnessing that property, combined with the move $c \in \mathsf{Mv}(q, A)$, yields a strategy for enforcing $P_1 \,\mathbf{U}^{\leq i+1}_{\leq n}\, P_2$ from q, as required.

The converse implication follows the same lines: given a strategy σ_A enforcing $P_1 \,\mathbf{U}^{\leq i+1}_{\leq n}\, P_2$ from q, we let $c = \sigma_A(q)$, and deduce that n satisfies the same inequalities as above.

From Equation (1), it suffices to compute $v_{|\mathsf{Loc}|}(q)$, for each $q \in \mathsf{Loc}$, to deduce the set of locations where φ holds. This algorithm thus runs in time $O(|\mathsf{Loc}| \cdot |\mathsf{Edg}|)$. $\square$

The release modality is handled similarly: we define $v'_i(q)$ as follows:

$$\begin{cases} \text{if } q \models \neg P_2 : v'_0(q) = 0 \\ \text{if } q \models P_2 : \quad v'_0(q) = +\infty \end{cases}$$

$$\begin{cases} \text{if } q \models \neg P_2 : v'_{i+1}(q) = 0 \\ \text{if } q \models P_1 \wedge P_2 : \; v'_{i+1}(q) = +\infty \\ \text{otherwise} : \; v'_{i+1}(q) = \max\limits_{c \in \mathsf{Mv}(q,A)} \; \min\limits_{\overline{c} \in \mathsf{Mv}(q,\overline{A})} \Big(\mathsf{Edg}_\tau(q, c \cdot \overline{c}) + v'_i(\mathsf{Edg}_\ell(q, c \cdot \overline{c})) \Big) \end{cases}$$

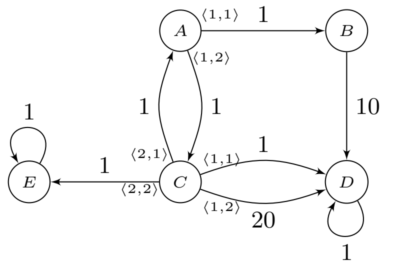
and $\mathbf{R}^{\leq i}_{\leq n}$ as the dual of $\mathbf{U}^{\leq i}_{\leq n}$. Then an equivalence similar to Equation (1) holds, and we have the following lemma (proof omitted):

Lemma 8. *For any $i \in \mathbb{N}$, for any $n \in \mathbb{N}$ and any $q \in \mathsf{Loc}$, we have:*

$$n < v'_i(q) \quad \Leftrightarrow \quad q \models \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{R}^{\leq i}_{\leq n}\, P_2.$$

Example 1. Consider the example depicted on Figure 1. On that TDCGS, the duration is the integer written in the middle of each transition. The tuples that are written close to the source location indicates the choices of the agents for firing that transition (for instance, $\langle 2, 1 \rangle$ means that player a_1 chooses move 2 and player a_2 chooses move 1). They are omitted when each agent has a single choice.

The valuations of atomic propositions are given in the table on the right of the figure. This table shows the computation of $v_i(q)$, for each location. This computation converges in three steps. For instance, that $v_3(A) = 21$ indicates that $A \models \langle\!\langle a_1 \rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq 21}\, P_2$ holds, but $A \not\models \langle\!\langle a_1 \rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq 20}\, P_2$.



$v_i(q)$	0	1	2	3
A (P_1, $\neg P_2$)	$+\infty$	$+\infty$	21	21
B (P_1, P_2)	0	0	0	0
C (P_1, $\neg P_2$)	$+\infty$	20	20	20
D ($\neg P_1$, P_2)	0	0	0	0
E (P_1, $\neg P_2$)	$+\infty$	$+\infty$	$+\infty$	$+\infty$

Fig. 1. The algorithm for $\mathbf{U}_{\leq\zeta}$.

2.2 Modalities $\mathbf{U}_{\geq\zeta}$ and $\mathbf{R}_{\geq\zeta}$

We now consider formula $\langle\!\langle A\rangle\!\rangle\, \varphi_1\, \mathbf{U}_{\geq\zeta}\, \varphi_2$ expressing that coalition A has a strategy for staying at least ζ time units in φ_1-states before reaching φ_2. We have:

Lemma 9. *Let $\mathcal{A}$ be a TDCGS, and $\varphi = \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}_{\geq\zeta}\, P_2$ be a* TATL *formula (with $P_1, P_2 \in \mathsf{AP}$). Then we can compute in time $O(|\mathsf{Loc}| \cdot |\mathsf{Edg}|)$ the set of locations of $\mathcal{A}$ where φ holds.*

