Reasoning with a Bounded Number of Resources in ATL+

Francesco Belardinelli, Stéphane Demri

Reasoning with a Bounded Number of Resources in ATL$^+$

Francesco Belardinelli[1] and **Stéphane Demri**[2]

Abstract. The resource-bounded alternating-time temporal logic RB±ATL combines strategic reasoning with reasoning about resources. Its model-checking problem is known to be 2EXPTIME-complete (the same as its proper extension RB±ATL*). Several fragments have been identified to lower the complexity.

In this work, we consider the variant RB±ATL$^+$ which permits Boolean combinations of path formulae starting with single temporal operators, but restricted to a single resource, providing an interesting trade-off between temporal expressivity and resource analysis. We show that the model-checking problem for RB±ATL$^+$ restricted to a single agent and a single resource is Δ_2^{p}-complete, hence the same as for CTL$^+$. In this case reasoning about resources comes at no extra computational cost. Furthermore, we show that, with an arbitrary number of agents and a fixed number of resources, the problem can be solved in EXPTIME using a Turing reduction to the parity game problem for alternating vector addition systems with states.

1 Introduction

Reasoning about resources with ATL-like logics. Reasoning about multi-agent systems (MAS) in which autonomous agents can interact and perform actions to reach (joint or individual) goals have benefited from the model-checking approach with the development of ATL-like logics (see, e.g., [11, 10]). The need for managing resources in MAS has been identified quite early and many logical formalisms based on ATL ("Alternating-time temporal logic") were introduced to endow actions with consumption or production of resources [13, 34, 14, 5, 3]. Unsurprisingly, dealing with resources can lead to high complexity, even undecidable model-checking problems, which is best illustrated in [13, 14, 3]. Still decidable resource logics have been identified, for instance by allowing only resource consumption [8]. The challenge for years has been to design resource logics, general enough to allow consumption and production of resources, while having a decidable model-checking, possibly low complexity, to allow the implementation in model-checking tools. A remarkable breakthrough occurred with the design of the logic RB±ATL, with production and consumption of resources, whose model-checking problem was shown decidable in [5]. Later on, fragments of RB±ATL have been designed with relatively low complexity: with a single agent and a single resource the PTIME upper bound was shown in [12] (see also [6, 7]).

When RB±ATL$^+$ comes into play. The model-checking problem for RB±ATL is shown 2EXPTIME-complete in [2] by using subroutines to solve decision problems for alternating vector addition systems with states (AVASS), see, e.g., [18, 26, 1]. The 2EXPTIME-completeness is also pushed further for RB±ATL*, the extension of RB±ATL in which path formulae are arbitrary LTL formulae, in the same way the temporal logic CTL* extends CTL (see, e.g. [19, Chapter 7]). Consequently, every logic between RB±ATL and RB±ATL* admits a 2EXPTIME-complete model-checking problem and this also applies to RB±ATL$^+$, the extension of RB±ATL in which path formulae are arbitrary Boolean combinations of LTL formulae of temporal depth one, in the same way the temporal logic CTL$^+$ extends CTL (see, e.g., [22]). Though the model-checking problem for CTL (resp. CTL$^+$, CTL*) is known to be PTIME-complete (resp. Δ_2^{p}-complete, PSPACE-complete) and the one for ATL (resp. ATL$^+$, ATL*) PTIME-complete [11] (resp. PSPACE-complete [15, 24], 2EXPTIME-complete [11]), there is no difference in complexity for RB±ATL, RB±ATL$^+$ and RB±ATL*, as far as worst-case complexity of the whole logics is concerned.

Our motivations. To identify fragments of RB±ATL* of tractable complexity, with the intention to implement model-checking algorithms in existing tools [30], it may happen that RB±ATL, RB±ATL$^+$ and RB±ATL* have distinct complexity when their fragments are considered, typically by restricting the number of resources or the number of agents. It is commonly accepted that the design of fragments is essential to find a good compromise between the expressive power of the logical language and the computational complexity of the reasoning tasks. That is why the quest for subproblems of RB±ATL* obtained by limiting the temporal expressivity, or the number of resources is important for practical use. In this paper, we study the complexity of the model-checking problem for RB±ATL$^+$ (strict syntactic fragment of RB±ATL*) restricted to a bounded number of resources, with a special attention to the case with a single resource. It is also worth noting that as the path formulae in CTL$^+$ (and therefore in ATL$^+$ and in RB±ATL$^+$) are Boolean combinations of LTL formulae of temporal depth one, this provides a generic and natural way to design linear-time constraints. Surprisingly, this specific viewpoint considering fragments of RB±ATL$^+$ will happen to be fruitful, not only for the identification of interesting fragments of lower complexity, but also in terms of proof techniques to solve the model-checking problem.

Our contribution. First, we introduce the logic RB±ATL$^+$, similarly to the way ATL$^+$ (resp. CTL$^+$) extends ATL (resp. CTL), and we show interesting features in terms of expressivity. We prove that the model-checking problem for RB±ATL$^+$ restricted to a single agent and to a single resource is Δ_2^{p}-complete (Theorem 2). The lower bound is inherited from CTL$^+$ [28], whereas the upper bound Δ_2^{p} combines ingredients for model-checking CTL$^+$ with procedures in PTIME for decision problems for vector addition systems with states (VASS) [27] restricted to one counter, newly introduced in this paper. This analysis substantially generalises [12, Section 3]. In particular, we introduce the new generalised control state reachability problem for VASS and we show that it is in PTIME, when restricted to VASS with one counter (Theorem 1). The same problem with

[1] Department of Computing, Imperial College London, UK and Laboratoire IBISC, Université d'Evry, France.
[2] LSV, CNRS, ENS Paris-Saclay, Université Paris-Saclay, France.

an unrestricted number of counters can be shown to be EXPSPACE-complete, but its PTIME subproblem with one counter is instrumental to get the Δ_2^{p}-completeness result. Since model checking CTL^+ is also Δ_2^{p}-complete, the enhanced expressivity provided by reasoning about resource comes at no extra computation cost.

Furthermore, we show that the model-checking problem for $\mathsf{RB}{\pm}\mathsf{ATL}^+$ restricted to r resources (for a fixed r) can be solved in EXPTIME (Theorem 3) by reduction to the parity game problem for AVASS [17]. Remarkly, the number of locations in the target AVASS is exponential in the input size, which makes a substantial difference with what is done in [2] for $\mathsf{RB}{\pm}\mathsf{ATL}^*$. Hence, the model-checking problem for $\mathsf{RB}{\pm}\mathsf{ATL}^+$ restricted to one resource is PSPACE-hard (inherited from ATL^+ [15, 24]) and in EXPTIME. Apart from the new complexity results that improve our understanding of fragments of $\mathsf{RB}{\pm}\mathsf{ATL}^*$, we present reductions to the parity game problem to AVASS that can be of interest for its own sake.

2 Preliminaries

Our presentation follows closely [2, 12]. In the rest of the paper, $\mathbb{N}$ (resp. $\mathbb{Z}$) is the set of natural numbers (resp. integers) and for $m, m' \in \mathbb{Z}$, $[m, m']$ is the set $\{j \in \mathbb{Z} \mid m \leq j \leq m'\}$. For a finite or infinite sequence $u \in X^* \cup X^\omega$ of elements in some set X, we write u_i or $u[i]$ for the $(i+1)$-th element of u, i.e., $u = u_0 u_1 \ldots$. For $i \geq 0$, $u_{\leq i} = u_0 u_1 \ldots u_i$ is the prefix of u of length $i+1$, while $u_{\geq i} = u_i u_{i+1} \ldots$ is a suffix of u. 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 $\mathsf{RB}{\pm}\mathsf{ATL}^+$: strategic reasoning and resources

Syntax. In the rest of the paper we assume a countably infinite set AP of atomic propositions. Given a finite non-empty set Ag of agents and a number $r \geq 1$ of resources (often called "resource types" in the literature), we write $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, r)$ to denote the resource-bounded logic with agents from Ag and r resources.

Definition 1 ($\mathsf{RB}{\pm}\mathsf{ATL}^+$ formulae) *State formulae ϕ and path formulae Ψ are built according to the following BNF, where $p \in \mathrm{AP}$, $A \subseteq Ag$, and $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$:*

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi \qquad \Psi ::= \neg\Psi \mid \Psi \wedge \Psi \mid \mathsf{X}\,\phi \mid \phi\,\mathsf{U}\,\phi$$

The formulae in $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, r)$ are all and only the state formulae. $\mathsf{RB}{\pm}\mathsf{ATL}(Ag, r)$ is the fragment of $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, r)$ in which path formulae have no outermost Boolean connective but with a new clause $\mathsf{G}\,\phi$. We write $\mathsf{RB}{\pm}\mathsf{ATL}^+(r)$ to denote the set of formulae in $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, r)$, for some non-empty set Ag of agents.

