



**HAL**  
open science

**Resource-bounded ATL: the Quest for Tractable  
Fragments En quête de fragments mécanisables pour  
ATL avec ressources Keywords Logics for agents and  
multi-agent systems, verification techniques for  
multi-agent systems, model-checking, vec- tor addition  
systems with states**

Francesco Belardinelli, Stephane Demri

► **To cite this version:**

Francesco Belardinelli, Stephane Demri. Resource-bounded ATL: the Quest for Tractable Fragments En quête de fragments mécanisables pour ATL avec ressources Keywords Logics for agents and multi-agent systems, verification techniques for multi-agent systems, model-checking, vec- tor addition systems with states. Conférence Nationale en Intelligence Artificielle, Jul 2019, Toulouse, France. hal-02328812

**HAL Id: hal-02328812**

**<https://hal.science/hal-02328812v1>**

Submitted on 23 Oct 2019

**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.

# Resource-bounded ATL : the Quest for Tractable Fragments

## En quête de fragments mécanisables pour ATL avec ressources

Francesco Belardinelli<sup>1</sup>

Stéphane Demri<sup>2</sup>

<sup>1</sup> Department of Computing, Imperial College London & Laboratoire IBISC, Université d'Evry

<sup>2</sup> LSV, CNRS, ENS Paris-Saclay, Université Paris-Saclay

francesco.belardinelli@imperial.ac.uk

demri@lsv.fr

### Résumé

*Dans ce travail, nous commençons par présenter un état de l'art des résultats sur le problème de model-checking en relation avec la logique  $RB\pm ATL$ , qui est une version de ATL avec ressources. Cela nous permet d'identifier plusieurs problèmes ouverts et d'établir des relations avec les logiques à la  $RBTL$ , lorsque  $RB\pm ATL$  est restreinte à un unique agent. Ensuite, nous montrons que le problème de model-checking pour  $RB\pm ATL$  restreinte à un agent et à une ressource est PTIME-complet. Pour ce faire, nous faisons un léger détour en passant par les systèmes d'addition de vecteurs avec états. Nous prouvons de nouveaux résultats de complexité pour l'accessibilité d'un état de contrôle et pour la non-terminaison, lorsqu'un seul compteur est autorisé. Cet article a été présenté à la conférence AAMAS'19, Montréal, mai 2019.*

### Mots Clef

Logiques pour systèmes multi-agent, model-checking, systèmes d'addition de vecteurs avec états, complexité.

### Abstract

*In this work, we begin by providing a general overview of the model-checking results currently available for the Resource-bounded Alternating-time Temporal Logic  $RB\pm ATL$ . This allows us to identify several open problems in the literature, as well as to establish relationships with  $RBTL$ -like logics, when  $RB\pm ATL$  is restricted to a single agent. Then, we show that model checking  $RB\pm ATL$  is PTIME-complete when restricted to a single agent and a single resource. To do so, we make a valuable detour on vector addition systems with states, by proving new complexity results for their state-reachability and nontermination problems, when restricted to a single counter. **This paper has been presented at the conference AAMAS'19, Montréal, May 2019.***

### Keywords

Logics for agents and multi-agent systems, verification techniques for multi-agent systems, model-checking, vector addition systems with states.

## 1 Introduction

In recent years, logic-based languages for specifying the strategic behaviours of agents in multi-agent systems have been the object of increasing interest. A wealth of logics for strategies have been proposed in the literature, including Alternating-time Temporal Logic [AHK02], possibly with strategy contexts [LM15], Coalition Logic [Pau02], Strategy Logic [CHP07, MMPV14], among others. The expressive power of these formalisms has been thoroughly studied, as well as the corresponding verification problems, thus leading to model checking tools for game structures and multi-agent systems [CLMM14, LQR15, AdAG<sup>+</sup>01, KNN<sup>+</sup>08].

It is worth noticing that the computational models underlying these logic-based languages share a common feature : actions are normally modelled as abstract objects (typically a labelling on transitions) that bear no computational cost. However, if logics for strategies are to be applied to concrete multi-agent systems of interest, it is key to account for the resources actions might consume or produce. These considerations have prompted recently investigations in resource-aware logics for strategies. Obviously, there is a long tradition in resource-aware logics that dates back at least to substructural and linear logics (see e.g. [POY04]). More specifically, in this paper we follow the line of *Resource-bounded Alternating-time Temporal Logics* [ALNR14, ABLN15, ALN<sup>+</sup>15, ABLN17, AL18, ABDL18], which are characterised by two main features : firstly, actions in concurrent game structures are endowed with (positive/negative) costs ; and secondly, the standard strategy operators of Alternating-time Temporal Logic (ATL) are indexed by tuples of natural numbers, intuitively representing the resource budget available to agents in the coalition. This account has proved successful in the modelling and verification of a number of multi-agent scenarios, where reasoning about resources is critical [ABLN17].

Our motivation for the present contribution is threefold. First of all, in the literature there are several gaps in the results available for the decidability and complexity of the related model checking problem. For instance, if we assume

two resources and two agents in our multi-agent system, then model checking is known to be PSPACE-hard and in EXPTIME, but no tight complexity result is available. Our long-term aim is to fill all such gaps eventually. Further, while completing this picture, it is of interest to identify model checking instances that are tractable. Although the notion of tractable problem is open to discussion, in the context of strategy and temporal logics a model checking problem decidable in polynomial time (in the size of the formula and model) falls certainly within the description. Finally, complexity results for Resource-bounded ATL appear disseminated in a number of references, and are proved by using a wealth of different techniques, thus hindering a clear vision of the state of the art. We aim at developing a unified framework based on general proof techniques. Vector addition systems with states (VASS) are key in this respect [KM69].

Our contribution in this paper is also threefold. Firstly, we give an overview of the complexity results currently available for both RB±ATL and RB±ATL\*, the two most significant flavours of Resource-bounded ATL. This allows us to point out that, while for RB±ATL\* we have tight complexity results for any number of resources and agents, in RB±ATL there are still several open problems, whose solution is not apparent. Secondly, we extend current model checking results for RB±ATL to a more expressive language including the release operator R too. Thirdly, we prove that model checking RB±ATL is PTIME-complete, when we reason about a single resource and a single agent. Since we show that this setting is tantamount to the Computation Tree Logic CTL with a single resource, our result means that we can reason about resources in CTL at no extra computational cost. Most interestingly, to prove this main contribution we establish new complexity results for the state-reachability and nontermination problems in VASS with a single counter. The latter can be seen as self-standing contributions in formal methods.

**Structure of the paper.** In Sect. 2, we present background notions on resource-bounded concurrent game structures and ATL-like logics. In Sect. 2.3, we show that the resource-bounded logics RBTL\* and RB±ATL\* restricted to a single agent have the same expressive power. In Sect. 3, we prove the main theoretical contributions of the paper. Specifically, in Sect. 3.1 we review the state of the art on model checking RB±ATL. Then, in Sect. 3.2 we show that the state-reachability and nontermination problems for VASS with a single counter are decidable in PTIME. Finally, in Sect. 3.3 we leverage on our new results for 1-VASS to prove that model checking RB±ATL with a single agent and a single resource is PTIME-complete. Sect. 4 concludes the paper, discusses the complexity of RB±ATL\* fragments, and evokes directions for future work.

**This paper has been presented at the conference AAMAS'19, Montréal, May 2019.**

## 2 Preliminaries

Below, we introduce preliminary notions on models for resource-bounded logics, as well as the logical languages themselves. Our presentation follows closely [ABDL18]. In the rest of the paper,  $\mathbb{N}$  (resp.  $\mathbb{Z}$ ) is the set of natural numbers (resp. integers) and  $[m, m']$  with  $m, m' \in \mathbb{Z}$  is the set  $\{j \in \mathbb{Z} \mid m \leq j \leq m'\}$ . For a finite or infinite sequence  $u \in X^\omega \cup X^*$  of elements in some set  $X$ , we write  $u_i$  for the  $(i + 1)$ -th element of  $u$ , i.e.,  $u = u_0u_1 \dots$ . For  $i \geq 0$ ,  $u_{\leq i}$  is the prefix of  $u$  of length  $i + 1$ , i.e.,  $u_{\leq i} = u_0u_1 \dots u_i$  and  $u_{\geq i}$  is the suffix of  $u$  defined as  $u_{\geq i} = u_iu_{i+1} \dots$ . The length of a finite or infinite sequence  $u \in X^\omega \cup X^*$  is denoted as  $|u|$ , where  $|u| = \omega$  for  $u \in X^\omega$ .

### 2.1 Resource-bounded CGS

Resource-bounded CGS are concurrent game structures [AHK02] enriched with counters and a cost function that assigns a cost (either positive or negative) to every action, thus updating the values of the counters as the system executes. Hereafter we follow closely [ABDL18] and assume a countably infinite set  $AP$  of propositional variables (or atoms).

**Definition 1 (RB-CGS)** A resource-bounded concurrent game structure is a tuple  $M = \langle Ag, S, Act, r, act, cost, \delta, L \rangle$  such that :

- $Ag$  is a finite, non-empty set of agents (by default  $Ag = [1, k]$  for some  $k \geq 1$ );
- $S$  is a finite, non-empty set of states  $s, s', \dots$ ;
- $Act$  is a finite, non-empty set of actions with a distinguished action `idle`;
- $r \geq 1$  is the number of resources;
- $act : S \times Ag \rightarrow \wp(Act) \setminus \{\emptyset\}$  is the protocol function, such that for all  $s \in S$  and  $a \in Ag$ , `idle`  $\in act(s, a)$ ;
- $cost : S \times Ag \times Act \rightarrow \mathbb{Z}^r$  is the (partial) cost function; that is,  $cost(s, a, \mathbf{a})$  is defined only when  $\mathbf{a} \in act(s, a)$ , and moreover, we assume that  $cost(s, a, \text{idle}) = \mathbf{0}$ ;
- $\delta : S \times (Ag \rightarrow Act) \rightarrow S$  is the (partial) transition function such that  $\delta$  is defined for state  $s$  and map  $f : Ag \rightarrow Act$  only if for every agent  $a \in Ag$ ,  $f(a) \in act(s, a)$ ;
- $L : AP \rightarrow \wp(S)$  is the labelling function.