Proof. The idea is similar to that of the proof of Lemma 6. We introduce the following modality:

$$\rho \models_{\mathcal{A}} p\, \mathbf{U}^{\geq i}\, q \quad \Leftrightarrow \quad \exists j \geq i.\ q_j \models_{\mathcal{A}} q \text{ and } q_k \models_{\mathcal{A}} p \text{ for any } 0 \leq k < j$$

We then compute a sequence of values, defined by the following recursive rules:

$$\begin{cases} \text{if } q \models \neg \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}\, P_2 :\ v_0(q) = -\infty \\ \text{if } q \models \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}\, P_2 \wedge \neg \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}^{\geq 1}\, P_2 :\ v_0(q) = 0 \\ \text{if } q \models \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}^{\geq 1}\, P_2 : v_0(q) = +\infty \end{cases}$$

$$\begin{cases} \text{if } q \models \neg \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}\, P_2 :\ v_{i+1}(q) = -\infty \\ \text{if } q \models \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}\, P_2 \wedge \neg \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}^{\geq 1}\, P_2 :\ v_{i+1}(q) = 0 \\ \text{if } q \models \langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}^{\geq 1}\, P_2 : \\ \qquad v_{i+1}(q) = \max\limits_{c \in \mathsf{Mv}(q,A)} \ \min\limits_{\overline{c} \in \mathsf{Mv}(q,\overline{A})} \Big(\mathsf{Edg}_\tau(q, c \cdot \overline{c}) + v_i(\mathsf{Edg}_\ell(q, c \cdot \overline{c}))\Big) \end{cases}$$

This computation requires that we first compute the set of locations satisfying $\langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}\, P_2$ and those satisfying $\langle\!\langle A\rangle\!\rangle\, P_1\, \mathbf{U}^{\geq 1}\, P_2$. This can be done in time $O(|\mathsf{Edg}|)$ using standard ATL model-checking algorithms. Thus, computing $v_i(q)$ for each $q \in \mathsf{Loc}$ and each $i \leq |\mathsf{Loc}|$ can be achieved in time $O(|\mathsf{Loc}| \cdot |\mathsf{Edg}|)$.

Those values satisfy the following lemma:

Lemma 10. *For any $i \in \mathbb{N}$, for any $n \in \mathbb{N}$ and $q \in \mathsf{Loc}$, we have*

$$n \leq v_i(q) \quad \Leftrightarrow \quad q \models \langle\!\langle A\rangle\!\rangle \big[(P_1\, \mathbf{U}_{\geq n}\, P_2) \vee (P_1\, \mathbf{U}^{\geq i+1}\, P_2)\big].$$

This will conclude the proof of Lemma 9, thanks to the following equivalence:

$$q \models \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{\geq n}\, P_2 \;\Leftrightarrow\; q \models \langle\!\langle A \rangle\!\rangle \big[(P_1 \,\mathbf{U}_{\geq n}\, P_2) \vee (P_1 \,\mathbf{U}^{\geq |\mathsf{Loc}|+1}\, P_2)\big].$$

This equivalence relies on the fact that all durations are strictly positive: if some outcome of the strategy satisfies $P_1 \,\mathbf{U}^{\geq |\mathsf{Loc}|+1}\, P_2$, then one location is visited twice, and it is possible to adapt the strategy so that it is visited n times (thus with total duration larger than n) before visiting P_2.

We now prove Lemma 10: we omit the easy cases, and only focus on the inductive step in the case when $q \models \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}^{\geq 1}\, P_2$. In particular, we have $q \models P_1$. First, pick some $n \leq v_{i+1}(q)$, assuming the result holds up to step i. By definition of $v_{i+1}(q)$, there exists $c \in \mathsf{Mv}(q, A)$ such that, for any $\overline{c} \in \mathsf{Mv}(q, \overline{A})$, we have

$$n \leq \mathsf{Edg}_\tau(q, c \cdot \overline{c}) + v_i(\mathsf{Edg}_\ell(q, c \cdot \overline{c})).$$

From the induction hypothesis, this means that

$$\mathsf{Edg}_\ell(q, c \cdot \overline{c}) \models \langle\!\langle A \rangle\!\rangle \big[(P_1 \,\mathbf{U}_{\geq n - \mathsf{Edg}_\tau(q, c \cdot \overline{c})}\, P_2) \vee (P_1 \,\mathbf{U}^{\geq i+1}\, P_2)\big].$$

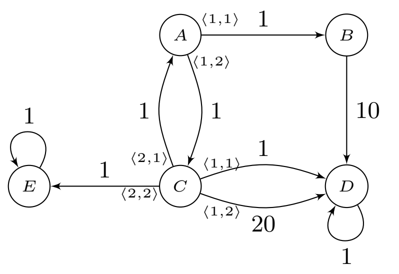
The strategy witnessing that property, combined with move $c \in \mathsf{Mv}(q, A)$, yields a strategy witnessing that

$$q \models \langle\!\langle A \rangle\!\rangle \big[(P_1 \,\mathbf{U}_{\geq n}\, P_2) \vee (P_1 \,\mathbf{U}^{\geq i+2}\, P_2)\big].$$

The converse implication follows the same (reversed) steps. □

We omit the case of the release modality, which is very similar.