As will become clear, $\mathsf{RB}{\pm}\mathsf{ATL}^+$ extends ATL^+ from [11] by indexing the strategic operator $\langle\!\langle A \rangle\!\rangle$ with tuples $\vec{b}$, whose intuitive meaning is that coalition A can achieve their goal with an initial budget $\vec{b}$. The value ω accounts for an infinite supply of the resource. The linear-time operators X and U have their standard readings; while $\vee$, $\Rightarrow$, and the temporal operator *always* G are introduced as usual. For instance, $\langle\!\langle A^{(8,\omega)} \rangle\!\rangle((p_1\,\mathsf{U}\,q_1) \Rightarrow \mathsf{G}\,r_1 \wedge (p_2\,\mathsf{U}\,q_2) \Rightarrow \mathsf{G}\,r_2)$ is an $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, 2)$ formula but not an $\mathsf{RB}{\pm}\mathsf{ATL}(Ag, 2)$ formula.

Semantics. We interpret the formulae by using resource-bounded concurrent game structures [9, 2], endowed a weight function that assigns an integer to every action (herein understood as a gain).

Definition 2 (RB-CGS) *A* resource-bounded concurrent game structure *is a tuple $M = \langle Ag, r, S, Act, act, wf, \delta, L \rangle$ such that*

- *Ag is the finite, non-empty set of agents (by default $Ag = [1, k]$ for some $k \geq 1$); $r \geq 1$ is the number of* resources*;*
- *S is a finite, non-empty set of* states*;*
- *Act is a finite, non-empty set of* actions*;*
- *$act : S \times Ag \to \wp(Act) \setminus \{\emptyset\}$ is the* protocol function*;*
- *$wf : S \times Ag \times Act \to \mathbb{Z}^r$ is the (partial)* weight function*; that is, $wf(s, a, \mathtt{a})$ is defined only when $\mathtt{a} \in act(s, a)$;*
- *$\delta : S \times (Ag \to Act) \to S$ is the (partial)* transition function *such that δ is defined for state s and joint action $f : Ag \to Act$ only if for every agent $a \in Ag$, $f(a) \in act(s, a)$;*
- *$L : \mathrm{AP} \to \wp(S)$ is the* labelling function*.*

Intuitively, a resource-bounded CGS describes the interactions of a group Ag of agents, who perform the actions in Act according to the protocol function act. The execution of a joint action generates a transition in the system, as specified by the transition function δ. Moreover, on each transition the values of the r resources are updated according to the weight of the joint action. It is worth noting that in Definition 2 we do not assume the existence of an `idle` action (with zero weight). It is introduced in [5, 7] and it is often advantageous in terms of computational complexity (see e.g. [4, 2]). In this paper, we do not rely on the action `idle` to establish our complexity results, but of course, nothing prevents us from having such a distinguished action in our instances of the model-checking problem.

An RB-CGS M is *finite* whenever L is restricted to a finite subset of AP. The size $|M|$ of a finite RB-CGS 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 a state $s \in S$, a *joint action available to A in s* is a map $f : A \to Act$ such that for every agent $a \in A$, $f(a) \in act(s, a)$. The set of all such joint actions is denoted by $D_A(s)$. 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 $\mathtt{out}(s, f)$ to denote the set of *immediate outcomes*: $\mathtt{out}(s, f) \stackrel{\text{def}}{=} \{\delta(s, g) \mid \text{for some } g \in D_{Ag}(s), f \sqsubseteq g\}$. Given a joint action $f \in D_A(s)$ and $s \in S$, the weight of a transition from s by f (w.r.t. coalition A) is defined as $wf_A(s, f) \stackrel{\text{def}}{=} \sum_{a \in A} wf(s, a, f(a))$. A *computation* λ is a finite or infinite sequence $s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ such that for all $0 \leq i < |\lambda| - 1$ we have $s_{i+1} = \delta(s_i, f_i)$. To provide a formal semantics to $\mathsf{RB}{\pm}\mathsf{ATL}^+$ based on RB-CGS, we need a notion of resource-bounded strategy for the interpretation of the strategic operator $\langle\!\langle A^{\vec{b}} \rangle\!\rangle$.

Definition 3 (Strategy) *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 $F_A(s_0 \xrightarrow{f_0} s_1 \ldots s_{n-1} \xrightarrow{f_{n-1}} s_n) \in D_A(s_n)$.*

A computation $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ respects strategy F_A *iff for all $i < |\lambda|$, $s_{i+1} \in \mathtt{out}(s_i, F_A(s_0 \xrightarrow{f_0} s_1 \ldots s_{n-1} \xrightarrow{f_{i-1}} s_i))$.*

A computation λ that respects F_A is *maximal* if it cannot be extended further while respecting F_A. Herein, maximal computations starting in state s and respecting F_A are infinite, and we denote the set of all such maximal computations by $\mathtt{Comp}(s, F_A)$.

Given a bound $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$ and a computation $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ in $\mathtt{Comp}(s, F_A)$, let the *resource availability $\vec{v}_i$ at step $i < |\lambda|$* be defined as: $\vec{v}_0 = \vec{b}$ and for all $i \geq 0$, $\vec{v}_{i+1} = wf_A(s_i, f_i) + \vec{v}_i$ (assuming $n + \omega = \omega$ for every $n \in \mathbb{Z}$). Then, λ is *$\vec{b}$-consistent* iff for all $i < |\lambda|$, $\vec{v}_i \in (\mathbb{N} \cup \{\omega\})^r$. Since the resource availability depends only on the agents in A, this is called the *proponent restriction condition* (see, e.g., [3, 2]). A *$\vec{b}$-strategy F_A with*

respect to s is a strategy such that all the maximal computations in $\mathtt{Comp}(s, F_A)$ are $\vec{b}$-consistent. We define the satisfaction relation $\models$ for a state $s \in S$, an infinite computation λ, $p \in \mathrm{AP}$, a state formula ϕ, and a path formula Ψ as follows (clauses for Boolean connectives are standard and thus omitted):

$(M, s) \models p$	iff	$s \in L(p)$
$(M, s) \models \langle\langle A^{\vec{b}} \rangle\rangle \Psi$	iff	for some $\vec{b}$-strategy F_A w.r.t. s, for all $\lambda \in \mathtt{Comp}(s, F_A)$, $(M, \lambda) \models \Psi$
$(M, \lambda) \models \mathsf{X}\,\phi$	iff	$(M, \lambda_{\geq 1}) \models \phi$
$(M, \lambda) \models \phi \,\mathsf{U}\, \phi'$	iff	for some $i \geq 0$, $(M, \lambda_{\geq i}) \models \phi'$, and for all $0 \leq j < i$, $(M, \lambda_{\geq j}) \models \phi$

In the sequel, we consider the following decision problem.

Definition 4 (Model Checking) *Let $k, r \geq 1$, ϕ be a formula in $\mathsf{RB}{\pm}\mathsf{ATL}^+([1,k], r)$, 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 $(M, s) \models \phi$.*

In the sequel, given a logic $\mathcal{L}$, we write MC($\mathcal{L}$) to denote the corresponding model-checking problem. Below, we summarize known complexity results for subproblems of MC($\mathsf{RB}{\pm}\mathsf{ATL}^+$) obtained by bounding the number of agents or resources and we identify the contributions made in this paper.

$r \backslash \lvert Ag \rvert$	∞	$\geq \mathbf{2}$	$\mathbf{1}$
∞	2EXPTIME-c. [2, Th. 2 and 3]		EXPSPACE-c. [2, Th. 4]
$\geq \mathbf{4}$	in EXPTIME **[this paper]** EXPTIME-h. [2, Th. 2 and 3]		PSPACE-c [2, Cor. 2] [15]
3	in EXPTIME **[this paper]** PSPACE-h. [15]		
2			
1	in EXPTIME **[this paper]** PSPACE-h. (from ATL^+ [15])		in Δ_2^{p} **[this paper]** Δ_2^{p}-h. (from CTL^+ [28])

Remark 1 *When $Ag = \{1\}$, the only strategy modalities are $\langle\langle Ag^{\vec{b}} \rangle\rangle$ and $\langle\langle \emptyset^{\vec{b}} \rangle\rangle$. Whereas $\langle\langle \emptyset^{\vec{b}} \rangle\rangle$ behaves as universal path quantifier A in CTL^+ (and the weights in the RB-CGS are ignored because of the proponent restriction condition), the modality $\langle\langle Ag^{\vec{b}} \rangle\rangle$ amounts to quantify existentially on a path satisfying a temporal goal (but this time, the weights are taken into account). Hence, $\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\}, r)$ can be thought of as a resource-bounded version of CTL^+ with r resources. Finally, to stick to the definition of the original version of $\mathsf{RB}{\pm}\mathsf{ATL}$ from [5] and to provide more flexibility to the specification language, we have kept the bound ω but the way to handle it consists in reducing the dimension (eliminate the components where it appears as it does not bring any quantitative constraint), which can be easily performed.*