Intuitively, a resource-bounded CGS describes the interactions of a group  $Ag$  of agents, who are able to perform the actions in  $Act$  according to the protocol function  $act$ . The execution of a joint action entails a transition in the system, as specified by the function  $\delta$ . Moreover, on each transition the values of the  $r$  resources are updated according to the cost of the joint action. The `idle` action is introduced in [ALNR14, ALNR17] and it is often advantageous in terms of computational complexity (see e.g. [ABLN17, ABDL18] or Section 3). An RB-CGS  $M$  is finite whenever  $L$  is restricted to a finite subset of  $AP$ . The

size  $|M|$  of a finite  $M$  is the size of its encoding when integers are encoded in binary and, maps and sets are encoded in extension using a reasonably succinct encoding.

Given a coalition  $A \subseteq Ag$  and state  $s \in S$ , a *joint action available to  $A$  in  $s$*  is a map  $f : A \rightarrow Act$  such that for every agent  $a \in A$ ,  $f(a) \in act(s, a)$ . The set of all such joint actions is denoted as  $D_A(s)$ . Given a state  $s \in S$ , the set of joint actions available to  $Ag$  is simply denoted as  $D(s)$ , and the function  $\delta$  is defined only for such joint actions. We write  $f \sqsubseteq g$  if  $Dom(f) \subseteq Dom(g)$ , and for every agent  $a \in Dom(f)$ ,  $g(a) = f(a)$ . Given a joint action  $f \in D_A(s)$ , we write  $out(s, f)$  to denote the set of *immediate outcomes* :

$$\{s' \in S \mid \text{for some } g \in D(s), f \sqsubseteq g \text{ and } s' = \delta(s, g)\}.$$

Further, given a joint action  $f \in D_A(s)$  and a state  $s$ , the cost of a transition from  $s$  by  $f$  (w.r.t. coalition  $A$ ) is defined as

$$cost_A(s, f) \stackrel{\text{def}}{=} \sum_{a \in A} cost(s, a, f(a)).$$

A *computation*  $\lambda$  is a finite or infinite sequence  $s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \dots$  such that for all  $0 \leq i < |\lambda| - 1$  we have  $s_{i+1} = \delta(s_i, f_i)$ .

## 2.2 The logics $RB\pm ATL^*$ and $RB\pm ATL$

To specify the strategic properties of agents in resource-bounded CGS, we present the logics  $RB\pm ATL^*$  and its fragment  $RB\pm ATL$ , which are extensions of  $ATL^*$  and  $ATL$  respectively, introduced in [ALNR14, ALN<sup>+</sup>15] to explicitly account for the production and consumption of resources by agents. Once more, in the presentation of  $RB\pm ATL^*$  and  $RB\pm ATL$  we follow [ABDL18].

**Syntax.** Given a finite set  $Ag$  of agents and a number  $r \geq 1$  of resources, we write  $RB\pm ATL^*(Ag, r)$  to denote the resource-bounded logic with agents from  $Ag$  and  $r$  resources, whose models are resource-bounded CGS with the same parameters.

**Definition 2 ( $RB\pm ATL^*$ )** *The state-formulas  $\phi$  and path-formulas  $\psi$  in  $RB\pm ATL^*(Ag, r)$  are built according to the following BNF :*

$$\begin{aligned} \phi & ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle A^{\vec{b}} \rangle\rangle\psi \\ \psi & ::= \phi \mid \neg\psi \mid \psi \wedge \psi \mid X\psi \mid \psi U \psi, \end{aligned}$$

where  $p \in AP$ ,  $A \subseteq Ag$ , and  $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$ . The formulas in  $RB\pm ATL^*(Ag, r)$  are understood as the state-formulas.

Clearly,  $RB\pm ATL^*$  extends  $ATL^*$  by indexing the strategic operator  $\langle\langle A \rangle\rangle$  with tuple  $\vec{b}$ , whose intuitive meaning is that the coalition  $A$  can achieve their goal by using at most  $\vec{b}$  resources. Alternatively,  $\vec{b}$  can be understood as the initial budget of the computations, which is the interpretation followed along the paper. Then, the value  $\omega$  plays the role of an infinite supply of the resource.

The dual operator  $\llbracket A^{\vec{b}} \rrbracket$  is introduced as  $\llbracket A^{\vec{b}} \rrbracket\psi \stackrel{\text{def}}{=} \neg\langle\langle A^{\vec{b}} \rangle\rangle\neg\psi$ . The linear-time operators  $X$  and  $U$  have their standard readings ; while the propositional connectives  $\vee$ ,  $\rightarrow$ , and temporal operators *release*  $R$ , *always*  $G$ , and *eventually*  $F$  are introduced as usual. For instance,  $\phi R \psi \stackrel{\text{def}}{=} \neg(\neg\phi U \neg\psi)$ , and therefore  $\phi R \psi$  shall be equivalent to  $G\psi \vee (\neg\phi \wedge \psi) U(\phi \wedge \psi)$ .

We also consider the fragment  $RB\pm ATL(Ag, r)$  of  $RB\pm ATL^*(Ag, r)$ , where path formulas are restricted by  $\psi ::= X\phi \mid \phi U \phi \mid \phi R \phi$ .

**Remark 1** *Differently from [ABDL18], we explicitly consider the release operator  $R$  in our definition of  $RB\pm ATL$ . Indeed, in [LMO08] it is proved that, differently from the case of the Computation Tree Logic CTL, it is not possible to express  $R$  in terms of  $X$  and  $U$  in ATL. This proof can be quite easily adapted to the case of  $RB\pm ATL$  by considering the subclass of CGS assigning the cost 0 to all actions. Hence, we explicitly introduce the operator  $R$ . In Section 3.3, we will prove that this extra expressivity comes at no cost in terms of the complexity of the verification problem.*

**Semantics.** We provide a formal interpretation of the languages  $RB\pm ATL^*$  and  $RB\pm ATL$  by using resource-bounded CGS. Specifically, we need a formal notion of resource-bounded strategy for the interpretation of strategic operators  $\langle\langle A^{\vec{b}} \rangle\rangle$ . To start with, a (*memoryful*) strategy  $F_A$  for coalition  $A$  is a map from the set of finite computations to the set of joint actions of  $A$  such that  $F_A(s_0 \xrightarrow{f_0} s_1 \dots \xrightarrow{f_{n-1}} s_n) \in D_A(s_n)$ . A computation  $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \dots$  respects strategy  $F_A$  iff for all  $i < |\lambda|$ ,  $s_{i+1} \in out(s_i, F_A(s_0 \xrightarrow{f_0} s_1 \dots \xrightarrow{f_{i-1}} s_i))$ . A computation  $\lambda$  that respects  $F_A$  is *maximal* if it cannot be extended further while respecting the strategy. In the present context, maximal computations starting in state  $s$  and respecting  $F_A$  are infinite and we denote the set of all such computations by  $Comp(s, F_A)$ .

Given a bound  $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$  and a computation  $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \dots$  in  $Comp(s, F_A)$ , let the *resource availability*  $\vec{v}_i$  at step  $i < |\lambda|$  be defined as  $\vec{v}_0 = \vec{b}$  and for all  $i < |\lambda| - 1$ ,  $\vec{v}_{i+1} = cost_A(s_i, f_i) + \vec{v}_i$  (assuming  $n + \omega = \omega$  for every  $n \in \mathbb{Z}$ ). Then,  $\lambda$  is  $\vec{b}$ -consistent iff for all  $i < |\lambda|$ ,  $\vec{v}_i \in (\mathbb{N} \cup \{\omega\})^r$  (negative values are not allowed). If  $\vec{b}(i) = \omega$ , we actually have an infinite supply of the  $i$ -th resource, thus not constraining the behaviour of agents with respect to that particular resource. Since the resource availability depends only on the agents in  $A$ , in [ABDL18] this is called the *proponent restriction condition* (see also [ABLN15]). Without this restriction about the action costs of the opponent coalitions, the model-checking problem can be shown undecidable when the number of agents is unbounded, see e.g. [ABLN15]. The set of all the  $\vec{b}$ -consistent (infinite) computations is denoted by  $Comp(s, F_A, \vec{b})$ . A  $\vec{b}$ -strategy  $F_A$  with respect to

$s$  is a strategy such that  $Comp(s, F_A) = Comp(s, F_A, \vec{b})$ .