Example 2. In Figure 2, we apply that algorithm to the same TDCGS as before. This table shows the computation of $v_i(q)$, for each location. This com-



$v_i(q)$	0	1	2	3
A (P_1, $\neg P_2$)	$+\infty$	$+\infty$	2	2
B (P_1, P_2)	$+\infty$	10	10	10
C (P_1, $\neg P_2$)	$+\infty$	1	1	1
D ($\neg P_1$, P_2)	0	0	0	0
E (P_1, $\neg P_2$)	$-\infty$	$-\infty$	$-\infty$	$-\infty$

Fig. 2. The algorithm for $\mathbf{U}_{\geq \zeta}$.

putation converges in three steps. For instance, that $v_3(A) = 2$ indicates that $A \models \langle\!\langle a_1 \rangle\!\rangle\, P_1 \,\mathbf{U}_{\geq 2}\, P_2$ holds, but $A \not\models \langle\!\langle a_1 \rangle\!\rangle\, P_1 \,\mathbf{U}_{\geq 3}\, P_2$.

2.3 Modalities $\mathbf{U}_{=\zeta}$ and $\mathbf{R}_{=\zeta}$

Lemma 11. *Let $\mathcal{A}$ be a TDCGS, and $\varphi = \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{=\zeta}\, P_2$ be a* TATL *formula. We can compute in time $O(\zeta \cdot |\mathsf{Edg}|)$ the set of locations of $\mathcal{A}$ where φ holds.*

Since ζ is encoded in binary, this algorithm runs in time exponential in the size of the formula.

Proof. We use dynamical programming, and recursively build a table $T\colon \mathsf{Loc} \times \{0, \ldots, \zeta\} \to \{\top, \bot\}$ such that

$$T(q, i) = \top \quad \Leftrightarrow \quad q \models_{\mathcal{A}} \langle\!\langle A \rangle\!\rangle\, \varphi_1 \,\mathbf{U}_{=i}\, \varphi_2. \tag{2}$$

When $i = 0$, letting $T(q, 0) = \top$ if, and only if, $q \models P_2$ clearly fulfills equation (2) (since all durations are non-zero). Now, pick $i < \zeta$, and assume all the $T(q, j)$ have been computed for $j \leq i$. Then

$$\begin{aligned} T(q, i+1) = \top \quad \Leftrightarrow \quad q \models P_1 \text{ and } \exists c \in \mathsf{Mv}(q, A).\ \forall \overline{c} \in \mathsf{Mv}(q, \overline{A}).\\ \mathsf{Edg}(q, c \cdot \overline{c}) = (q', t) \text{ with } T(q', i - t) = \top. \end{aligned}$$

This computation can be done since all durations are non-zero. It is achieved by running through the transition table, and is thus in time linear in the size of Edg. It is clear that equation (2) is preserved by this construction, so that in the end, $q \models \varphi$ iff $T(q, \zeta) = \top$. This algorithm runs in time $O(\zeta \times |\mathsf{Edg}|)$. $\square$

A similar algorithm can be defined for handling $\langle\!\langle A \rangle\!\rangle\, \varphi_1 \,\mathbf{R}_{=\zeta}\, \varphi_2$: the table $T'(q, i)$ is initialized in the same way (i.e., $T'(q, 0) = \top \;\Leftrightarrow\; q \models P_2$), and each step is computed according to the following rule:

$$\begin{aligned} T'(q, i) = \top \quad \Leftrightarrow \quad q \models P_1 \vee \exists c \in \mathsf{Mv}(q, A).\ \forall \overline{c} \in \mathsf{Mv}(q, \overline{A}).\\ \mathsf{Edg}(q, c \cdot \overline{c}) = (q', t) \text{ with } T'(q', i - t) = \top. \end{aligned}$$

2.4 Results for **TATL** and **TATL**$_{\leq,\geq}$

From Lemmas 6, 9 and 11, we can deduce procedures to handle all modalities and this gives the following result:

Theorem 12. *Model checking a* TATL *formula φ over a TDCGS $\mathcal{A}$ can be achieved in time $O(|\mathcal{A}|^2 \cdot |\varphi| \cdot \zeta_{\max})$, where $\zeta_{\max}$ is the maximal constant appearing in φ. It is thus in* EXPTIME*.*