Example 1 *We illustrate the formal machinery introduced so far with a toy example. We consider a scenario in which two drones, d_1 and d_2, have to deliver two parcels to addresses (X_1, Y_1) and (X_2, Y_2) on a $[-M, M]^2$ grid for some $M > 0$. We assume that they both start from location $(0, 0)$. At any time, each drone can choose between two modes: either it moves in one of the four directions ($\mathtt{N}$, $\mathtt{S}$, $\mathtt{E}$, $\mathtt{W}$) or it recharges its battery through a solar panel ($\mathtt{charge}$), but it cannot do both actions at the same time. Moving around consumes one energy unit at every time step; whereas the drone can recharge one unit at a time. Each drone can carry one parcel at a time, and they can only pick them up from location $(0, 0)$.*

This scenario can be modelled as a resource-bounded CGS $M = \langle \{d_1, d_2\}, 1, [-M, M]^2, \{\mathtt{N}, \mathtt{S}, \mathtt{E}, \mathtt{W}, \mathtt{charge}\}, act, wf, \delta, L \rangle$ where in every state s, for $i \in \{1, 2\}$ and $\mathtt{D}_i \in \{\mathtt{N}, \mathtt{S}, \mathtt{E}, \mathtt{W}\}$:

- $act(s, d_i) = \{\mathtt{N}, \mathtt{S}, \mathtt{E}, \mathtt{W}, \mathtt{charge}\}$;
- $wf(s, d_i, \mathtt{D}_i) = -1$ *and* $wf(s, d_i, \mathtt{charge}) = +1$;
- $\delta(((x_1, y_1), (x_2, y_2)), (\mathtt{D}_1, \mathtt{D}_2)) = ((x_1', y_1'), (x_2', y_2'))$, *where (x_i', y_i') is obtained from (x_i, y_i) by decrementing/incrementing either x_i or y_i depending on $\mathtt{D}_i$ (notice that on the frontier, movement in some directions has no effect), and $\delta(s, (\mathtt{charge}, \mathtt{charge})) = s$.*
- $\mathrm{AP} = \{charge_1, charge_2, del_1, del_2\}$ *and $L(del_i) = \{(X_i, Y_i)\}$, while a state is labelled with $charge_i$ if drone d_i has charged as previous action (we can keep track of this information with an extra bit of memory in the system state).*

Even in such a simple scenario we can express interesting strategic properties in $\mathsf{RB}{\pm}\mathsf{ATL}^+(\{d_1, d_2\}, 1)$. Consider a budget $b = \max\{(|X_1| + |Y_1|), (|X_2| + |Y_2|)\}$ that allows any drone d to deliver a single parcel to the farthest address. We can check that in location $(0, 0)$ the following formulae are true:

1. *With budget b, every drone d_i, for $i \in \{1, 2\}$, can deliver parcel 1 and can deliver parcel 2, without charging in the meantime:*

$$\langle\langle \{d_i\}^b \rangle\rangle (\neg charge \,\mathsf{U}\, del_1) \wedge \langle\langle \{d_i\}^b \rangle\rangle (\neg charge \,\mathsf{U}\, del_2) \qquad (1)$$

2. *With budget $2b$, neither drone can deliver both parcel 1 and 2, without charging in the meantime (recall that drones have to pick parcels in location $(0, 0)$):*

$$\neg\langle\langle \{d_i\}^{2b} \rangle\rangle ((\neg charge \,\mathsf{U}\, del_1) \wedge (\neg charge \,\mathsf{U}\, del_2)) \qquad (2)$$

(1) belongs to $\mathsf{RB}{\pm}\mathsf{ATL}$, but we make use of Boolean combinations of goals and the expressivity of $\mathsf{RB}{\pm}\mathsf{ATL}^+$ to formalize (2).

2.2 Correspondence between RB-CGS and AVASS

In this section, we recall the notion of alternating vector addition systems with states (AVASS), as well as relevant related decision problems. Indeed, AVASS are closely related to the model-checking problem for $\mathsf{RB}{\pm}\mathsf{ATL}$-like logics, as explained below (see also [2]). First, we need to introduce some preliminaries.

A *binary tree* T, which may contain nodes with a single child, is a non-empty subset of $\{1, 2\}^*$ such that, for all $\mathfrak{n} \in \{1, 2\}^*$ and $i \in \{1, 2\}$, $\mathfrak{n} \cdot i \in T$ implies $\mathfrak{n} \in T$ and, $\mathfrak{n} \cdot 2 \in T$ implies $\mathfrak{n} \cdot 1 \in T$.

Definition 5 (AVASS [18]) *An* alternating VASS *is a tuple $\mathcal{A} = (Q, r, R_1, R_2)$ such that Q is a finite set of* locations*, $r \geq 0$ is the number of resources, R_1 is a finite subset of $Q \times \mathbb{Z}^r \times Q$ (*unary rules*) and R_2 is a subset of Q^3 (*fork rules*).*

Standard VASS [27] (known to be equivalent to Petri nets) are AVASS with no fork rules (i.e., $R_2 = \emptyset$). A *derivation skeleton* in $\mathcal{A}$ is a labelling $\mathcal{D} : T \to (R_1 \cup R_2 \cup \{\bot\})$ such that T is a binary tree, and if $\mathfrak{n}$ has one child (resp. has two children, is a leaf) in T, then $\mathcal{D}(\mathfrak{n}) \in R_1$ (resp. $\mathcal{D}(\mathfrak{n}) \in R_2$, $\mathcal{D}(\mathfrak{n}) = \bot$). A *derivation* in $\mathcal{A}$ based on $\mathcal{D}$ is a labelling $\hat{\mathcal{D}} : T \to Q \times \mathbb{Z}^r$ such that:

- if $\mathfrak{n}$ has one child $\mathfrak{n}'$ in T, $\mathcal{D}(\mathfrak{n}) = (q, \vec{u}, q')$ and $\hat{\mathcal{D}}(\mathfrak{n}) = (q, \vec{v})$, then $\hat{\mathcal{D}}(\mathfrak{n}') = (q', \vec{v} + \vec{u})$;
- if $\mathfrak{n}$ has two children $\mathfrak{n}'$ and $\mathfrak{n}''$ in T, $\mathcal{D}(\mathfrak{n}) = (q, q_1, q_2)$ and $\hat{\mathcal{D}}(\mathfrak{n}) = (q, \vec{v})$, then $\hat{\mathcal{D}}(\mathfrak{n}') = (q_1, \vec{v})$ and $\hat{\mathcal{D}}(\mathfrak{n}'') = (q_2, \vec{v})$.

So, the fork rules do not update the value of resources. Hence, there is an asymmetry between unary and fork rules. Unlike branching VASS (see, e.g., [38, 20, 23]), fork rules have no effect on the counter values. A derivation $\hat{\mathcal{D}}$ is *admissible* (or a *proof*) whenever only natural numbers occur in it.

$$\dfrac{\dfrac{\dfrac{\vdots}{(q_3,(9,9))}}{(q_2,(5,5))} \qquad \dfrac{\dfrac{\vdots \qquad \vdots}{(q_1,(3,8)) \quad (q_2,(3,8))}}{(q_0,(3,8))}}{(q_1,(5,5))}}{(q_0,(5,5))}$$
$$(q_1,(7,2))$$

By way of example, a proof can be found left (with the root at the bottom).

$R_1 = \{q_1 \xrightarrow{(-2,3)} q_0, q_2 \xrightarrow{(4,4)} q_3\}$

$R_2 = \{q_0 \to q_1, q_2\}$.

Hereafter, we consider the following decision problem pertaining to AVASS. Given an AVASS $\mathcal{A} = (Q, r, R_1, R_2)$, a *colouring* $\mathtt{col}$ is defined as a map $Q \to [0, \mathfrak{p}-1]$ for some number $\mathfrak{p} \geq 1$ of *colours*. The *parity game problem* is defined as follows:

Parity game problem for AVASS: given $\mathcal{A}$, q_0, $\vec{b} \in \mathbb{N}^r$ and $\mathtt{col}$, is there a proof with root labelled by $(q_0, \vec{b})$, all the maximal branches are infinite, and the maximal colour occurring infinitely often is even?