**Definition 3 (Satisfaction relation)** We define the satisfaction relation  $\models$  for a state  $s \in S$ , an infinite computation  $\lambda, p \in AP$ , a state-formula  $\phi$ , and a path-formula  $\psi$  as follows (clauses for Boolean connectives are standard and thus omitted) :

$$\begin{aligned} (M, s) \models p & \quad \text{iff} \quad s \in L(p) \\ (M, s) \models \langle\langle A^{\vec{b}} \rangle\rangle \psi & \quad \text{iff} \quad \text{for some } \vec{b}\text{-strategy } F_A \text{ w.r.t. } s, \\ & \quad \forall \lambda \in Comp(s, F_A), (M, \lambda) \models \psi \\ (M, \lambda) \models \phi & \quad \text{iff} \quad (M, \lambda_0) \models \phi \\ (M, \lambda) \models X\psi & \quad \text{iff} \quad (M, \lambda_{\geq 1}) \models \psi \\ (M, \lambda) \models \psi \cup \psi' & \quad \text{iff} \quad \text{for some } i \geq 0, (M, \lambda_{\geq i}) \models \psi', \\ & \quad \text{and } \forall 0 \leq j < i, (M, \lambda_{\geq j}) \models \psi \end{aligned}$$

Clearly,  $ATL^*$  and  $ATL$  [AHK02] can be seen as fragments of  $RB\pm ATL^*$  and  $RB\pm ATL$  respectively. In particular, the unindexed strategic operator  $\langle\langle A \rangle\rangle$  can be expressed as  $\langle\langle A^{\vec{\omega}} \rangle\rangle$ .

In the sequel, we consider the following decision problem.

**Definition 4 (Model Checking)** Let  $k, r \geq 1$ ,  $\phi$  a formula in  $RB\pm ATL^*([1, k], r)$  (resp.  $RB\pm ATL([1, k], r)$ ),  $M$  be a finite RB-CGS for  $Ag = [1, k]$  and  $r$  resources, and let  $s$  be a state in  $M$ . The model checking problem amounts to decide whether  $(M, s) \models \phi$ .

We conclude this section with a remark on the case of a single agent, which will be prominent in what follows.

**Remark 2** In the case of a single agent, that is, for  $Ag = \{1\}$ , in our languages we only have modalities  $\langle\langle Ag^{\vec{b}} \rangle\rangle$  and  $\llbracket \emptyset^{\vec{b}} \rrbracket$ , as well as duals  $\llbracket Ag^{\vec{b}} \rrbracket$  and  $\llbracket \emptyset^{\vec{b}} \rrbracket$ , for  $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$ . By Definition 3, the meaning of these operators is as follows :

$$\begin{aligned} (M, s) \models \langle\langle \emptyset^{\vec{b}} \rangle\rangle \psi & \quad \text{iff for every computation } \lambda \text{ from } s, \\ & \quad (M, \lambda) \models \psi \\ (M, s) \models \llbracket \emptyset^{\vec{b}} \rrbracket \psi & \quad \text{iff for some computation } \lambda \text{ from } s, \\ & \quad (M, \lambda) \models \psi \\ (M, s) \models \langle\langle Ag^{\vec{b}} \rangle\rangle \psi & \quad \text{iff for some } \vec{b}\text{-consistent computation} \\ & \quad \lambda \text{ from } s, (M, \lambda) \models \psi \\ (M, s) \models \llbracket Ag^{\vec{b}} \rrbracket \psi & \quad \text{iff for every } \vec{b}\text{-consistent computation} \\ & \quad \lambda \text{ from } s, (M, \lambda) \models \psi \end{aligned}$$

Notice that the semantics of operators  $\langle\langle \emptyset^{\vec{b}} \rangle\rangle$  and  $\llbracket \emptyset^{\vec{b}} \rrbracket$  corresponds to the meaning of modalities  $A$  and  $E$  in  $CTL^*$ ; whereas  $\langle\langle Ag^{\vec{b}} \rangle\rangle$  and  $\llbracket Ag^{\vec{b}} \rrbracket$  can be used to introduce resource-bounded counterparts  $E^{\vec{b}}$  and  $A^{\vec{b}}$  of modalities  $E$  and  $A$ . In Section 2.3, we show that  $RB\pm ATL^*$  for the single agent case is basically equivalent to a different resource-bounded logic  $RBTL^*$  introduced in [BF09].

One of our goals is to provide a framework for the complexity classification of (fragments of)  $RB\pm ATL(Ag, r)$ , as well as extensions such as  $RB\pm ATL^*(Ag, r)$ . Mainly, we focus on bounding the number of agents or resources, proving novel results along the way.

## 2.3 When $RBTL^*$ comes into play

Below, we present a resource-bounded temporal logic that extends  $CTL^*$  by adding resources [BF10, BF09]. Then, we show that this logic is essentially the same as single-agent  $RB\pm ATL^*$  described in Example 2. While such a result is not surprising, apparently it has so far been overlooked in the literature<sup>1</sup>. Such an equivalence allows us to apply results for single-agent  $RB\pm ATL$  to  $RBTL$  as well. We first introduce the syntax and semantics of  $RBTL^*$  as given in [BF09].

**Definition 5** Given  $r \geq 1$ , the state-formulas  $\phi$  and path-formulas  $\psi$  in  $RBTL^*$  are built according to the following BNF :

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle \vec{b} \rangle\rangle \psi$$

$$\psi ::= \phi \mid \neg\psi \mid \psi \wedge \psi \mid X\psi \mid \psi \cup \psi,$$

where  $p \in AP$  and  $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$ . Formulas in  $RBTL^*$  are all and only the state-formulas generated by the BNF.

The fragment  $RBTL$  of  $RBTL^*$  is obtained by restricting path formulas just like in the case of  $RB\pm ATL$  :  $\psi ::= X\phi \mid \phi \cup \phi \mid \phi R \phi$ . In [ABDL18] the interpretation of  $RBTL^*$  is given on a particular class of models, based on vector addition systems with states :

**Definition 6 (Model)** A model for  $RBTL^*$  is a tuple  $A = \langle Q, r, R, L \rangle$  s.t. (i)  $(Q, r, R)$  is a vector addition system with states (VASS), that is,

1.  $Q$  is a non-empty finite set of control states ;
2.  $r \geq 1$  is the number of counters ;
3.  $R$  is a finite subset of  $Q \times \mathbb{Z}^r \times Q$  ;

and (ii)  $L : AP \rightarrow \wp(Q)$  is a labelling function.

In a model  $A$ , a pseudo-run  $\lambda$  is an infinite sequence  $(q_0, \vec{v}_0) \rightarrow (q_1, \vec{v}_1) \rightarrow \dots$  such that for all  $i \geq 0$ , there exists  $(q, \vec{u}, q') \in R$  such that  $q_i = q$ ,  $q_{i+1} = q'$ , and  $\vec{v}_{i+1} = \vec{u} + \vec{v}_i$ . A pseudo-run  $\lambda$  is a run iff for all  $i \geq 0$ ,  $\vec{v}_i \in (\mathbb{N} \cup \omega)^r$ .

**Definition 7 (Satisfaction relation)** We define the satisfaction relation  $\models$  in model  $A$ , for state  $q \in Q$ , run  $\lambda, p \in AP$ , state-formula  $\phi$ , and path-formula  $\psi$  as follows (clauses for Boolean connectives are immediate and thus omitted) :

$$\begin{aligned} (A, q) \models p & \quad \text{iff} \quad q \in L(p) \\ (A, q) \models \langle\langle \vec{b} \rangle\rangle \psi & \quad \text{iff for some run } \lambda \text{ from } (q, \vec{b}), \\ & \quad (A, \lambda) \models \psi \\ (A, \lambda) \models \phi & \quad \text{iff} \quad (A, \lambda_0) \models \phi \\ (A, \lambda) \models X\psi & \quad \text{iff} \quad (A, \lambda_{\geq 1}) \models \psi \\ (A, \lambda) \models \psi \cup \psi' & \quad \text{iff for some } i \geq 0, (A, \lambda_{\geq i}) \models \psi', \\ & \quad \text{and for all } 0 \leq j < i, (A, \lambda_{\geq j}) \models \psi \end{aligned}$$

<sup>1</sup> Indeed, in [ABDL18], complexity results are given independently for both  $RBTL^*$  and single-agent  $RB\pm ATL^*$ , even though the two logics can be translated one into the other.

Next, we prove that the logics  $\text{RBTL}^*$  and  $\text{RB}\pm\text{ATL}^*(\{1\}, r)$  with a single agent are semantically equivalent, in the sense that truth-preserving translations exist between models and formulas. First, consider the translation map  $\tau$  from  $\text{RBTL}^*$  to  $\text{RB}\pm\text{ATL}^*(\{1\}, r)$  such that  $\tau$  is the identity on  $AP$ , it is homomorphic for Boolean and temporal operators, and  $\tau(\langle \vec{b} \rangle \psi) \stackrel{\text{def}}{=} \mathbb{E}^{\vec{b}} \tau(\psi)$ . Actually, it can be shown that  $\tau$  is a bijection between  $\text{RBTL}^*$  and  $\text{RB}\pm\text{ATL}^*(\{1\}, r)$ . Not only that, but  $\tau$  is a bijection between  $\text{RBTL}$  and  $\text{RB}\pm\text{ATL}(\{1\}, r)$  as well. Further, given a resource-bounded CGS  $M = \langle \{1\}, AP, S, Act, r, act, cost, \delta, L \rangle$  with a single agent 1, define the associated model  $A_M = \langle S, r, R, L \rangle$  for  $\text{RBTL}^*$  such that

- $R$  is the set of tuples  $(q, \vec{u}, q')$  such that  $\delta(q, \mathbf{a}) = q'$  for some action  $\mathbf{a} \in act(q, 1)$  with  $cost(q, 1, \mathbf{a}) = \vec{u}$ .

Symmetrically, given a model  $A = \langle Q, r, R, L \rangle$ , define the associated single-agent, resource-bounded CGS  $M_A = \langle \{1\}, Q, R, r, act, cost, \delta, L \rangle$  such that for every  $q \in Q$ ,

- $act(q, 1) = \{(q', \vec{u}, q'') \in R \mid q = q'\}$ ;
- for every  $(q, \vec{u}, q') \in act(q, 1)$ ,  $cost(q, 1, (q, \vec{u}, q')) = \vec{u}$ ;
- for every  $(q, \vec{u}, q') \in act(q, 1)$ ,  $\delta(q, (q, \vec{u}, q')) = q'$ .