Note that this algorithm is polynomial in the size of the TDCGS, and is exponential only because of the binary encoding of the constants that appear in the formula. This complexity blow-up cannot be avoided:

Theorem 13. *Model-checking* TATL *over TDCGSs is* EXPTIME*-complete.*

Proof. This result is based on the fact that deciding the *countdown games* is $\mathsf{EXPTIME}$-hard [15]. A countdown game is a two-player game. It consists of a weighted graph (V, E) where V is the set of vertices and $E \subseteq V \times \mathbb{N} \times V$ is the weighted transition relation. A configuration of a countdown game (V, E) is a pair $(v, C) \in V \times \mathbb{N}$. At every turn, Player 1 chooses, from the current configuration (v, C), a duration $1 \le d \le C$ s.t. (1) $0 < d \le C$ and (2) there exist at least one transition $(v, d, v') \in E$. Then Player 2 chooses one of these transitions (issued from v and whose duration is d) and then the new configuration is $(v', C-d)$. Any configuration $(v, 0)$ is terminal and it is a winning configuration for Player 1. Any configuration (v, C) s.t. (1) $C > 0$ and (2) there is no transition $(v, d, -)$ with $d \le C$ is terminal and it is a winning configuration for Player 2. Deciding whether Player 1 has a wining strategy for a configuration (v, C) is an $\mathsf{EXPTIME}$-hard problem [15].

We can easily build a TDCGS $\mathcal{A} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg} \rangle$ corresponding to the countdown game (V, E). We let $\mathsf{Agt} = \{a_1, a_2\}$. The set of locations $\mathsf{Loc} \subseteq V \cup V \times \mathbb{N}$ is defined as follows: $v \in \mathsf{Loc}$ if $v \in V$, and $(v, t) \in \mathsf{Loc}$ if there exists a transition $(v, t, -)$ in E. This is a turn-based game: agent a_1 plays (i.e., has several possible moves) in locations v, and agent a_2 plays in locations (v, t). From v, the moves of agent a_1 are the weights t s.t. there exist some transitions $(v, t, -)$ in E, and the moves of agent a_2 from (v, t) are the possible successors v' s.t. $(v, t, v') \in E$. The transition table is defined in a natural way and the duration associated with transitions leading from v to (v, t) or from (v, t) to v' is t. Then deciding whether the configuration (v, C) with $C \in \mathbb{N}$ is winning for Player 1 reduces to a model checking problem: $v \models_{\mathcal{A}} \langle\!\langle a_1 \rangle\!\rangle \, \mathbf{F}_{=2C} \top$. $\square$

Still, we can have efficient algorithm if we restrict to the fragment $\mathsf{TATL}_{\le,\ge}$:

Theorem 14. *Model-checking a* $\mathsf{TATL}_{\le,\ge}$ *formula* φ *over a TDCGS* $\mathcal{A}$ *can be achieved in time* $O(|\mathcal{A}|^2 \cdot |\varphi|)$*, and is thus in* PTIME.

This is an immediate consequence of Lemmas 6 and 9. PTIME-hardness follows from that of CTL model-checking over Kripke structures. Thus:

Corollary 15. *Model checking* $\mathsf{TATL}_{\le,\ge}$ *over TDCGSs is* PTIME*-complete.*

2.5 Unitary TDCGSs

Unitary TDCGSs are TDCGSs where all durations equal 1. Intuitively, unitary TDCGSs are easier to handle because it is possible to verify $\varphi = \langle\!\langle A \rangle\!\rangle \, \varphi_1 \, \mathbf{U}_{=c} \, \varphi_2$ by dichotomy, i.e., by verifying $\langle\!\langle A \rangle\!\rangle \, \varphi_1 \, \mathbf{U}_{=\lfloor c/2 \rfloor} \, (\langle\!\langle A \rangle\!\rangle \, \varphi_1 \, \mathbf{U}_{=c-\lfloor c/2 \rfloor} \, \varphi_2)$, and so on [14]. That way, model-checking an $\mathbf{U}_{=\zeta}$ formula can be achieved in time $O(\log(\zeta) \cdot |\mathcal{A}|)$. In the end:

Theorem 16. *Model checking a* TATL *formula* φ *over a unitary TDCGS* $\mathcal{A}$ *can be achieved in time* $O(|\mathcal{A}|^2 \cdot |\varphi|)$*, and is thus* PTIME*-complete.*

3 Durational CGSs

We now propose an extension of the models above that now allows transitions labeled with intervals (instead of a single integer). That way, agents don't know in advance the duration of the transitions. Special agents, one per transition, decide for the durations. This gives a very expressive framework, in which coalitions can mix "classical" agents with "time" agents.