By [17, Corollary 5.7] on the parity energy game problem with initial credit, and by [1, Lemma 4] relating the parity game problem for single-sided VASS – a.k.a. AVASS – and the parity energy game problem, the former can be solved in time

$$(|Q| \times ||R_1||)^{2^{\mathcal{O}(r \times log(r+\mathfrak{p}))}} + \mathcal{O}(r \times log||\vec{b}||),$$

where $||\vec{b}|| \stackrel{\text{def}}{=} \max\{|\ \vec{b}[i]\ | :\ i \in [1,r]\}$ and $||R_1|| \stackrel{\text{def}}{=} \max\{||\vec{u}||\ :\ q \xrightarrow{\vec{u}} q' \in R_1\}$. When r and $\mathfrak{p}$ are bounded, the problem is therefore in EXPTIME. Indeed, the expression $2^{\mathcal{O}(r \times log(r+\mathfrak{p}))}$ and r are bounded whereas $||R_1||$ is at most exponential in the size of the input (the integers are encoded in binary).

We now briefly recall the correspondence established in [2] between strategies in RB$\pm$ATL-like logics and proofs in AVASS. As we use it in the paper, we explain it in detail for the reader's benefit. Below, we consider AVASS with fork rules in $\bigcup_{\beta \geq 2} Q^\beta$ (arbitrary arity), and where the proofs are trees with nodes labelled by elements in $Q \times \mathbb{N}^r$. Let M be a finite RB-CGS, $A \subseteq Ag$ be a coalition, and $s^\star$ be one of its states. We construct the AVASS $\mathcal{A}_{M,A,s^\star}$ such that the set of computations in M starting in $s^\star$ and respecting strategy F_A corresponds to the derivation skeleton whose root is labelled by a unary rule with first state $s^\star$. Hence, we leverage on the fact that a $\vec{b}$-strategy generates a set of maximal computations that can be arranged as a finitely-branching tree with only infinite branches, and such infinite trees can be viewed precisely as infinite proofs on $\mathcal{A}_{M,A,s^\star}$. Given $M = (Ag, r, S, Act, act, wf, \delta, L)$ and $s^\star \in S$, the AVASS $\mathcal{A}_{M,A,s^\star} \stackrel{\text{def}}{=} (Q, r, R_1, R_2)$ is built as follows:

$$\begin{aligned} Q \stackrel{\text{def}}{=}\ & \{s^\star\} \cup \{(s,f)\ |\ s \in S,\ f \in D_A(s)\} \cup \\ & \{(g,s')\ |\ s', s'' \in S,\ g \in D_{Ag}(s''),\ \delta(s'',g) = s'\}. \end{aligned}$$

Every location in Q is attached to a state in S and contains a finite piece of information: in (s,f), f is a joint action available to A implementing its strategy whereas in (g,s'), we remember the global joint action to reach s'.

- The set R_1 of unary rules contains the following elements:
  - $(s^\star, wf_A(s^\star, f), (s^\star, f))$, for all $f \in D_A(s^\star)$. Indeed, the locations in Q attached to $s^\star$ have a special treatment.
  - $((g,s), wf_A(s,f), (s,f))$, for all $(g,s) \in Q$ and $f \in D_A(s)$.
- The set R_2 of fork rules contains the following elements.
  - For $(s,f) \in Q$, let $\{(g_1,s_1), \ldots, (g_\alpha, s_\alpha)\} = \{(g,s') \in S\ |\ s' = \delta(s,g),\ g \in D_{Ag}(s),\ f \sqsubseteq g\}$. This set is non-empty because the protocol function always returns a non-empty set of actions. We add the α-ary fork rule $((s',f), (g_1,s_1), \ldots, (g_\alpha, s_\alpha))$. The joint actions available for the opponent coalition $(Ag \setminus A)$ determine which locations (g_1,s_1), …, (g_α, s_α) can be reached. It is the proponent restriction condition that guarantees that using fork rules in this place (i.e. without updating the counter values) is correct.

Given an infinite computation $\lambda = s_0 \xrightarrow{g_1} s_1 \xrightarrow{g_2} s_2 \ldots$ starting in $s^\star = s_0$ and respecting F_A, we can associate it with an infinite sequence (which we call an *extended computation*)

$$\mathtt{ext}(\lambda, F_A) \stackrel{\text{def}}{=} s_0 \xrightarrow{\vec{u}_0} (s_0,f_0) \to (g_1,s_1) \xrightarrow{\vec{u}_1} (s_1,f_1) \to \cdots$$

where $s_0 = s^\star$, and for all $n \geq 0$, $F_A(s_0 \xrightarrow{g_1} s_1 \ldots \xrightarrow{g_n} s_n) = f_n$ and $wf_A(s_n, f_n) = \vec{u}_n$.

Further, transitions in M can be viewed as triples (s', g, s'') such that $\delta(s', g) = s''$, also written as $s' \xrightarrow{g} s''$. The finite set of such transitions is denoted by Σ_M. An infinite computation $\lambda = s_0 \xrightarrow{g_1} s_1 \xrightarrow{g_2} s_2 \ldots$ can be represented by the ω-word $(s_0 \xrightarrow{g_1} s_1) \cdot (s_1 \xrightarrow{g_2} s_2) \cdot (s_2 \xrightarrow{g_3} s_3) \cdots$ in Σ_M^ω. Given an infinite branch of the proof in $\mathcal{A}_{M,A,s^\star}$ corresponding to the extended computation $s_0 \xrightarrow{\vec{u}_0} (s_0,f_0) \to (g_1,s_1) \xrightarrow{\vec{u}_1} (s_1,f_1) \to \cdots$, its Σ_M-*projection* is defined as the sequence $(s_0 \xrightarrow{g_1} s_1) \cdot (s_1 \xrightarrow{g_2} s_2) \cdot (s_2 \xrightarrow{g_3} s_3) \cdots$ Formal correspondences between M and $\mathcal{A}_{M,A,s^\star}$ are below.

Proposition 1 *([2, Lemma 4]) Let* $\mathrm{L} \subseteq \Sigma_M^\omega$ *and* $\vec{b} \in \mathbb{N}^r$. *There is a* $\vec{b}$*-strategy* F_A *w.r.t.* $s^\star$ *in* M *such that the set of computations* $\mathtt{Comp}(s^\star, F_A)$ *is included in* L *iff there is a proof in* $\mathcal{A}_{M,A,s^\star}$ *with root labelled by* $(s^\star, \vec{b})$, *every maximal branch is infinite and its* Σ_M*-projection is in* L.

By way of example, let S_ϕ be the set of states in S satisfying the (state) formula ϕ (S is from some RB-CGS M) and L_ϕ be the restriction of Σ_M^ω to ω-words involving only states from S_ϕ. By Proposition 1, $(M, s^\star) \models \langle\langle A^{\vec{b}} \rangle\rangle\, \mathsf{G}\, \phi$ iff there is a proof in $\mathcal{A}_{M,A,s^\star}$ restricted to locations involving only states in S_ϕ whose root is labelled by $(s^\star, \vec{b})$ and every maximal branch is infinite.

3 Model-checking Problem for RB$\pm$ATL$^+$ $(\{1\}, 1)$

To show that MC(RB$\pm$ATL$^+(\{1\},1)$) is PTIME-complete, we introduce a new decision problem for vector addition systems with states (VASS), and we show that when restricted to a single counter, it can be solved in PTIME, generalising Theorems 3.3 and 3.5 from [12]. We recall that a *VASS* is an AVASS with an empty set R_2 of fork rules: it is a structure $V = (Q, r, R)$, where R is a finite subset of $Q \times \mathbb{Z}^r \times Q$. A *configuration* of a VASS V is defined as a pair $(q, \vec{x}) \in Q \times \mathbb{N}^r$. 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 $\vec{x'} = \vec{u} + \vec{x}$. A *run* is defined as a sequence $(q_0, \vec{x}_0) \xrightarrow{t_1} (q_1, \vec{x}_1) \xrightarrow{t_2} (q_2, \vec{x}_2) \cdots$ A *simple run* ρ is a finite run such that no control state appears twice. A *simple path* is a sequence of contiguous transitions $t_1 \ldots t_k$ such that no control state occurs more than once. A *simple loop* is a sequence of contiguous transitions $t_1 \ldots t_k$ 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. An r-VASS is a VASS with $r \geq 1$ counters. By convention, a 0-VASS is a finite digraph.

3.1 Generalised control state reachability problem

We now present a new decision problem on VASS that plays a crucial role in solving MC(RB±ATL$^+$($\{1\}$, 1)).
Generalised control state reachability problem GREACH(VASS):

Input: a VASS V, $(q_0, \vec{x}_0)$, a sequence $(X_1, h_1), \ldots, (X_\alpha, h_\alpha)$ for some $\alpha \geq 0$ s.t. $X_\alpha \subseteq \cdots \subseteq X_1 \subseteq Q$ and $\{h_1, \ldots, h_\alpha\} \subseteq Q$.
Question: is there an infinite run $(q_0, \vec{x}_0) \to (q_1, \vec{x}_1) \to (q_2, \vec{x}_2) \cdots$ such that there are $0 < i_1 < i_2 < \cdots < i_\alpha$ with for all $j \in [1, \alpha]$, $q_{i_j} = h_j$ and $\{q_k \mid k < i_j\} \subseteq X_j$?