We now state the following auxiliary lemma, whose proof follows immediately by the definitions of  $A_M$  and  $M_A$  above.

- Lemma 1** 1. Given a single-agent, resource-bounded CGS  $M$  and state  $s \in S$ , for every  $\vec{b}$ -consistent computation  $\lambda$  in  $M$ , in  $A_M$  there exists a run  $\lambda'$  from  $(s, \vec{b})$  such that for every  $i \geq 0$ ,  $(\lambda_i, \vec{v}_i) \rightarrow (\lambda_{i+1}, \vec{v}_{i+1})$  with  $\vec{v}_{i+1} = \vec{v}_i + \vec{u}$  for  $\vec{u} = cost(\lambda_i, 1, \mathbf{a}_i)$  and  $\lambda_i \xrightarrow{\mathbf{a}_i} \lambda_{i+1}$ .
2. Given a model  $A$  for  $\text{RBTL}^*$  and state  $q \in Q$ , for every run  $\lambda$  from  $(s, \vec{b})$ , in  $M_A$  there exists a  $\vec{b}$ -consistent computation  $\lambda'$  such that for every  $i \geq 0$ ,  $\lambda_i \xrightarrow{(\lambda_i, \vec{u}, \lambda_{i+1})} \lambda_{i+1}$  for  $(\lambda_i, \vec{v}_i) \rightarrow (\lambda_{i+1}, \vec{v}_{i+1})$  and  $\vec{v}_{i+1} = \vec{u} + \vec{v}_i$ .

By using Lemma 1 we can finally prove that  $\text{RB}\pm\text{ATL}^*(\{1\}, r)$  and  $\text{RBTL}^*$  are closely related semantically.

**Theorem 1** (1) For every  $\phi$  in  $\text{RBTL}^*$  and model  $A$  with state  $q \in Q$ ,  $(A, q) \models \phi$  iff  $(M_A, q) \models \tau(\phi)$ . (2) For every  $\phi'$  in  $\text{RB}\pm\text{ATL}^*$  and single-agent, resource-bounded CGS  $M$  with state  $s \in S$ ,  $(M, s) \models \phi'$  iff  $(A_M, s) \models \tau^{-1}(\phi')$ .

Consequently,  $\text{RBTL}^*$  and the restriction of  $\text{RB}\pm\text{ATL}^*$  to a single agent are essentially the same logic in the sense that their translations are semantically faithful when single-agent RB-CGS are understood as  $\text{RBTL}^*$  models (i.e., a VASS with a valuation). A similar result holds for  $\text{RBTL}$  and single-agent  $\text{RB}\pm\text{ATL}$ . This result is particularly relevant in the light of Section 3, where we dig deeper into the verification of single-agent, resource-bounded logics.

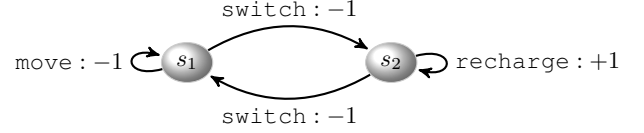


FIGURE 1 – The resource-bounded CGS in Example 1. Transitions with action *idle* are omitted.

**Example 1** We illustrate the formal machinery introduced so far, particularly the single-agent case, with a toy example. We consider a scenario in which a rover is exploring an unknown area. At any time the rover can choose between two modes : either it moves around or it recharges its battery through a solar panel, but it cannot do both things at the same time. Moving around consumes one energy unit at every time step, whereas the rover can recharge of one energy unit at a time. Switching between these modes requires one energy unit.

This simple scenario can be modelled as the resource-bounded CGS  $M = \langle \{\text{rover}\}, \{s_1, s_2\}, \{\text{move}, \text{recharge}, \text{switch}, \text{idle}\}, 1, act, cost, \delta, L \rangle$  depicted in Figure 1, where in particular :

- $act(s_1, \text{rover}) = \{\text{move}, \text{switch}, \text{idle}\}$  and  $act(s_2, \text{rover}) = \{\text{recharge}, \text{switch}, \text{idle}\}$ ;
- $cost(s_1, \text{rover}, \text{move}) = cost(s_1, \text{rover}, \text{switch}) = cost(s_2, \text{rover}, \text{switch}) = -1$  and  $cost(s_2, \text{rover}, \text{recharge}) = +1$ ;
- $\delta(s_1, \text{move}) = s_1$ ,  $\delta(s_1, \text{switch}) = s_2$ ,  $\delta(s_2, \text{recharge}) = s_2$ , and  $\delta(s_2, \text{switch}) = s_1$ ;
- $AP = \{\text{moving}\}$  and  $L(\text{moving}) = \{s_1\}$ .

Even in such a simple scenario with a single agent, we can express interesting properties such as “no matter what the rover does, at any time it has a strategy, with an initial budget of at most  $b$  energy units, such that it will eventually be moving”. This specification can be expressed in  $\text{RB}\pm\text{ATL}$  as

$$\llbracket \{\text{rover}\}^\omega \rrbracket G(\llbracket \{\text{rover}\}^b \rrbracket F \text{moving}) \quad (1)$$

Next, we show that specifications such as (1), concerning a single agent and a single resource, can be efficiently verified in PTIME.

### 3 Model-checking $\text{RB}\pm\text{ATL}(\{1\}, 1)$

This section is devoted to the technical developments of our main theoretical results. Specifically, in Section 3.1 we review the known complexity results for model checking  $\text{RB}\pm\text{ATL}$  and its fragments. Then, in Section 3.2 we prove that the control-state reachability and nontermination problems for vector addition systems with states (VASS) with one counter are decidable in PTIME. These results are then used in Section 3.3 to show that the model-checking problem for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$  is also in PTIME. Thus, our

contribution shows that reasoning about a single resource in  $\text{RB}\pm\text{ATL}$  with a single agent comes at no extra computational cost compared to CTL.

### 3.1 Model Checking Results for $\text{RB}\pm\text{ATL}$

In Table 1, we summarize the main complexity results available in the literature for  $\text{RB}\pm\text{ATL}(Ag, r)$ , depending on the number  $|Ag|$  of agents and the number  $r$  of resources. The result in boldface is original from this contribution. All the results hold in the presence of R instead of G, except the PSPACE upper bound from [ALNR17, Theo. 2].

For an unbounded number of resources and at least two agents, the model-checking problem is known to be 2EXPTIME-complete. This result follows from Theorem 2 (membership) and Theorem 3 (hardness) in [ABDL18]. When restricted to a single agent, the problem becomes EXSPACE-complete [ABDL18, Th. 4].

For a fixed number of resources greater than four and at least two agents, the model-checking problem is again EXPTIME-complete. The upper bound follows from [ABDL18, Cor. 1], while the lower bound derives from the complexity of the control-state reachability problem for alternating VASS [CS14], which can be simulated by using two agents only [ABDL18, Th. 3]. Further, for a fixed amount of resources greater than two, and two agents, the model-checking problem is in EXPTIME [ABDL18, Cor. 1]. In the case of a single agent, the same problem is in PSPACE [ABDL18, Cor. 2]; whereas it is PSPACE-hard in both cases, as we can reduce to it the control-state reachability problem for 2-VASS, which is PSPACE-complete [BFG<sup>+</sup>15].

Finally, in the case of a single resource, the problem is known to be in PSPACE [ALNR17, Th. 2] (the result is established for a language with G and it is plausible to extend it to R). For the case of a single agent, model checking is in PTIME, which is the main theoretical contribution of this section. It is therefore PTIME-complete as model checking CTL is already PTIME-hard (see, e.g., [Sch03, DGL16]). The characterisation of the complexity for one resource and at least two agents is still open : currently, neither the proof of the PTIME upper bound in Section 3.3, nor the PSPACE-hardness results from [JS07] and [FLL<sup>+</sup>17, Sect. 5] could be advantageously used to close this complexity gap.

### 3.2 Decision problems for 1-VASS

In order to show that the model-checking problem for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$  is PTIME-complete, we establish that two well-known decision problems on vector addition systems with states (VASS), when restricted to a single counter, can be solved in polynomial time. More precisely, we show that the control-state reachability and nontermination problems for 1-VASS are in PTIME, whereas, for instance, the control-state reachability problem for VASS is EXSPACE-complete in general [Lip76, Rac78]. Although control-state reachability is a subproblem of the covering problem, that has been quite studied (see, e.g., [AH09, BS11, Dem13]), to the best of our knowledge there is no

result in the literature on the upper bound when restricted to a single counter. Hereafter, we provide formal arguments for tractability by appropriately tuning and correcting the proof technique dedicated to the boundedness problem for 1-VASS from [RY86]. Note also that in [GHLT16], the updates in the BVASS (extending VASS) are restricted to the set  $\{-1, 0, +1\}$  (see [GHLT16, Def. 1]). Therefore the upper bound in [GHLT16] does not extend to our present case where updates are arbitrary integers encoded in binary. When updates are arbitrary integers encoded in binary (as done herein), the relevant problems for 1-BVASS are known to be PSPACE-complete [FLL<sup>+</sup>17].