3.1 Definition

We write $\mathcal{I}$ for the set of intervals with bounds in $\mathbb{N}^{>0} \cup \{+\infty\}$.

Definition 17. *A durational CGS (DCGS) is a* 6*-tuple* $\mathcal{S} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg} \rangle$ *such that:*

- Loc*,* Agt*,* AP*,* Lab *and* Mv *have the same characteristics as in Definition 1;*
- $\mathsf{Edg}\colon \mathsf{Loc} \times \mathbb{N}^k \to \mathsf{Loc} \times \mathcal{I}$ *is the transition table, associating with each transition an interval containing its possible durations.*

The size of the transition table is again the space needed to write it in binary notation, and the size of a DCGS is $|\mathsf{Loc}| + |\mathsf{Edg}|$. Again, we use $\mathsf{Edg}_\tau(q, c)$ to denote the interval of durations of the transition $\mathsf{Edg}(q, c)$, and $\mathsf{Edg}_\ell(q, c)$ to denote its location.

While the syntax is not very different to that of TDCGSs, the semantics is rather more involved: the crucial point is that the agents must select the transition the system will fire, but they must also choose the duration of that transition within the interval it's labeled with. This last part is achieved by special "time-agents": we consider one time-agent $ta_{q,c}$ per location q and move c in $\mathsf{Mv}(q, \mathsf{Agt})$. Formally the semantics of $\mathcal{S}$ is defined as a TDCGS $\mathcal{A}[\mathcal{S}] = \langle \mathsf{Loc}, \mathsf{Agt}', \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}', \mathsf{Edg}' \rangle$ with:

- $\mathsf{Agt}' = \mathsf{Agt} \cup \{ta_{q,c} \mid q \in \mathsf{Loc} \text{ and } c \in \mathsf{Mv}(q, \mathsf{Agt})\}$,
- $\mathsf{Mv}'(q, a) = \mathsf{Mv}(q, a)$ for any $a \in \mathsf{Agt}$; $\mathsf{Mv}'(q, ta_{q,c}) = \mathsf{Edg}_\tau(q, c)$ for any $ta_{q,c} \in \mathsf{Agt}'$, and $\mathsf{Mv}'(q, ta_{q',c}) = \{0\}$ for any $ta_{q,c} \in \mathsf{Agt}'$ with $q' \neq q$,
- $\mathsf{Edg}'(q, c, t_{q_0,c_0}, \ldots, t_{q_n,c_m}) = (q', t)$ iff $c \in \mathsf{Mv}'(q, \mathsf{Agt})$ for any i, $t = t_{q,c}$ and $t_{q,c} \in \mathsf{Mv}'(q, ta_{q,c})$, and $t_{q',c'} = 0$ when $q' \neq q$ or $c \neq c'$.

As for TDCGSs, we use the notions of coalition of agents (including the time-agents), strategy and outcome.

Note that the transition table of the corresponding TDCGS $\mathcal{A}[\mathcal{S}]$ is infinite when there exist infinite intervals in the definition of $\mathcal{S}$. And when it is finite, $|\mathsf{Edg}'|$ is bounded by $|\mathsf{Edg}| \cdot b_M$ where b_M is the maximal constant occurring in the intervals of durations in $\mathcal{S}$: in Edg', we replace each entry (q, c) of Edg by $(b - a) + 1$ entries when $\mathsf{Edg}_\tau(q, c) = [a; b]$. Thus the size of $\mathcal{A}[\mathcal{S}]$ is potentially exponential in $|\mathcal{S}|$ (due to the binary encoding). Note that in every entry of Edg', only one time-agent may have more than one possible move.

Moreover time agents can be used in TATL modalities in order to express the existence of strategies for coalitions that may control the duration of a subset

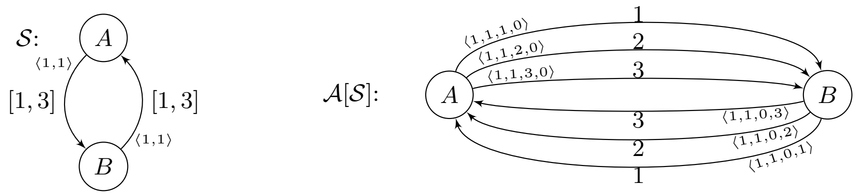


Fig. 3. A DCGS for the simplified Nim game, and its associated TDCGS.

of transitions. Given a DCGS $\mathcal{S}$, a location q, and a TATL formula φ, we write $q \models_{\mathcal{S}} \varphi$ when $q \models_{\mathcal{A}[\mathcal{S}]} \varphi$. We might omit the subscript if it raises no ambiguity.