Here is an illustration of the witness run when $\alpha = 2$.

$$\overbrace{(q_0, \vec{x_0}) \cdots (q_{i_1-1}, \vec{x_{i_1-1}})}^{\{q_0,\ldots,q_{i_1-1}\} \subseteq X_1} \to (q_{i_1}, \vec{x_{i_1}}) \cdots \underbrace{(q_{i_2-1}, \vec{x_{i_2-1}}) \to (q_{i_2}, \vec{x_{i_2}}) \to \cdots}_{\{q_0,\ldots,q_{i_2-1}\} \subseteq X_2}$$

When $\alpha = 0$, the question is about the existence of an infinite run from $(q_0, \vec{x}_0)$, which is in PTIME when $r = 1$ (see, e.g., [12]). Similarly, GREACH(VASS) restricted to 0-VASS (say GREACH(0-VASS)) is NLOGSPACE-complete as the Graph Accessibility Problem (GAP) can be reduced to it and the NLOGSPACE upper bound can be established by showing that a positive instance requires the existence of a path of length at most $|Q| \times (\alpha+1)$ whose constraints can be checked on-the-fly in NLOGSPACE. Unlike for 0-VASS, the addition of counters forbids a similar reduction of an instance of GREACH(VASS) to reachability questions on finite graphs.

We introduce a few more notions. Given an RB-CGS $M = (\{1\}, S, Act, 1, act, wf, \delta, L)$ with a single agent and a single resource, let us define the 1-VASS $V_M = (S, 1, R_V)$ such that $q \xrightarrow{u} q' \in R_V$ iff there is some action $\mathtt{a} \in act(q, 1)$ such that $\delta(q, \mathtt{a}) = q'$ and $wf(q, 1, \mathtt{a}) = u$. Similarly, we write $K_M = (S, R, L_K)$ to denote the Kripke-style structure such that $q \; R \; q'$ iff there is some action $\mathtt{a} \in act(q, 1)$ such that $\delta(q, \mathtt{a}) = q'$ and $L_K = L$. We introduce Kripke-style structures as the modality $\langle\langle\{1\}^\omega\rangle\rangle$ amounts to forgetting about weights in M, and therefore M can be understood as the Kripke-style structure K_M, and model checking reduces to CTL$^+$ model checking. Similarly, the strategy modality $\langle\langle\emptyset^b\rangle\rangle$ behaves as the universal path quantifier A in weight-free transition systems, and therefore model-checking reduces again to CTL$^+$ model-checking. 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/states in S_1 only.

Let V, $(q_0, \vec{x}_0)$ and (X_1, h_1), …, (X_α, h_α) be an instance of GREACH(1-VASS). W.l.o.g., we assume that $q_0 \neq h_1$. If $\alpha = 0$, then it is a positive instance of GREACH(1-VASS) iff V, $(q_0, \vec{x}_0)$ is a positive instance of the non-termination problem for 1-VASS, known to be in PTIME (see, e.g., [12, Theorem 3.5]). When $\alpha > 0$, one can show that the instance is positive iff either (A) or (B) holds.

(A) The following conditions hold:

(a) For $N \geq 0$, there is a run $(q_0, z_0) \cdots (q_n, z_n)$ with $(q_0, z_0) = (q_0, x_0)$, $q_n = h_1$, $z_n \geq N$, and $\{q_0, \ldots, q_{n-1}\} \subseteq X_1$.

(b) V', h_1 and (X_2, h_2), …, (X_α, h_α) is a positive instance of GREACH(0-VASS) where V' is the directed graph underlying V (integers on transitions are simply removed).

(c) There a simple path from h_α to some location q' and there is a simple loop from q' with positive overall effect.

(B) (a) above cannot be satisfied and there is a simple run $(q_0, x_0), \ldots, (q_k, x_k)$ in $V^{X_1 \cup \{h_1\}}$ with $q_k = h_1$ and V, (h_1, x_{max}) and (X_2, h_2), …, (X_α, h_α) is a positive instance of GREACH(1-VASS) with x_{max} the maximal counter value for reaching h_1 among all the simple runs from (q_0, x_0).

The condition "(A) or (B)" can be rewritten as "((a) and Φ) or (not (a) and Φ')", which implies "Φ or Φ'" but is not logically equivalent to it. Moreover, Φ' makes sense only when (a) does not hold, whence the current form of (A) and (B).

Lemma 1 *Let V, (q_0, x_0) and (X_1, h_1), …, (X_α, h_α) be an instance of GREACH(1-VASS) with $\alpha > 0$. It is a positive instance iff either (A) or (B) holds.*

We briefly explain why the characterisation is correct and leads to PTIME. When (a) holds, one can reach the control state h_1 with a counter value as large as we want. The constraints from the subsequence (X_2, h_2), …, (X_α, h_α) induce only reachability questions in the underlying directed graph V'. Otherwise, when (a) cannot be satisfied, the instance is positive only if there is a run $\rho = (q_0, x_0), \ldots, (q_k, x_k)$ in $V^{X_1 \cup \{h_1\}}$ with $q_k = h_1$ and V, (h_1, x_k) and (X_2, h_2), …, (X_α, h_α) is a positive instance of GREACH(1-VASS). However, not (a), we can assume that ρ is a simple run and the best we can do is to pick x_k equal to the maximal counter value for reaching h_1 among all the simple runs from (q_0, x_0) visiting only states in X_1. Checking (a), (b) or (c) can be done in PTIME (see the proof of [12, Theorem 3.3]) using dynamic programming. Similarly, computing x_{max} in (B) can be done in PTIME too.

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

The bound PTIME in Theorem 1 is a drastic drop compared to the complexity for GREACH(VASS). Indeed, GREACH(VASS) is EXPSPACE-hard as the control-state reachability problem for VASS can be reduced to it (and then we use [29]). GREACH(VASS) is definitely in EXPSPACE by [25, Theorem 5.4] dealing with the linear-time μ-calculus on VASS. Besides, for all $r \geq 1$, model-checking r-VASS with linear-time μ-calculus has been shown in PSPACE [25, Theorem 4.1] (PSPACE-hardness still holds with a unique counter, inherited from LTL model-checking [36]). Herein, we establish that GREACH(1-VASS) behaves even better as it can be solved in polynomial time only.

3.2 The model-checking algorithm

To solve MC(RB±ATL$^+$($\{1\}$, 1)), there is an essential case to consider, namely how to verify whether $(M, s) \models \langle\langle\{1\}^b\rangle\rangle\Psi$ (this is the key case that needs to be solved in a satisfactory way, complexity-wise). With a single agent in M, this amounts to checking the existence of a computation λ starting from s, with initial budget $b \in \mathbb{N}$, satisfying Ψ. This is the place where GREACH(1-VASS) helps.

Lemma 2 *Let Ψ be a Boolean formula built over path formulae of the form $\mathsf{X}\, p$ or $p \,\mathsf{U}\, q$, where p, q are propositional variables. Let M be an RB-CGS, s be one of its states and $b \in \mathbb{N}$. The problem of checking $(M, s) \models \langle\langle\{1\}^b\rangle\rangle\Psi$ (in RB±ATL$^+$($\{1\}$, 1)) is in* NP.

Hence, the whole model-checking algorithm for RB±ATL$^+$($\{1\}$, 1) can be shown in Δ_2^{p} as it uses a polynomial amount of instances of the problem in Lemma 2. Each instance can be solved in NP since GREACH(1-VASS) is in PTIME.

Theorem 2 MC(*RB±ATL$^+$*($\{1\}$, 1)) *is Δ_2^{p}-complete.*

Proof: (**sketch.**) As regards the lower bound, it follows from the Δ_2^{p}-hardness of MC(CTL$^+$) [28]. As for the upper bound, let $M = (\{1\}, 1, S, Act, act, wf, \delta, L)$ be an RB-CGS, and ϕ be a formula in