We recall the notion of VASS as given in Definition 6, so a VASS is a structure  $V = (Q, r, R)$ , where  $R$  is a finite set of transitions. A *configuration* of a VASS  $V$  is defined as a pair  $(q, \vec{x}) \in Q \times \mathbb{N}^r$  ( $\omega$  is discarded in this section). Given  $(q, \vec{x})$ ,  $(q', \vec{x}')$  and a transition  $t = q \xrightarrow{\vec{u}} q'$ , we write  $(q, \vec{x}) \xrightarrow{t} (q', \vec{x}')$  whenever  $\vec{x}' = \vec{u} + \vec{x}$ . Then,  $(q_0, \vec{x}_0)$  is called the *initial configuration*.

An  $r$ -VASS is a VASS with  $r$  counters. We present two standard decision problems on VASS that play a crucial role in solving the model checking problem for  $\text{RB}\pm\text{ATL}(Ag, 1)$ .

*Control state reachability problem* CREACH(VASS) :

**Input :** a VASS  $V$ , a configuration  $(q_0, \vec{x}_0)$ , and a control state  $q_f$ .

**Question :** is there a finite run with initial configuration  $(q_0, \vec{x}_0)$  and with final configuration with state  $q_f$  ?

*Nontermination problem* NONTER(VASS) :

**Input :** a VASS  $V$  and a configuration  $(q_0, \vec{x}_0)$ .

**Question :** is there an infinite run with initial configuration  $(q_0, \vec{x}_0)$  ?

Other classical decision problems for VASS have been considered in the literature (see e.g. recent developments about the reachability problem in [Sch16, CLL<sup>+</sup>18]), but in this paper we only need to tame the control-state reachability and nontermination problems for 1-VASS in order to solve the model-checking problem for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$ .

**Definition 8 (Simple Run, Path, and Loop)** A simple run  $\rho = (q_0, \vec{x}_0), \dots, (q_k, \vec{x}_k)$ ,  $k \geq 0$ , is a finite run such that no control state appears twice. A simple path is a sequence of transitions  $t_1 \dots t_k$  such that no control state occurs more than once. A simple loop is a sequence of transitions  $t_1 \dots t_k$  such that the first control state of  $t_1$  is equal to the second control state of  $t_k$  (and it occurs nowhere else) and no other control state occurs more than once.

In a 1-VASS, a simple loop is (*strictly*) positive if the cumulated effect is (*strictly*) positive. Given a run  $\rho = (q_0, x_0), \dots, (q_k, x_k), \dots$  and  $\alpha \geq 0$ , we write  $\rho^{+\alpha}$  to denote the sequence  $(q_0, x_0 + \alpha), \dots, (q_k, x_k + \alpha), \dots$ . If  $\rho$  is a run, the sequence  $\rho^{+\alpha}$  is also a run. The following lemma provides 1-VASS with a characterisation of runs ending in a distinguished final state.

$r \setminus  Ag $	$\infty$	<b>2</b>	<b>1</b>
$\infty$	$2\text{EXPTIME-c. [ABDL18, Th. 2 and 3]}$	$\text{EXPSpace-c. [ABDL18, Th. 4]}$	
$\geq 4$	$\text{EXPTIME-c. [ABDL18, Cor. 1]}$	$\text{PSPACE-h. [BFG}^+15]$	$\text{in PSPACE [ABDL18, Cor. 2]}$
<b>3</b>	$\text{PSPACE-h. [BFG}^+15]$		
<b>2</b>	$\text{in EXPTIME [ABDL18, Cor. 1]}$		
<b>1</b>	$\text{PTIME-h. (from ATL)}$ $\text{in PSPACE [ALNR17, Th. 2]}$	$\text{PTIME-h. (from CTL)}$ <b>in PTIME (Th. 4)</b>	

TABLE 1 – The complexity of model checking  $\text{RB}\pm\text{ATL}(Ag, r)$ .

**Lemma 2** *Let  $V$  be a 1-VASS,  $(q_0, x_0)$  an initial configuration, and  $q_f$  a location. There is a finite run from  $(q_0, x_0)$  to configuration  $(q_f, x_f)$  for some  $x_f \geq 0$  iff (1) either  $q_0 = q_f$ ; or*

2. *there is a simple path  $(q_0, x_0), \dots, (q_k, x_k)$  with  $q_k = q_f$ ; or*
3. *we have that*
  - *there is a simple run  $(q_0, x_0), \dots, (q_n, x_n)$ ,*
  - *there is a strictly positive simple loop  $t_1 \dots t_\beta$  such that  $(q_n, x_n) \xrightarrow{t_1 \dots t_\beta} (q_n, x_n + \alpha)$  is a run ( $\alpha > 0$ ),*
  - *there is a simple path starting at  $q_n$  and ending at  $q_f$ .*

As illustration, Figure 2 presents a 1-VASS  $V$ , and witness runs and path for the positive instance  $(V, (q_0, 7), q_f)$  of  $\text{CREACH}(1\text{-VASS})$ . By contrast, the configuration  $(q_0, 5)$  cannot reach  $q_f$ .

**Proof** First, it is not difficult to check that if either (1), (2) or (3) holds, then there is a finite run from  $(q_0, x_0)$  to configuration  $(q_f, x_f)$  for some  $x_f \geq 0$ . By way of example, firing the strictly positive simple loop at least  $(|Q| \times \max\{u \mid q \xrightarrow{u} q' \text{ is a transition}\})$  times, allows to pursue the run following the path from  $q_n$  to  $q_f$ .

Conversely, let us suppose that  $\rho = (q_0, x_0), \dots, (q_k, x_k)$  is a run with  $q_k = q_f$ . If  $q_0 = q_f$ , then the witness run can be reduced to  $(q_0, x_0)$ . Otherwise, either  $\rho$  is a simple run and condition (2) holds, or there are  $0 \leq i < j \leq k$  such that  $q_i = q_j$ . In case  $x_i \geq x_j$ , the subrun  $(q_i, x_i), \dots, (q_j, x_j)$  can be removed from  $\rho$  while leading to a run reaching  $q_f$ . Typically, the suffix subrun  $(q_i, x_i), \dots, (q_j, x_j), \dots, (q_k, x_k)$  with  $\rho_{\dagger} = (q_j, x_j), \dots, (q_k, x_k)$  is replaced by  $\rho_{\dagger}^{+\alpha}$  for  $\alpha = x_j - x_i$ . Such a transformation can be performed as soon as the subruns correspond to the application of simple loops with negative effect. Without loss of generality, we can assume that  $\rho$  has no loop with strictly negative effect.

If  $\rho$  is not a simple run, there are  $0 \leq I < J \leq |Q|$  such that  $q_I = q_J$  and  $x_I < x_J$ . Consequently,

- there is a simple run  $(q_0, x_0), \dots, (q_I, x_I)$ ;
- there is a strictly positive simple loop  $t_I \dots t_{J-1}$  such that  $(q_I, x_I) \xrightarrow{t_I \dots t_{J-1}} (q_I, x_I + (x_J - x_I))$ ;

- there is a path from  $q_J$  to  $q_k = q_f$  such that  $(q_J, x_J), \dots, (q_k, x_k)$  is a run. So, there is a simple path from  $q_J$  to  $q_k$ .

As a result, condition (3) is satisfied and the lemma holds.

The characterisation in Lemma 2 can be turned into an algorithm running in polynomial time.

**Theorem 2** *The problem  $\text{CREACH}(1\text{-VASS})$  is in PTIME.*

**Proof** Let  $V$  be a 1-VASS,  $(q_0, x_0)$  an initial configuration, and  $q_f$  a location. If  $q_0 = q_f$ , we are done. Otherwise, define values  $\text{maxval}_q^i$  for  $i \in [0, |Q|]$  and  $q \in Q$  such that if there is a run  $(q_0, x_0), \dots, (q_j, x_j)$  with  $q_j = q$  and  $j \leq i$ , then the maximal value  $x_j$  among all these runs is precisely  $\text{maxval}_q^i$ . When there is no such run, by convention  $\text{maxval}_q^i = -\infty$ . Similar values have been considered to solve the boundedness problem for 1-VASS in [RY86]. Let us compute the values  $\text{maxval}_q^i$ :

- $\text{maxval}_{q_0}^0 \stackrel{\text{def}}{=} x_0$  and  $\text{maxval}_q^0 \stackrel{\text{def}}{=} -\infty$  for all  $q \neq q_0$ .
- For all  $q$  and  $i + 1 \in [1, |Q|]$ ,

$$\text{maxval}_q^{i+1} \stackrel{\text{def}}{=} \max(\text{maxval}_q^i,$$

$$\{\text{maxval}_{q'}^j + u \in \mathbb{N} \mid j \leq i, q \xrightarrow{u} q' \text{ is a transition}\}).$$

The values  $\text{maxval}_q^i$ 's can be computed in polynomial time in the size of  $V$  (the number  $|Q|$  of locations being an essential parameter as well as the maximal absolute value  $|u|$  from updates –integers being written in binary). One can show that  $\text{maxval}_q^i$  is indeed the maximal value as specified above.

Further, note that condition (2) in Lemma 2 is equivalent to  $\text{maxval}_{q_f}^{|Q|} \neq -\infty$ . Similarly, the three conditions in (3) from Lemma 2 are equivalent to: there are  $q \in Q$  and  $I < J \leq |Q|$  such that

- $\text{maxval}_q^I \neq -\infty$ .
- $\text{maxval}_q^I < \text{maxval}_q^J$  and  $\text{auxval}_q^0 < \text{auxval}_q^{J-I}$ , where the values  $\text{auxval}_q^i$ 's ( $i \in [0, J-I], q' \in Q$ ) are defined as follows (similarly to what is done for the  $\text{maxval}_{q'}^j$ 's):
  - $\text{auxval}_q^0 \stackrel{\text{def}}{=} \text{maxval}_q^I$  and  $\text{auxval}_{q'}^0 \stackrel{\text{def}}{=} -\infty$  for all  $q' \neq q$ .