Example 3. We illustrate our models by a simple example: the simplified Nim game. In that game, a set of N matches are aligned on a table, and each player, in turn, picks between 1 and 3 matches. The players who takes the last match is declared the winner. This game can easily be encoded as the DCGS (where "durations" are in fact the number of matches taken by the players) depicted on the left of Figure 3. Player A wins iff formula $\langle\!\langle A, t_A \rangle\!\rangle\, \mathbf{F}_{=N}\, B$ holds.

Time agents. The motivation for using time-agents is that the time elapsing should not be controlled by the same player along an execution. Depending on the state or the transition, it is convenient to be able to specify who decides the duration of an event. Note that assigning one time-agent per transition is more general than assigning one time-agent per location: indeed, in the former approach, a time-agent ta_q (for controlling the duration of all the transitions issued from q) can be easily simulated by the coalition $\{ta_{q,c_1}, \ldots, ta_{q,c_m}\}$ containing all the time-agents of the transitions leaving q.

3.2 Model checking $\mathsf{TATL}_{\leq,\geq}$

When verifying TATL formulae containing modalities with timing constraints of the form "$\leq c$" or "$\geq c$", we do not need to consider all the transitions of $\mathcal{A}[\mathcal{S}]$. We can restrict the analysis to an abstraction of $\mathcal{A}[\mathcal{S}]$:

Definition 18. *Let* $\mathcal{S} = \langle \mathsf{Loc}, \mathsf{Agt}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}, \mathsf{Edg}\rangle$ *be a DCGS and let* $\mathcal{A}[\mathcal{S}] = \langle \mathsf{Loc}, \mathsf{Agt}', \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}', \mathsf{Edg}'\rangle$ *be the TDCGS corresponding to the semantics of* $\mathcal{S}$*. Given an integer* C*, we define the* C*-abstraction of* $\mathcal{S}$ *as the TDCGS* $\mathcal{A}[\mathcal{S}]_C = \langle \mathsf{Loc}, \mathsf{Agt}', \mathsf{AP}, \mathsf{Lab}, \mathsf{Mv}'', \mathsf{Edg}''\rangle$ *with:*

- $\mathsf{Mv}''(q, ta_{q,c})$ *is* $\{a, b\}$ *(resp.* $\{a; C{+}1\}$*) if* $\mathsf{Edg}_\tau(q, c) = [a, b]$ *(resp.* $\mathsf{Edg}_\tau(q, c) = [a, +\infty)$*); and* Mv'' *coincides with* Mv' *for other cases (*$a \in \mathsf{Agt}$ *or* $ta_{q',c}$ *with* $q' \neq q$*).*
- Edg'' *is defined as* Edg' *but with* Mv'' *instead of* Mv'*.*

In the TDCGS $\mathcal{A}[\mathcal{S}]_C$, we replace the set of transitions corresponding to all durations in an interval λ by two transitions: a short one —the left-end value of λ— and a long one: either the right-end value of λ if λ is finite, or $C+1$ (or the left-end of λ if $C+1 \notin \lambda$). Indeed, an open interval is interesting for the truth value of some properties because it may allow arbitrary long durations. But delaying for $C+1$ t.u. is always enough when considering $\mathsf{TATL}_{\leq,\geq}$ formulae with constants less than C:

Lemma 19. *Let $\mathcal{S}$ be the DCGS and C be an integer. For any $\zeta \leq C$ and $q \in \mathsf{Loc}$, we have:*

$$q \models_{\mathcal{A}[\mathcal{S}]} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq\zeta}\, P_2 \quad\Leftrightarrow\quad q \models_{\mathcal{A}[\mathcal{S}]_C} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{\leq\zeta}\, P_2 \tag{3}$$
$$q \models_{\mathcal{A}[\mathcal{S}]} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{\geq\zeta}\, P_2 \quad\Leftrightarrow\quad q \models_{\mathcal{A}[\mathcal{S}]_C} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{U}_{\geq\zeta}\, P_2 \tag{4}$$
$$q \models_{\mathcal{A}[\mathcal{S}]} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{R}_{\leq\zeta}\, P_2 \quad\Leftrightarrow\quad q \models_{\mathcal{A}[\mathcal{S}]_C} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{R}_{\leq\zeta}\, P_2 \tag{5}$$
$$q \models_{\mathcal{A}[\mathcal{S}]} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{R}_{\geq\zeta}\, P_2 \quad\Leftrightarrow\quad q \models_{\mathcal{A}[\mathcal{S}]_C} \langle\!\langle A \rangle\!\rangle\, P_1 \,\mathbf{R}_{\geq\zeta}\, P_2 \tag{6}$$

Proof (sketch). There are more behaviors in $\mathcal{A}[\mathcal{S}]$ than in $\mathcal{A}[\mathcal{S}]_C$, but these additional executions do not change the truth value of $\mathsf{TATL}_{\leq,\geq}$ formulae.