$\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1)$. Let us present Algorithm 1 that computes the finite set $\{s \in S \mid (M,s) \models \phi\}$, where we assume that $b \in \mathbb{N}$ and Ψ is a Boolean formula in NNF built over atomic path formulae of the form $\mathsf{X}\,\phi$ or $\phi\,\mathsf{U}\,\phi'$, for state formulae ϕ, ϕ'. Moreover, let us assume that the maximal state formulae occurring in Ψ are $\phi_1, \ldots, \phi_N$ and the model-checking algorithm has already determined that they hold true exactly on the states in $S_1^\star, \ldots, S_N^\star$, respectively. Let $\Psi^\star$ be the formula obtained from Ψ by replacing each ϕ_i by a fresh propositional variable p_i, and $M^\star$ be the RB-CGS obtained from M by modifying the labelling function as follows: $L^\star(p_i) \stackrel{\text{def}}{=} S_i^\star$. We have the following equivalences: $(M,s) \models \langle\!\langle \emptyset^b \rangle\!\rangle \Psi$ iff $(K_{M^\star}, s) \models \mathsf{A}\Psi^\star$ in CTL^+, and $(M,s) \models \langle\!\langle \{1\}^\omega \rangle\!\rangle \Psi$ iff $(K_{M^\star}, s) \models \mathsf{E}\Psi^\star$ in CTL^+. Concerning the case $(M,s) \models \langle\!\langle \{1\}^b \rangle\!\rangle \Psi$ with $b \in \mathbb{N}$, we have $(M,s) \models \langle\!\langle \{1\}^b \rangle\!\rangle \Psi$ iff $(M^\star, s) \models \langle\!\langle \{1\}^b \rangle\!\rangle \Psi^\star$, which can be checked in NP by Lemma 2. By structural induction, one can show that $(M,s) \models \phi$ iff $s \in \mathrm{MC}(M,\phi)$. As far as complexity is concerned, $\mathrm{MC}(M,\phi)$ is computed with a recursion depth linear in the size of ϕ and a polynomial number of NP requests. More precisely, for each occurrence of a subformula ψ of ϕ, $\mathrm{MC}(M,\psi)$ can be computed only once, which guarantees the overall number of calls of the form $\mathrm{MC}(M,\psi)$: it is sufficient to take advantage of dynamic programming and to work with a table to remember the values $\mathrm{MC}(M,\psi)$ already computed (omitted in the present algorithm). We recall that Δ_2^{p}, a.k.a. $\mathrm{PTIME}^{\mathrm{NP}}$, is precisely the class of problems that can be solved in polynomial time with an oracle in NP. We also take advantage of the fact that the model-checking problem for CTL^+ is in Δ_2^{p} (see, e.g., [19]). □

Algorithm 1 – $\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1)$ model checking –

1: **procedure** $\mathrm{MC}(M,\phi)$
2: **case** ϕ **of**
3: p: **return** $\{s \in S \mid s \in L(p)\}$
4: $\neg\psi$: **return** $S \setminus \mathrm{MC}(M,\psi)$
5: $\phi_1 \wedge \phi_2$: **return** $\mathrm{MC}(M,\phi_1) \cap \mathrm{MC}(M,\phi_2)$
6: $\langle\!\langle \emptyset^b \rangle\!\rangle \Psi / \langle\!\langle \emptyset^\omega \rangle\!\rangle \Psi$: **return** $\{s \mid (K_{M^\star}, s) \models \mathsf{A}\Psi^\star \text{ in } \mathsf{CTL}^+\}$
7: $\langle\!\langle \{1\}^\omega \rangle\!\rangle \Psi$: **return** $\{s \mid (K_{M^\star}, s) \models \mathsf{E}\Psi^\star \text{ in } \mathsf{CTL}^+\}$
8: $\langle\!\langle \{1\}^b \rangle\!\rangle \Psi$: **return** $\{s \mid (M^\star, s) \models \langle\!\langle \{1\}^b \rangle\!\rangle \Psi^\star$ with the algorithm in the proof of Lemma 2 $\}$
9: **end case**
10: **end procedure**

It is natural to wonder whether the model-checking problem for the Parity-Energy ATL, also called pe-ATL in [33], when the energy bound is bounded below and unbounded above can be used to analyse the complexity of $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(1))$. $\mathrm{MC}(\text{pe-}\mathsf{ATL})$ is in Δ_2^{p} [33] and it seems hopeless to take advantage of this bound[3] to solve efficiently $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(1))$ since $\mathrm{MC}(\mathsf{ATL}^+)$ is already PSPACE-hard [15]. A notable difference between $\mathsf{RB}{\pm}\mathsf{ATL}^+(1)$ and pe-ATL, is that the parity condition in pe-ATL is actually a global fairness condition for all the computations of the CGS.

Besides, we observe that the decision procedure we propose for solving the model-checking problem for $\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1)$ invokes the subroutine for the model-checking problem for CTL^+ with one counter, which we can solve optimally thanks to Lemma 2. The existence of a polynomial-time reduction between $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1))$ and the model-checking problem for CTL^+ is guaranteed, as both problems are Δ_2^{p}-complete. However, in Algorithm 1, we rather use a Turing reduction using as subroutines $\mathrm{MC}(\mathsf{CTL}^+)$ on finite transition systems and model-checking for CTL^+ on 1-VASS. The question about the encoding of $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1))$ into plain CTL^+ is certainly interesting and solving Condition (A) and (B) used in Lemma 1, actually amounts to detect graph-theoretical properties on the 1-VASS, so a direct encoding is plausible.

[3] We thank Dario Della Monica for checking that the NP upper bound stated in [33, Theorem 5.1] is actually an Δ_2^{p} upper bound in view of the algorithm developped in [33, Section 4]

4 Model-checking $\mathsf{RB}{\pm}\mathsf{ATL}^+(r)$

In [2], the 2EXPTIME upper bound for $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL})$ is obtained with a decision procedure calling subroutines to solve the non-termination and the control state reachability problems for AVASS on instances of the form $\mathcal{A}_{M,A,s^\star}$, as described in Section 2.2. On the other hand, to obtain the 2EXPTIME upper bound for $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^*)$ also in [2], the decision procedure calls a subroutine for solving the parity game problem for AVASS (see Section 2.2) but, in the worst-case, on systems with a doubly-exponential number of locations in the size of the input formula. Indeed, synchronised products are considered between deterministic parity automata (on ω-words) and systems of the form $\mathcal{A}_{M,A,s^\star}$. This is due to the fact that given an LTL formula, an equivalent deterministic parity automaton might have a doubly-exponential number of states in the size of the input LTL formula (see e.g. [37, 35]). Hence, even when the number of resources r is fixed (which is an assumption made in this section), solving $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(r))$ by using the method in [2], would lead to an 2EXPTIME upper bound.

In this section, we explain how to reduce one exponential by providing a decision procedure that solves $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(r))$ in exponential time, when r is fixed (this is optimal as soon as $r \geq 4$). Such an improvement can be explained by the fact that reasoning about LTL formulae of temporal depth one is usually simpler than for arbitrary LTL formulae (see, e.g., [21, Section 7.3]). Apart from our new EXPTIME bound, this section can be viewed as providing an alternative decision procedure for solving $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+)$ without going through the determinisation of Büchi automata, or as generalising the reduction used to decide $\mathsf{RB}{\pm}\mathsf{ATL}$ [2], but at the cost of building AVASS with an exponential number of locations (which is strictly more expensive than for $\mathsf{RB}{\pm}\mathsf{ATL}$, but strictly less than for $\mathsf{RB}{\pm}\mathsf{ATL}^*$). As CTL^+ is exponentially more succinct than CTL [39], which entails that some ATL^+ (resp. $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag,r)$) formulae can be exponentially more succinct than ATL (resp. $\mathsf{RB}{\pm}\mathsf{ATL}(Ag,r)$) formulae, this extra cost for $\mathrm{MC}(\mathsf{RB}{\pm}\mathsf{ATL}^+(r))$ is most likely the best we can hope for.

4.1 Carefully constructing AVASS

In this section, we assume that $Ag = [1,k]$ for some $k \geq 1$ and we explain how to handle the verification of the satisfaction of $(M,s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$, for $A \subseteq Ag$ and $\vec{b} \in \mathbb{N}^r$. To fix notations, let $M = \langle Ag, r, S, Act, act, wf, \delta, L \rangle$. As in Section 3, without loss of generality, we assume that Ψ is a Boolean formula in NNF built over atomic formulae of the form $\mathsf{X}\,p$ or $p\,\mathsf{U}\,q$ where p, q are atoms. To determine the satisfaction of $(M,s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$, we construct an AVASS $\mathcal{A}^\star = (Q^\star, r, R_1^\star, R_2^\star)$, a colouring $\mathtt{col}^\star : Q^\star \to [0,1]$ (defining a co-Büchi condition), and $q^\star \in Q^\star$ such that

1. $(M,s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$ iff there is a proof with root labelled by $(q^\star, \vec{b})$ such that all the maximal branches are infinite and the maximal colour that appears infinitely often is 0.

2. $|Q^\star|$ is (only) exponential in $|M| + |\Psi|$ and $||R_1^\star|| \leq k \times \max(||wf(s,a,\mathtt{a})|| : s \in S, a \in Ag, \mathtt{a} \in Act)$.

In Section 2.2, we show how to define an AVASS $\mathcal{A}_{M,A,s^\star} = (Q, r, R_1, R_2)$ from M, A and $s^\star$. The AVASS $\mathcal{A}^\star$ is made of copies of $\mathcal{A}_{M,A,s^\star}$ so that the general behavior of $\mathcal{A}_{M,A,s^\star}$ is preserved, but we decorate the locations with a finite memory taking care of the satisfaction of the path formula Ψ or those generated from it.