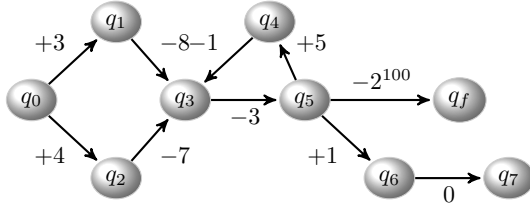


FIGURE 2 – Witness runs and path from Lemma 2(3)

Witness runs and path for  $(V, (q_0, 7), q_f)$  :

initial run :  $(q_0, 7), (q_2, 11), (q_3, 4)$

“> 0 loop” :  $(q_3, 4), (q_5, 1), (q_4, 6), (q_3, 5)$

final path :  $q_3 \rightarrow q_5 \rightarrow q_f$

— For all  $q'$  and  $i + 1 \in [1, J - I]$ ,

$$\text{auxval}_{q'}^{i+1} \stackrel{\text{def}}{=} \max(\text{auxval}_{q'}^i,$$

$$\max(\text{auxval}_{q'}^i,$$

$$\{\text{auxval}_{q''}^j + u \in \mathbb{N} \mid j \leq i, q' \xrightarrow{u} q'' \text{ is a transition}\}).$$

— There is a simple path starting at  $q$  and ending at  $q_f$ .

The first two points above can be checked in PTIME, and the third one in NLOGSPACE as it is an instance of the standard graph reachability problem GAP. So, CREACH(1-VASS) is in PTIME.

Note that the values  $\text{auxval}_{q'}^i$ 's in the proof of Theorem 2 are necessary to guarantee that the values  $\text{maxval}_q^I$  and  $\text{maxval}_q^J$  are obtained following a common subrun until reaching the configuration  $(q, \text{maxval}_q^I)^2$ . Now, let us turn to the characterisation of runs and paths witnessing nontermination.

**Lemma 3** *Let  $V$  be a 1-VASS and  $(q_0, x_0)$  an initial configuration. There is an infinite run starting at  $(q_0, x_0)$  iff*

- *there is a simple run  $(q_0, x_0), \dots, (q_n, x_n)$ ; and*
- *there is a positive simple loop  $t_1 \dots t_k$  such that  $(q_n, x_n) \xrightarrow{t_1 \dots t_k} (q_n, x_n + \alpha)$  is a run ( $\alpha \geq 0$ ).*

**Proof** Clearly, the satisfaction of the two conditions implies that there is an infinite run starting at  $(q_0, x_0)$  : just consider the run generated by  $(t_1 \dots t_k)^\omega$  from configuration  $(q_n, x_n)$ . Let us prove the other direction, similarly to what is done in the proof of Lemma 2. Suppose that  $\rho = (q_0, x_0), \dots, (q_k, x_k), \dots$  is an infinite run. Without loss of generality, we can assume that  $\rho$  has no simple loop with strictly negative effect. There are  $n \geq 0$  and  $q \in Q$  such that  $q_n = q, \{i \in \mathbb{N} \mid q_i = q\}$  is infinite and  $(q_0, x_0), \dots, (q_n, x_n)$  is a simple run. Consider some  $J > I \geq |Q|$  such that  $q_J = q_I = q$  (such an index  $J$  necessarily exists). Obviously, there is a positive simple loop  $t_I \dots t_{J-1}$  such that  $(q_I, x_I) \xrightarrow{t_I \dots t_{J-1}} (q_J, x_J)$  is a run. Hence, both conditions in the statement of the lemma hold.

2. We remark that in the proof of [RY86, Theorem 3.4] for solving the boundedness problem for 1-VASS in PTIME, a similar argument should have been used.

Once more, the characterisation in Lemma 3 can be turned into an algorithm to check nontermination, running in polynomial time.

**Theorem 3** *The problem NONTER(1-VASS) is in PTIME.*

**Proof** Let  $V$  be a 1-VASS and  $(q_0, x_0)$  an initial configuration. Define the values  $\text{maxval}_q^i$  for  $i \in [0, |Q|]$  and  $q \in Q$  such that if there is a run  $(q_0, x_0), \dots, (q_i, x_i)$  with  $q_i = q$ , then the maximal value  $x_i$  among all these runs is precisely  $\text{maxval}_q^i$ . Note that these values are not the same as those from the proof of Theorem 2 as we consider runs of length *exactly*  $i$ . When there is no such run, by convention  $\text{maxval}_q^i = -\infty$ .

- $\text{maxval}_{q_0}^0 \stackrel{\text{def}}{=} x_0$  and  $\text{maxval}_q^0 \stackrel{\text{def}}{=} -\infty$  for all  $q \neq q_0$ .
- For all  $q$  and  $i + 1 \in [1, |Q|]$ ,

$$\text{maxval}_q^{i+1} \stackrel{\text{def}}{=} \max(\{\text{maxval}_{q'}^i + u \in \mathbb{N} \mid q \xrightarrow{u} q' \text{ is a transition, } \text{maxval}_{q'}^i \neq -\infty\}).$$

By convention, the maximal value of the empty set is  $-\infty$ .

All the values  $\text{maxval}_q^i$ 's can be computed in polynomial time in the size of  $V$ . One can show that  $\text{maxval}_q^i$  is the maximal value as specified above. Finally, the characterisation in Lemma 3 is equivalent to : there are  $q \in Q$  and  $I < J \leq |Q|$  such that  $\text{maxval}_q^I \neq -\infty$  and  $\text{maxval}_q^I \leq \text{maxval}_q^J$  and  $\text{auxval}_q^0 \leq \text{auxval}_q^{J-I}$ , where values  $\text{auxval}_{q'}^i$ 's ( $i \in [0, J - I], q' \in Q$ ) are defined as

- $\text{auxval}_q^0 \stackrel{\text{def}}{=} \text{maxval}_q^I$  and  $\text{auxval}_{q'}^0 \stackrel{\text{def}}{=} -\infty$  for all  $q' \neq q$ ;
- for all  $q'$  and  $i + 1 \in [1, J - I]$ ,

$$\text{auxval}_{q'}^{i+1} \stackrel{\text{def}}{=} \max\{\text{auxval}_{q''}^j + u \in \mathbb{N} \mid j \leq i, q' \xrightarrow{u} q'' \text{ is a transition}\}.$$

All conditions can be checked in polynomial time and therefore the nontermination problem for 1-VASS is in PTIME.

To conclude, by Theorem 2 and 3 both the state-reachability and nontermination problems for 1-VASS are decidable in PTIME.

### 3.3 Model-checking $\text{RB}\pm\text{ATL}(\{1\}, 1)$ is in PTIME

In this section we establish our main theoretical result, that is, the model-checking problem for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$  is PTIME-complete (forthcoming Theorem 4) by leveraging on Theorem 2 and 3. Hereafter, for every  $b \in \mathbb{N} \cup \{\omega\}$ , we write  $\langle\langle b \rangle\rangle\phi$  instead of  $\langle\langle \{1\}^b \rangle\rangle\phi$ . We observe that the case of a single resource can also capture situations in which  $r > 1$  resources can be converted into a unique resource (e.g., money), possibly with different rates.

As done in Section 2.3, given a resource-bounded CGS  $M = (\{1\}, S, \text{Act}, 1, \text{act}, \text{cost}, \delta, L)$  with a single agent and a single resource, let us define the 1-VASS  $V_M = (S, 1, R_V)$  such that  $q \xrightarrow{a} q' \in R_V$  iff there is some action  $\mathbf{a} \in \text{act}(q, 1)$  such that  $\delta(q, \mathbf{a}) = q'$  and  $\text{cost}(q, 1, \mathbf{a}) = u$ . Similarly, we write  $K_M = (S, R, L_K)$  to denote the Kripke structure such that  $q R q'$  iff there is some action  $\mathbf{a} \in \text{act}(q, 1)$  such that  $\delta(q, \mathbf{a}) = q'$  and  $L_K(q) = \{q\}$  (by a slight abuse of notations, we assume that  $AP = Q$ ). Note that, thanks to the `idle` action,  $K_M$  is a total Kripke structure, i.e., every world has at least one successor. We introduce Kripke structures as the modality  $\langle\langle \omega \rangle\rangle$  amounts to forget about the costs in  $M$ , and therefore  $M$  can be understood as the Kripke structure  $K_M$ , and model checking reduces to CTL model checking. Similarly, as remarked in Section 2.3, the strategy modality  $\langle\langle \emptyset^b \rangle\rangle$  behaves as the universal path quantifier  $\mathbf{A}$  in cost-free transition systems.

We now investigate the relationship between computations in  $M$  and runs in  $V_M$  and in  $K_M$ , respectively (a variant of Lemma 1).

**Lemma 4** *Let  $M$  be a RB-CGS with a single agent and a single resource.*

- (I) *Let  $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \cdots$  be a  $b$ -consistent computation associated to the family of resource values  $(v_i)_{i \in \mathbb{N}}$ . If  $b \in \mathbb{N}$ , then  $(q_0, v_0) \rightarrow (q_1, v_1) \rightarrow (q_2, v_2) \cdots$  is an infinite run in  $V_M$ ; otherwise  $q_0 \rightarrow q_1 \rightarrow q_2 \cdots$  is an infinite path in  $K_M$ .*
- (II) *Let  $(q_0, v_0) \rightarrow (q_1, v_1) \rightarrow (q_2, v_2) \cdots$  be an infinite run in  $V_M$ . Then, there is a  $v_0$ -consistent computation  $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \cdots$  associated to the family of resource values  $(v_i)_{i \in \mathbb{N}}$ .*
- (III) *Let  $q_0 \rightarrow q_1 \rightarrow q_2 \cdots$  is an infinite path in  $K_M$ . Then, there is an  $\omega$ -consistent computation  $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \cdots$  in  $M$ .*

The proof of Lemma 4 follows immediately by definition, and this result is instrumental to the three following lemmas that are at the heart of the model-checking algorithm for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$ . Given  $S_1 \subseteq S$ , we write  $V_M^{S_1}$  (resp.  $K_M^{S_1}$ ) to denote the restriction of  $V_M$  (resp.  $K_M$ ) to the locations in  $S_1$  only.