Indeed let $\rho = (q_0, d_0)\ldots(q_i, d_i)\ldots$ be an execution in $\mathcal{A}[\mathcal{S}]$ and let c_i^a (resp. c_i^t) be the $i+1$-st move of the agents Agt (resp. the time-agents) along ρ [2]. We can change the move of the agent ta_{q_i,c_i} and obtain another run ρ' with the *same prefix and the same suffix* as ρ: the duration spent in q_i has changed (and thus the global dates of actions) but not the time spent in other locations. This property allows to make local changes on delays without changing the sequence of visited states.

Consider a strategy σ_A for the coalition A in $\mathcal{A}[\mathcal{S}]$ to ensure $\psi = \mathsf{P}_1 \,\mathbf{U}_{\leq\zeta}\, P_2$. From σ_A, we can build a strategy σ'_A ensuring ψ in $\mathcal{A}[\mathcal{S}]_C$. Indeed the only changes we have to make are for the moves of time-agents when, in q with a move c for Agt, σ_A requires to wait for $t_{q,c}$ while $t_{q,c}$ is not in the restricted set of moves of $\mathcal{A}[\mathcal{S}]_C$ (*i.e.* $t_{q,c} \notin \mathsf{Mv}''(q, ta_{q,c})$). In that case, the strategy σ'_A can propose the minimal duration in $\mathsf{Mv}''(q, ta_{q,c})$: the ending state satisfying ψ will be reached sooner than along ρ and then ψ will be true. If the formula to be verified was $\mathbf{U}_{\geq c}$ then the maximal duration will be enough to ensure the formula. The same holds for the release operators.

Now consider a strategy σ'_A for the coalition A in $\mathcal{A}[\mathcal{S}]_C$ to ensure $\psi = \mathsf{P}_1 \,\mathbf{U}_{\leq\zeta}\, P_2$. This strategy can be completed for ensuring ψ in the full TDCGS. Consider a finite execution ρ in $\mathcal{A}[\mathcal{S}]$, we can define a corresponding execution $\bar{\rho}$ in $\mathcal{A}[\mathcal{S}]_C$ where any move of time-agent $ta_{q,c}$ from the location q is either left unchanged if $ta_{q,c} \in A$ —this is an "A-controllable" time-agent and having applied σ_A from the beginning of the execution ensures that its move is in $\mathcal{A}[\mathcal{S}]_C$—, or replaced by the maximal duration in $\mathsf{Mv}''(q, ta_{q,c})$ if $ta_{q,c}$ is not an A-controllable time-agent. Then it is sufficient to define $\sigma_A(\rho)$ as $\sigma'_A(\bar{\rho})$.

Of course, if we consider $\psi = \mathsf{P}_1 \,\mathbf{U}_{\leq\zeta}\, P_2$, we build $\bar{\rho}$ differently and consider minimal durations. □

Note that the size of $\mathcal{A}[\mathcal{S}]_C$ is bounded by $2 \cdot |\mathcal{S}|$. Thus:

Theorem 20. *Model checking $\mathsf{TATL}_{\leq,\geq}$ over DCGSs is* PTIME*-complete.*

[2] *i.e.* $(q_{i+1}, t_{i+1} - t_i) = \mathsf{Edg}'(q_i, c_i^a \cdot c_i^t)$ with $c_i^a \cdot c_i^t \in \mathsf{Mv}(q_i, \mathsf{Agt}')$.

3.3 Model checking TATL

For full TATL, we also reduce the problem to that of finite TDCGSs. Given a DCGS $\mathcal{S}$ and a formula $\varphi = \langle\!\langle A \rangle\!\rangle\, P_1\, \mathbf{U}_{=\zeta}\, P_2$, we explicitly add one extra agent per transition, with moves in $[a, b]$ if the corresponding transition is labeled with $[a, b]$, and moves in $[a, \max(a, \zeta + 1)]$ if it is labeled with $[a, +\infty)$. This restriction makes the corresponding TDCGS to be finite, its size is in $O(|\mathcal{S}| \cdot \max(b_M, \zeta))$ where b_M is the maximal integer appearing as a bound of an interval in $\mathcal{S}$. And applying the algorithm of Theorem 12 to that TDCGS, we get an EXPTIME algorithm for model-checking φ on our DCGS $\mathcal{S}$. The algorithm is similar for the release modality. This must be repeated a polynomial number of times for verifying a TATL formula, yielding an algorithm in time $O(|\mathcal{S}|^2 \cdot b_M \cdot \zeta_{\max} \cdot |\varphi|)$ (where $\zeta_{\max}$ is the largest constant in the formula φ), that is, in time exponential in both the structure and the formula. Note that the blow-up is only due to the binary notation for the integers. Combined to Theorem 13, this gives:

Theorem 21. *Model-checking* TATL *over DCGSs is* EXPTIME*-complete.*

Conclusion

We have introduced a new family of models of concurrent game structures, in which transitions carry a (set of) durations. The semantics of those models involve "time-agents", i.e., agents that decide the duration of the transition that will be fired. This allows to break the symmetry in time games, allowing some coalition to decide the duration of *some* of the transitions.

We proved that those models enjoy efficient quantitative model-checking algorithms, as soon as no timing constraint $= \zeta$ is involved. Equality constraints yield an exponential blow-up, which we saw cannot be avoided.

As future work, we plan to extend this "time agent"-semantics to concurrent game structures with clocks, similar to timed automata [2]. As a first step, it would be nice if we could extend the algorithms presented here to the "continuous" semantics (as defined in [17]) of DCGSs.

References

1. R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking in dense real-time. *Information & Computation*, 104(1):2–34, 1993.
2. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.
3. R. Alur and T. A. Henzinger. Modularity for timed and hybrid systems. In *Proc. 8th Intl Conf. on Concurrency Theory (CONCUR'97)*, volume 1243 of *LNCS*, pages 74–88. Springer, 1997.
4. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. In *Proc 38th Annual Symp. on Foundations of Computer Science (FOCS'97)*, pages 100–109. IEEE Comp. Soc. Press, 1997.
5. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

6. E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *Proc IFAC Symp. on System Structure and Control*. Elsevier, 1998.
7. B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools*. Springer, 2001.
8. R. E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, 1992.
9. S. Campos and E. M. Clarke. Real-time symbolic model checking for discrete time models. In *Theories and Experiences for Real-Time System Development*, volume 2 of *AMAST Series in Computing*, pages 129–145. World Scientific, 1995.
10. F. Cassez, A. David, E. Fleury, K. G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In M. Abadi and L. de Alfaro, editors, *Proc. 16th Intl Conf. on Concurrency Theory (CONCUR'05)*, volume 3653 of *LNCS*, pages 66–80. Springer, Aug. 2005.
11. E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proc. Workshop Logics of Programs 1981*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.
12. E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.
13. L. de Alfaro, M. Faella, T. A. Henzinger, R. Majumdar, and M. Stoelinga. The element of surprise in timed games. In R. Amadio and D. Lugiez, editors, *Proc. 14th Intl Conf. on Concurrency Theory (CONCUR'03)*, volume 2761 of *LNCS*, pages 142–156. Springer, Aug. 2003.
14. E. A. Emerson, A. K.-L. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. *Real-Time Systems*, 4:331–352, 1992.
15. M. Jurdziński. Countdown games, Mar. 2006. Personal communication.
16. F. Laroussinie, N. Markey, and G. Oreiby. Expressiveness and complexity of ATL. Research Report LSV-06-03, Laboratoire Spécification et Vérification, ENS Cachan, France, Feb. 2006.
17. F. Laroussinie, N. Markey, and Ph. Schnoebelen. Efficient timed model checking for discrete-time systems. *Theoretical Computer Science*, 353(1-3):249–271, 2006.
18. O. Maler, A. Pnueli, and J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proc. 12th Annual Symp. on Theoretical Aspects of Computer Science (STACS'95)*, volume 900 of *LNCS*, pages 229–242. Springer, 1995.
19. N. Markey and Ph. Schnoebelen. Symbolic model checking of simply-timed systems. In Y. Lakhnech and S. Yovine, editors, *Proc. Joint Conf. Formal Modelling and Analysis of Timed Systems (FORMATS'04) and Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'04)*, volume 3253 of *Lecture Notes in Computer Science*, pages 102–117, Grenoble, France, Sept. 2004. Springer.
20. K. L. McMillan. *Symbolic Model Checking: An Approach to the State Explosion Problem*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1992.
21. A. Pnueli. The temporal logic of programs. In *Proc. 18th IEEE Symp. Foundations of Computer Science (FOCS'77)*, pages 46–57. IEEE Comp. Soc. Press, 1977.
22. J.-P. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. 5th Intl Symp. on Programming*, volume 137 of *LNCS*, pages 337–351. Springer, 1982.
23. P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, 1989.
24. P.-Y. Schobbens and Y. Bontemps. Real-time concurrent game structures, Dec. 2005. Personal communication.