Let us provide a few more definitions. W.l.o.g., we assume that Ψ is a positive Boolean combination of atomic path formulae of one of the following forms: $\mathsf{G}\,\ell$, $\ell\,\mathsf{U}\,\ell'$, $\ell\,\mathsf{U}(\ell'_1 \wedge \ell'_2)$, $\mathsf{X}\,\ell$ where the ℓ's are literals. Notice that the transformation to obtain this shape can be performed in linear time.

Let s be in S and Φ be a Boolean combination of atomic path formulae of the above form. We write $s(\Phi)$ to denote the path formula obtained from Φ according to clauses (1)–(4) below. Intuitively, when Φ is a path formula to be satisfied on the extension of a current finite computation and s is visited next, the path formula $s(\Phi)$ is obtained from Φ by replacing the atomic path formulae in Φ that are definitely true or false by $\top$ or $\bot$, respectively.

1. If $\mathsf{G}\,\ell$ occurs in Φ and $(M,s) \not\models \ell$, then replace every occurrence of $\mathsf{G}\,\ell$ by $\bot$.
2. If $\ell\,\mathsf{U}\,\ell'$ occurs in Φ and $(M,s) \models \ell'$ (resp. $(M,s) \not\models \ell' \vee \ell$) , then replace every occurrence of $\ell\,\mathsf{U}\,\ell'$ by $\top$ (resp. by $\bot$).
3. The clauses for handling $\ell\,\mathsf{U}(\ell'_1 \wedge \ell'_2)$ are similar.
4. If $\mathsf{X}\,\ell$ occurs in Φ and $(M,s) \models \ell$ (resp. $(M,s) \not\models \ell$), then replace every occurrence of $\mathsf{X}\,\ell$ by $\top$ (resp. by $\bot$).

The final value for $s(\Phi)$ is obtained by using simplification rules to eliminate $\top$ or $\bot$, if possible. Hence, $s(\Phi)$ may take the values $\bot$ or $\top$ (not strictly speaking path formulae, but we assume so below). We write $S(\Psi)$ to denote the set of path formulae obtained from Ψ by successive applications of $s(\cdot)$ for some $s \in S$. Here are its main properties.

Lemma 3 *Let Ψ be a path formula as above, M be an RB-CGS and λ be one of its computations.*

(correctness) $M, \lambda \models \Psi$ *implies* $M, \lambda_{\geq 1} \models s(\Psi)$ *with* $\lambda[0] = s$.
(small size) $\Psi \in S(\Psi)$ *and* $|S(\Psi)|$ *is in* $\mathcal{O}(2^{|\Psi|})$.
(stabilisation) *There is* $i \geq 0$ *such that for all* $j \geq i$, *we have* $\lambda[j](\cdots(\lambda[0](\Psi))\cdots) = \lambda[i](\cdots(\lambda[0](\Psi))\cdots)$.

The proof is by an easy verification. Let us return to the construction of $Q^\star$. By definition, $Q^\star$ is equal to $S(\Psi)\times Q$ so that each $\Phi \in S(\Psi)$ has its copy of $\mathcal{A}_{M,A,s^\star}$. A location (Φ, q) is intended to follow the rules of $\mathcal{A}_{M,A,s^\star}$ to satisfy the path formula Φ. In order to update Φ, when the proof visits a next location q' attached to the state $s' \in S$, the next location in $Q^\star$ becomes $(s'(\Phi), q')$. The location $q^\star$ is defined as the pair $(s^\star(\Psi), s^\star)$, where $s^\star(\Psi)$ is built from the first three clauses only for defining $s(\cdot)$ (i.e., the atomic path formulae of the form $\mathsf{X}\,\ell$ are exceptionnally ignored to start with). Let us explain now how to define the unary and fork rules when only locations in $S(\Psi) \times Q$ are involved ($\Phi \in S(\Psi)$).

- If $(s^\star, \vec{u}, (s^\star, f)) \in R_1$, then $((\Phi, s^\star), \vec{u}, (\Phi, (s^\star, f))) \in R_1^\star$ and if $((g,s), \vec{u}, (s,f)) \in R_1$, then $((\Phi,(g,s)), \vec{u}, (\Phi,(s,f))) \in R_1^\star$. The unary rules (related to the actions of the coalition A) do not update the first argument Φ as no next location is reached.
- If $((s,f),(g_1,s_1),\ldots,(g_\alpha,s_\alpha)) \in R_2$, then $((\Phi,(s,f)),(s_1(\Phi),(g_1,s_1)),\ldots,(s_\alpha(\Phi),(g_\alpha,s_\alpha))) \in R_2^\star$.

Consequently, a proof involving locations in $S(\Psi) \times Q$ leads to a proof in $\mathcal{A}_{M,A,s^\star}$ when the first component in (Φ, q) is removed. For all (Φ, q) in $Q^\star$, we have $\mathtt{col}^\star((\Phi,q)) \stackrel{\text{def}}{=} 0$ if $\{\mathsf{G}\,\ell_1, \ldots, \mathsf{G}\,\ell_n\} \models_{\text{PROP}} \Phi$, where $\mathsf{G}\,\ell_1$, …, $\mathsf{G}\,\ell_n$ are the G-formulae in Φ and $\models_{\text{PROP}}$ is the propositional entailment (which can be checked in LOGSPACE [31]). Otherwise, $\mathtt{col}^\star((\Phi,q)) \stackrel{\text{def}}{=} 1$. In particular, by definition, $\mathtt{col}^\star((\bot,q)) = 1$ and $\mathtt{col}^\star((\top,q)) = 0$. The rationale for the definition of $\mathtt{col}^\star$ is simply that an infinite branch of a derivation in $\mathcal{A}^\star$, at some point Φ stabilises and all the states in S attached to the locations on that branch satisfies all the G-formulae in Φ (otherwise some of them would be replaced by $\bot$). The colour zero is attached only to locations in $Q^\star$ for which the satisfaction of all the G-formulae entails the satisfaction of the terminal path formula Φ. Correctness of the construction is stated below.

Lemma 4 *Let $\vec{b} \in \mathbb{N}^r$. Then, $(M, s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$ iff there is a proof whose root is labelled by $(q^\star, \vec{b})$ and all the maximal branches are infinite and the maximal colour that appears infinitely often is 0.*

The tedious proof combines advantageously the properties of $\mathcal{A}_{M,A,s^\star}$ [2] and the ones for $\mathcal{A}^\star$ related to the satisfaction of path formulae. As $|Q|$ is in $\mathcal{O}(|M|^2)$, we have $|Q^\star|$ is in $\mathcal{O}(2^{|\Psi|} \times |M|^2)$. The value $||R_1^\star||$ is bounded by $k \times \max(||wf(s,a,\mathtt{a})|| : s \in S, a \in Ag, \mathtt{a} \in Act)$, which is bounded by $|M|^2$ (direct consequence of the construction of $\mathcal{A}_{M,A,s^\star}$). By using the expression $(|Q| \times ||R_1||)^{2^{\mathcal{O}(r \times log(r+\mathtt{p}))}} + \mathcal{O}(r \times log||\vec{b}||)$ for the parity game problem for AVASS (see Section 2.2), when r is fixed and $\mathtt{p} = 2$, $(M, s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$ can be checked in exponential time in $|M| + |\langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi|$.

4.2 The model-checking algorithm

We immediately state the main result of the section.

Theorem 3 *For all $r \geq 1$,* MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(r)$) *is in* EXPTIME.

The proof of Theorem 3 uses a labelling algorithm, very similarly to the one in the proof of Theorem 2. However, to decide whether $(M, s^\star) \models \langle\!\langle A^{\vec{b}} \rangle\!\rangle \Psi$, for $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$, we first perform a reduction of the dimension by identifying the component in $\vec{b}$ equal to ω. Then, we take advantage of Lemma 4 in order to check an instance of the corresponding parity game problem for AVASS, which can be done in exponential time. As the number of such requests is only polynomial in the size of the input formula, the whole algorithm runs in exponential time. Based on [18, 2], we get the following corollary.

Corollary 1 *Let $r \geq 4$ and $|Ag| \geq 2$.*
Then, MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag, r)$) *is* EXPTIME*-complete*

The EXPTIME lower bound follows from [2, Corollary 1]. Specifically, in the proof of [2, Theorem 3] (by reduction from the control state reachability for AVASS [18]) RB-CGS can be restricted to two agents. The EXPTIME upper bound is from Theorem 3. In the case of a single resource, we obtain the following bounds.

Theorem 4 MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(1)$) *is* PSPACE*-hard and in* EXPTIME.