**Lemma 5** *Let  $M$  be an RB-CGS for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$ ,  $S_1 \subseteq S$  with  $s \in S_1$ , and  $b \in \mathbb{N}$ .*

- (I) *There is a  $b$ -consistent computation starting at  $s$  in  $M$  that visits only states in  $S_1$  iff  $(V_M^{S_1}, (s, b))$  is a positive instance of  $\text{NONTER}(1\text{-VASS})$ .*
- (II) *There is an  $\omega$ -consistent computation starting in  $s$  in  $M$  that visits only states in  $S_1$  iff  $(K_M, s) \models \text{EG}(\bigvee_{s' \in S_1} s')$  in CTL.*

This is a consequence of Lemma 4 (which will be generalised in Lemma 7). Let us focus now on the until operator  $\mathbf{U}$ .

**Lemma 6** *Let  $M$  be a RB-CGS for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$ ,  $S_1, S_2 \subseteq S$  with  $s \in S$ , and  $b \in \mathbb{N}$ .*

- (I) *There is a  $b$ -consistent computation starting at  $s$  in  $M$  such that its projection on  $S$  is in  $S_1^* \cdot S_2 \cdot S^\omega$  (understood as an  $\omega$ -regular expression) iff for some  $s' \in S_2$ ,  $(V_M^{S_1 \cup S_2}, (s, b), s')$  is a positive instance of  $\text{CREACH}(1\text{-VASS})$ .*
- (II) *There is an  $\omega$ -consistent computation starting at  $s$  in  $M$  such that its projection on  $S$  is in  $S_1^* \cdot S_2 \cdot S^\omega$  iff in CTL, we have*

$$(K_M, s) \models \text{E}(\bigvee_{s' \in S_1} s') \mathbf{U}(\bigvee_{s' \in S_2} s').$$

This is again a consequence of Lemma 4 but here, we have to use the fact that the distinguished action `idle` is enabled in any state (which is handy to extend to the infinity a finite witness run). Finally, we consider the linear-time temporal operator  $\mathbf{R}$ .

**Lemma 7** *Let  $M$  be a RB-CGS for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$ ,  $S_1, S_2 \subseteq S$  with  $s \in S$ , and  $b \in \mathbb{N}$ .*

- (I) *There is a  $b$ -consistent computation starting at  $s$  in  $M$  such that its projection on  $S$  is in  $S_2^\omega \cup ((S \setminus S_1) \cap S_2)^* \cdot (S_1 \cap S_2) \cdot S^\omega$  iff either  $(V_M^{S_2}, (s, b))$  is a positive instance of  $\text{NONTER}(1\text{-VASS})$  or for some  $s' \in S_1 \cap S_2$ ,  $(V_M^{S_2}, (s, b), s')$  is a positive instance of  $\text{CREACH}(1\text{-VASS})$ .*
- (II) *There is an  $\omega$ -consistent computation starting at  $s$  in  $M$  such that its projection on  $S$  is in  $S_2^\omega \cup ((S \setminus S_1) \cap S_2)^* \cdot (S_1 \cap S_2) \cdot S^\omega$  iff in CTL, we have*

$$(K_M, s) \models (\text{EG} \bigvee_{s' \in S_2} s') \vee \text{E}(\bigvee_{s' \in (S \setminus S_1) \cap S_2} s') \mathbf{U}(\bigvee_{s' \in S_1 \cap S_2} s').$$

By using Lemmas 5-7 we derive our main theoretical result.

**Theorem 4** *The model-checking problem for  $\text{RB}\pm\text{ATL}(\{1\}, 1)$  is PTIME-complete.*

PTIME-hardness is inherited from the model-checking problem for CTL.

---

**Algorithm 1** – RB±ATL( $\{1\}, 1$ ) model checking –

---

```

1: procedure GMC( $M, \phi$ )
2:   case  $\phi$  of
3:      $p$ : return  $\{s \in S \mid s \in L(p)\}$ 
4:      $\neg\psi$ : return  $S \setminus GMC(M, \psi)$ 
5:      $\psi_1 \wedge \psi_2$ : return  $GMC(M, \psi_1) \cap GMC(M, \psi_2)$ 
6:      $\langle\langle b \rangle\rangle X \psi$ : return  $\{s \mid \exists \mathbf{a} \in act(s, 1), 0 \leq b + cost(s, 1, \mathbf{a}), \delta(s, \mathbf{a}) \in GMC(M, \psi)\}$ 
7:      $\langle\langle \omega \rangle\rangle X \psi$ :
           return  $\{s \mid \exists \mathbf{a} \in act(s, 1), \delta(s, \mathbf{a}) \in GMC(M, \psi)\}$ 
8:      $\langle\langle \emptyset^b \rangle\rangle X \psi$ :
           return  $\{s \mid \forall \mathbf{a} \in act(s, 1), \delta(s, \mathbf{a}) \in GMC(M, \psi)\}$ 
9:      $\langle\langle b \rangle\rangle \psi_1 \cup \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid \exists s' \in S_2 \text{ s.t. } V_M^{S_1 \cup S_2}(s, b), s' \text{ is a positive inst. of CREACH(1-VASS)}\}$ 
10:     $\langle\langle \omega \rangle\rangle \psi_1 \cup \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid K_{M, s} \models E(V_{s' \in S_1} s') U(V_{s' \in S_2} s')\}$ 
11:     $\langle\langle \emptyset^b \rangle\rangle \psi_1 \cup \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid K_{M, s} \models A(V_{s' \in S_1} s') U(V_{s' \in S_2} s')\}$ 
12:     $\langle\langle b \rangle\rangle \psi_1 R \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid V_M^{S_2}(s, b) \text{ is a positive inst. of NONTER(1-VASS)}\} \cup \{s \in S \mid \exists s' \in S_1 \cap S_2 \text{ s.t. } V_M^{S_2}(s, b), s' \text{ is a positive inst. of CREACH(1-VASS)}\}$ 
13:     $\langle\langle \omega \rangle\rangle \psi_1 R \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid K_{M, s} \models E(V_{s' \in S_1} s') R(V_{s' \in S_2} s')\}$ 
14:     $\langle\langle \emptyset^b \rangle\rangle \psi_1 R \psi_2$ :  $S_1 := GMC(M, \psi_1); S_2 := GMC(M, \psi_2);$ 
           return  $\{s \in S \mid K_{M, s} \models A(V_{s' \in S_1} s') R(V_{s' \in S_2} s')\}$ 
15:   end case
16: end procedure

```

---

**Proof** Let  $M = (\{1\}, S, Act, 1, act, cost, \delta, L)$  be a resource-bounded CGS, and  $\phi$  be a formula in  $RB\pm ATL(\{1\}, 1)$ . Let us present Algorithm 1, a polynomial-time algorithm that computes the finite set  $\{s \in S \mid (M, s) \models \phi\}$  (by default,  $b \in \mathbb{N}$ )<sup>3</sup>.

By induction, one can show that  $(M, s) \models \phi$  iff  $s \in GMC(M, \phi)$ . Lemmas 5-7 are used to prove the soundness of the subroutines for U and R, with  $b \in \mathbb{N} \cup \{\omega\}$ , respectively. When the strategy modality is  $\langle\langle \emptyset^b \rangle\rangle$ , it behaves as the standard path quantifier A, which is reflected in Algorithm 1. As far as computational complexity is concerned,  $GMC(M, \phi)$  is computed with a recursion depth linear in the size of  $\phi$  and the control-state reachability and nontermination problems can be solved in polynomial time by Theorem 2 and 3. More precisely, for each occurrence of a subformula  $\psi$  of  $\phi$ ,  $GMC(M, \psi)$  can be computed only once, which guarantees the overall number of calls of the form  $GMC(M, \psi)$ : it is sufficient to take advantage of dynamic programming and to work with a table to remember the values  $GMC(M, \psi)$  already computed (omitted in the present algorithm). It is also worth observing that the instances we consider are polynomial in the sizes of  $M$  and  $\phi$ . Finally, we take advantage of the fact that the model-checking problem for CTL including R remains in PTIME (see, e.g., [DGL16, Chapter 7]).

Consequently, reasoning about a single resource in the Computation Tree Logic CTL comes at no extra computational cost. Hence, in principle we can verify specification such as formula (1) in Example 1 efficiently. Based on the correspondences established in Theorem 1, we immediately derive the following consequence.

**Corollary 5** *The model-checking problem for RBTL restricted to a single resource is PTIME-complete.*

## 4 Concluding Remarks

We investigated the complexity of the model-checking problem for Resource-bounded Alternating-time Temporal Logics. In particular, we established that  $RBTL^*$  and  $RB\pm ATL^*(\{1\}, r)$  can be understood as slight variants of the same logic. More importantly, we provided a unified view of the model-checking problems for  $RB\pm ATL$ , then proved that model checking  $RB\pm ATL(\{1\}, 1)$  is PTIME-complete. To do so, we designed original algorithms to solve the control-state reachability and nontermination problems for 1-VASS. Hence, as far as worst-case complexity is concerned, the model-checking problems for CTL and  $RB\pm ATL(\{1\}, 1)$  behave similarly.

The paper has not touched very much on the model-checking problem for  $RB\pm ATL^*$ , for which the main results are summarised in Table 2. Unlike  $RB\pm ATL$ , tight complexity bounds are known for all variations on the number of agents and resources. For at least two agents, the

<sup>3</sup>. We omit the case for the operator G as  $G\phi$  is logically equivalent to  $\perp R\phi$ .

$r \setminus  Ag $	$\infty$	<b>2</b>	<b>1</b>
$\infty$	in 2EXPTIME [ABDL18, Th. 7]		EXPSPACE-c. [ABDL18, Th. 8]
$\geq 1$	2EXPTIME-h. (from ATL*)		in PSPACE ([ABDL18, Cor. 2] & Th. 1) PSPACE-h. (from CTL*)

TABLE 2 – The complexity of model checking  $\text{RB}\pm\text{ATL}^*(Ag, r)$ .

model-checking problem is 2EXPTIME-complete. The upper bound comes from [ABDL18, Th. 7], whereas the lower bound follows from the 2EXPTIME-hardness of  $\text{ATL}^*$ , which is proved by using two agents only [AHK02]. On the other hand, the problem restricted to a single agent becomes EXPSPACE-complete for an unbounded number of resources [ABDL18, Th. 8]; while for a bounded number  $r$ , model checking  $\text{RB}\pm\text{ATL}^*({1}, r)$  is PSPACE-complete : the lower bound follows immediately from the PSPACE-hardness of the model-checking problem for  $\text{CTL}^*$ ; as for the upper bound, we derive it from the fact that model checking  $\text{RBTL}^*$  is in PSPACE [ABDL18, Cor. 2] and Theorem 1.

As far as future work is concerned, we plan to implement the PTIME algorithm for model checking  $\text{RB}\pm\text{ATL}({1}, 1)$ , possibly taking advantage of the very recent results in [ACP<sup>+</sup>19], and to investigate the complexity of other meaningful fragments of  $\text{RB}\pm\text{ATL}(Ag, r)$  for which tight bounds are unknown. The synthesis of parameters for the parameterised version of  $\text{RB}\pm\text{ATL}({1}, 1)$  (as well as for other fragments of  $\text{RB}\pm\text{ATL}^*(Ag, r)$ ) is also worth further investigation.

**Acknowledgment.** We would like to thank Michael Blondin (University of Sherbrooke) for pointing us to [RY86] and for insightful feedback, as well as the anonymous referees for their suggestions and comments. F. Belardinelli acknowledges the support of the ANR JCJC Project SVE-DaS (ANR-16-CE40-0021).

## Références

- [ABDL18] N. Alechina, N. Bulling, S. Demri, and B. Logan. On the complexity of resource-bounded logics. *Theoretical Computer Science*, 750 :69–100, 2018.
- [ABLN15] N. Alechina, N. Bulling, B. Logan, and H.N. Nguyen. On the boundary of (un)decidability : Decidable model-checking for a fragment of resource agent logic. In *IJCAI’15*, pages 1494–1501. AAAI Press, 2015.
- [ABLN17] N. Alechina, N. Bulling, B. Logan, and H.N. Nguyen. The virtues of idleness : A decidable fragment of resource agent logic. *Artificial Intelligence*, 245 :56–85, 2017.
- [ACP<sup>+</sup>19] S. Almagor, N. Cohen, G. Pérez, M. Shirmohammadi, and J. Worrell. Coverability in 1-VASS with disequality tests. <http://arXiv:1902.06576>, February 2019.
- [AdAG<sup>+</sup>01] R. Alur, L. de Alfaro, R. Grosu, T. Henzinger, A. Thomas, M. Kang, C. Kirsch, R. Majumdar F. Mang, and B-Y. Wang. jMocha : A model checking tool that exploits design structure. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE01)*, pages 835–836. IEEE, 2001.
- [AH09] M. Faouzi Atig and P. Habermehl. On Yen’s path logic for Petri nets. In *RP’09*, volume 5797 of *Lecture Notes in Computer Science*, pages 51–63. Springer, 2009.
- [AHK02] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5) :672–713, 2002.
- [AL18] N. Alechina and B. Logan. Resource logics with a diminishing resource. In *AAMAS’18*, pages 1847–1849. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018.
- [ALN<sup>+</sup>15] N. Alechina, B. Logan, H.N. Nguyen, F. Raimondi, and L. Mostarda. Symbolic model-checking for resource-bounded atl. In *AAMAS’15*, pages 1809–1810. International Foundation for Autonomous Agents and Multiagent Systems, 2015.
- [ALNR14] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi. Decidable model-checking for a resource logic with production of resources. In *ECAI’14*, pages 9–14, 2014.
- [ALNR17] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi. Model-checking for resource-bounded ATL with production and consumption of resources. *Journal of Computer and System Sciences*, 88 :126–144, 2017.
- [BF09] N. Bulling and B. Farwer. Expressing properties of resource-bounded systems : The logics  $\text{RBTL}^*$  and  $\text{RBTL}$ . In *CLIMA X*, volume 6214 of *Lecture Notes in Computer Science*, pages 22–45. Springer, 2009.
- [BF10] N. Bulling and B. Farwer. On the (Un)Decidability of Model-Checking Resource-Bounded Agents. In *ECAI’10*, pages 567–572, 2010.

- [BFG<sup>+</sup>15] M. Blondin, A. Finkel, S. Göller, C. Haase, and P. McKenzie. Reachability in two-dimensional vector addition systems with states is PSPACE-complete. In *LICS'15*, pages 32–43. ACM Press, 2015.
- [BS11] M. Blockelet and S. Schmitz. Model-checking coverability graphs of vector addition systems. In *MFCS'11*, volume 6907 of *Lecture Notes in Computer Science*, pages 108–119. Springer, 2011.
- [CHP07] K. Chatterjee, T. Henzinger, and N. Piterman. Strategy logic. In *CONCUR'07*, volume 4703 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2007.
- [CLL<sup>+</sup>18] W. Czerwinski, S. Lasota, R. Lazić, J. Leroux, and F. Mazowiecki. The Reachability Problem for Petri Nets is Not Elementary (extended abstract). *CoRR*, abs/1809.07115, 2018. To appear in *STOC'19*.
- [CLMM14] P. Cermák, A. Lomuscio, F. Mogavero, and A. Murano. MCMAS-SLK : A model checker for the verification of strategy logic specifications. In *CAV'14*, volume 8559 of *Lecture Notes in Computer Science*, pages 525–532. Springer, 2014.
- [CS14] J.B. Courtois and S. Schmitz. Alternating vector addition systems with states. In *MFCS'14*, volume 8634 of *Lecture Notes in Computer Science*, pages 220–231. Springer, 2014.
- [Dem13] S. Demri. On selective unboundedness of VASS. *Journal of Computer and System Sciences*, 79(5) :689–713, 2013.
- [DGL16] S. Demri, V. Goranko, and M. Lange. *Temporal Logics in Computer Science*. Cambridge University Press, 2016.
- [FLL<sup>+</sup>17] D. Figueira, R. Lazić, J. Leroux, F. Mazowiecki, and G. Sutre. Polynomial-space completeness of reachability for succinct branching VASS in dimension one. In *ICALP'17*, volume 80 of *LIPICs*, pages 119 :1–119 :14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
- [GHLT16] St. Göller, Ch. Haase, R. Lazic, and P. Totzke. A polynomial-time algorithm for reachability in branching VASS in dimension one. In *ICALP'16*, volume 55 of *LIPICs*, pages 105 :1–105 :13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.
- [JS07] P. Jančar and Z. Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *Information Processing Letters*, 104(5) :164–167, 2007.
- [KM69] Richard M. Karp and Raymond E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2) :147 – 195, 1969.
- [KNN<sup>+</sup>08] M. Kacprzak, W. Nabialek, A. Niewiadomski, W. Penczek, A. Pólrola, M. Szreter, B. Woźna, and A. Zbrzezny. Verics 2007 - a model checker for knowledge and real-time. *Fundamenta Informaticae*, 85(1) :313–328, 2008.
- [Lip76] R.J. Lipton. The reachability problem requires exponential space. Technical Report 62, Department of Computer Science, Yale University, 1976.
- [LM15] F. Laroussinie and N. Markey. Augmenting atl with strategy contexts. *Information and Computation*, (245) :98–123, 2015.
- [LMO08] F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, 4(2), 2008.
- [LQR15] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS : A model checker for the verification of multi-agent systems. *Software Tools for Technology Transfer*, 2015. <http://dx.doi.org/10.1007/s10009-015-0378-x>.
- [MMPV14] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning about strategies : On the model-checking problem. *ACM Transactions on Computational Logic*, 15(4) :34 :1–34 :47, 2014.
- [Pau02] M. Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1) :149–166, 2002.
- [POY04] D.J. Pym, P.W. O’Hearn, and H. Yang. Possible worlds and resources : the semantics of BI. *Theoretical Computer Science*, 315(1) :257–305, 2004.
- [Rac78] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2) :223–231, 1978.
- [RY86] L. Rosier and H.-C. Yen. A multiparameter analysis of the boundedness problem for vector addition systems. *Journal of Computer and System Sciences*, 32 :105–135, 1986.
- [Sch03] Ph. Schnoebelen. The complexity of temporal logic model checking. In *AIML'02*, pages 437–459. King’s College Publication, 2003.
- [Sch16] S. Schmitz. The complexity of reachability in vector addition systems. *SIGLOG News*, 3(1) :4–21, 2016.