PSPACE-hardness follows from the PSPACE-hardness of MC(ATL^+) [15]. Though establishing an EXPTIME upper bound for MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(1)$) reveals a substantial improvement in complexity compared to the 2EXPTIME-completeness of

MC($\mathsf{RB}{\pm}\mathsf{ATL}^*$) [2], the exact complexity remains open and seems quite challenging. It is still unclear to us whether the developments in [15, Section 3.2] or in [24] about MC(ATL^+) could be adapted to obtain an optimal upper bound.

5 Conclusion

We proved that the model checking problem for $\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1)$ is in Δ_2^{P} (Theorem 2) by essentially introducing the generalised reachability problem for 1-VASS, and showing it to be in PTIME (Theorem 1). Hence, MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(\{1\},1)$) is no more complex than MC(CTL^+), despite the greater expressive power due to the presence of one resource. For $r \geq 1$, MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(r)$) is proved to be in EXPTIME, with an identical upper bound for its subproblem MC($\mathsf{RB}{\pm}\mathsf{ATL}(r)$). To show this, we presented a reduction to the parity game problem for AVASS that is optimal complexity-wise, so that we avoid the doubly-exponential blow-up observed when defining the reduction from MC($\mathsf{RB}{\pm}\mathsf{ATL}^*$) in [2]. When $r \geq 4$, we obtain EXPTIME-completeness (Corollary 1).

Though we managed to lower the complexity of MC($\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag,1)$) by one exponential (EXPTIME for $|Ag| \geq 2$), it is an open problem whether there is a procedure running in polynomial space (notice that MC(ATL^+) is known to be PSPACE-hard [15]). To this end, refined analyses from [16, 32, 17] might help. Besides, we know that CTL and CTL^+ are equally expressive [22], and this result extends to ATL and ATL^+ [15]. In both cases, the characterisation of the complexity for the respective model-checking problem differs. It is an open problem whether $\mathsf{RB}{\pm}\mathsf{ATL}(Ag,r)$ and $\mathsf{RB}{\pm}\mathsf{ATL}^+(Ag,r)$, for $|Ag| \geq 2$ and $r \geq 1$, are equally expressive.

Acknowledgments. We would like to thank the anonymous referees for their suggestions and comments that helped us to improve the quality of the paper. Besides, F. Belardinelli acknowledges the support of the ANR JCJC Project SVeDaS (ANR-16-CE40-0021).

REFERENCES

[1] P.A. Abdulla, R. Mayr, A. Sangnier, and J. Sproston, 'Solving Parity Games on Integer Vectors', in *CONCUR'13*, volume 8052 of *LNCS*, pp. 106–120. Springer, (2013).
[2] N. Alechina, N. Bulling, S. Demri, and B. Logan, 'On the complexity of resource-bounded logics', *TCS*, **750**, 69–100, (2018).
[3] 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*, pp. 1494–1501. AAAI Press, (2015).
[4] 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).
[5] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi, 'Decidable model-checking for a resource logic with production of resources', in *ECAI'14*, pp. 9–14, (2014).
[6] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi, 'Symbolic model checking for one-resource RB+-ATL', in *IJCAI'15*, pp. 1069–1075. AAAI Press, (2015).
[7] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi, 'Model-checking for resource-bounded ATL with production and consumption of resources', *JCSS*, **88**, 126–144, (2017).
[8] N. Alechina, B. Logan, H.N. Nguyen, and A. Rakib, 'A logic for coalitions with bounded resources', in *IJCAI'09*, pp. 659–664, (2009).
[9] N. Alechina, B. Logan, H.N. Nguyen, and A. Rakib, 'Resource-bounded alternating-time temporal logic', in *AAMAS'10*, pp. 481–488. IFAAMAS, (2010).
[10] R. Alur, L. de Alfaro, T. Henzinger, S. Krishnan, F. Mang, S. Qadeer, S. Rajamani, and S. Tasiran, 'MOCHA user manual', Technical report, University of California at Berkeley, (2000).
[11] R. Alur, T. A. Henzinger, and O. Kupferman, 'Alternating-time temporal logic', *Journal of the ACM*, **49**(5), 672–713, (2002).
[12] F. Belardinelli and S. Demri, 'Resource-bounded ATL: the quest for tractable fragments', in *AAMAS'19*, pp. 206–214, (2019).
[13] N. Bulling and B. Farwer, 'On the (Un-)Decidability of Model-Checking Resource-Bounded Agents', in *ECAI'10*, pp. 567–572, (2010).
[14] N. Bulling and V. Goranko, 'How to be both rich and happy: Combining quantitative and qualitative strategic reasoning about multi-player games (extended abstract)', in *Proceedings 1st International Workshop on Strategic Reasoning (SR'13)*, volume 112 of *EPTCS*, pp. 33–41, (2013).
[15] N. Bulling and W. Jamroga, 'Verifying agents with memory is harder than it seemed', *AI Communications*, **23**(4), 389–403, (2010).
[16] K. Chatterjee and L. Doyen, 'Energy parity games', *TCS*, **458**, 49–60, (2012).
[17] Th. Colcombet, M. Jurdziński, R. Lazić, and S. Schmitz, 'Perfect half space games', in *LiCS'17*, pp. 1–11. IEEE Press, (2017).
[18] J.B. Courtois and S. Schmitz, 'Alternating vector addition systems with states', in *MFCS'14*, volume 8634 of *LNCS*, pp. 220–231. Springer, (2014).
[19] S. Demri, V. Goranko, and M. Lange, *Temporal Logics in Computer Science*, Cambridge University Press, 2016.
[20] S. Demri, M. Jurdziński, O. Lachish, and R. Lazić, 'The covering and boundedness problems for branching vector addition systems', *JCSS*, **79**(1), 23–38, (2013).
[21] S. Demri and Ph. Schnoebelen, 'The complexity of propositional linear temporal logics in simple cases', *I&C*, **174**(1), 84–103, (2002).
[22] A. Emerson and J. Halpern, 'Decision procedures and expressiveness in the temporal logic of branching time', *JCSS*, **30**, 1–24, (1985).
[23] S. Göller, C. Haase, R. Lazić, and P. Totzke, 'A polynomial-time algorithm for reachability in branching VASS in dimension one', in *ICALP'16*, volume 55 of *LIPIcs*, pp. 105:1–105:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, (2016).
[24] V. Goranko, A. Kuusisto, and R. Rönnholm, 'Game-theoretic semantics for ATL^+ with applications to model checking', in *AAMAS'17*, pp. 1277–1285. ACM, (2017).
[25] P. Habermehl, 'On the complexity of the linear-time mu-calculus for Petri nets', in *ICATPN'97*, volume 1248 of *LNCS*, pp. 102–116. Springer, (1997).
[26] M. Jurdziński, R. Lazić, and S. Schmitz, 'Fixed-dimensional energy games are in pseudo-polynomial time', in *ICALP'15*, volume 9135 of *LNCS*, pp. 260–272. Springer, (2015).
[27] R.M. Karp and R.E. Miller, 'Parallel program schemata', *JCSS*, **3**(2), 147–195, (1969).
[28] F. Laroussinie, N. Markey, and Ph. Schnoebelen, 'Model checking CTL^+ and FCTL is hard', in *FoSSaCS'01*, volume 2030 of *LNCS*, pp. 318–331. Springer, (2001).
[29] R.J. Lipton, 'The reachability problem requires exponential space', Technical Report 62, Department of Computer Science, Yale University, (1976).
[30] 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.
[31] N. Lynch, 'Log Space recognition and translation of parenthesis languages', *Journal of the ACM*, **24**(4), 583–590, (1977).
[32] V. Malvone, A. Murano, and L. Sorrentino, 'Concurrent multi-player parity games', in *AAMAS'16*, pp. 689–697. ACM, (2016).
[33] D. Della Monica and A. Murano, 'Parity-energy ATL for qualitative and quantitative reasoning in MAS', in *AAMAS'18*, pp. 1441–1449, (2018).
[34] D. Della Monica, M. Napoli, and M. Parente, 'On a logic for coalitional games with priced-resource agents', *ENTCS*, **278**, 215–228, (2011).
[35] S. Schewe, 'Tighter bounds for the determinisation of Büchi automata', in *FoSSaCS'09*, volume 5504 of *LNCS*, pp. 167–181. Springer, (2009).
[36] A. Sistla and E. Clarke, 'The complexity of propositional linear temporal logic', *Journal of the ACM*, **32**(3), 733–749, (1985).
[37] M. Vardi and P. Wolper, 'Reasoning about infinite computations', *I&C*, **115**, 1–37, (1994).
[38] K.N. Verma and J. Goubault-Larrecq, 'Karp-Miller Trees for a Branching Extension of VASS', *Discrete Mathematics and Theoretical Computer Science*, **7**, 217–230, (2005).
[39] Th. Wilke, 'CTL^+ is exponentially more succinct than CTL', in *FST&TCS'99*, volume 1999 of *LNCS*, pp. 110–121. Springer, (1999).