Alechina, Natasha and Bulling, Nils and Demri, Stephane and Logan, Brian (2016) On the complexity of resource-bounded logics. In: 10th International Workshop on Reachability Problems (RP 2016), 19-12 September 2016, Aalborg, Denmark.

On the Complexity of Resource-Bounded Logics

N. Alechina[1], N. Bulling[2], S. Demri[3] and B. Logan[1]

[1] University of Nottingham [2] TU Delft [3] LSV, CNRS, ENS Cachan

Abstract. We revisit decidability results for resource-bounded logics and use decision problems for vector addition systems with states (VASS) to characterise the complexity of (decidable) model-checking problems. We show that the model-checking problem for the logic RB±ATL is 2EXPTIME-complete by using recent results on alternating VASS. In addition, we establish that the model-checking problem for RBTL is decidable and has the same complexity as for RBTL* (the extension of RBTL with arbitrary path formulae), namely EXPSPACE-complete, proving a new decidability result as a by-product of the approach. Finally, we establish that the model-checking problem for RB±ATL* is decidable by a reduction to parity games, and show how to synthesise values for resource parameters.

1 Introduction

Resource-bounded logics [11,10,29,3,2,12] extend alternating-time temporal logic (ATL) [5] by adding transitions that produce and consume resources to the models. As shown in [2], the introduction of implicit counters in the models (i.e. variables interpreted by natural numbers) and the ability to quantify over strategies for a given set of agents can lead to undecidability, or decidability with a very high worst-case upper bound on the complexity of the model checking problem.

The nature of the strategy modalities means that reasoning about resources has similarities to the analysis of runs of vector addition systems with states (a.k.a. VASS) [27], and more specifically to games on VASS, see e.g. [9]. In this paper, we exploit results on VASS in order to analyse the model-checking problem for resource-bounded logics. Model-checking problems on VASS based on temporal logics and games are not always decidable, or at least quite difficult to solve, but sharp results exist. Temporal logics on VASS often lead to undecidable model-checking problems, see e.g. [17,18], and this is more common with branching-time temporal logics such as CTL [18], or when the atomic formulae can state properties about the counter values [22]. However, there are exceptions. For example, CTL model-checking on one-counter VASS is PSPACE-complete [32,19] (see also [34]). Similarly, the control-state repeated reachability problem for VASS is shown to be decidable in [23], and this is generalised to full LTL (for which the atomic formulae are exactly control states) in [21], where the model-checking problem for LTL on VASS is shown to be EXPSPACE-complete. Also in [23], a strict fragment of LTL restricted to the "infinitely often" temporal operator GF and atomic formulae stating properties on counter values is shown decidable by a reduction into the reachability problem for VASS.

As far as games for VASS are concerned, the situation is even less encouraging. Two-player games on VASS in which each player can freely update the counter values

are undecidable, even with simple winning conditions such as the reachability of a given control state [9]. However, asymmetric VASS games in which at most one player can freely update the counter values and where the winning conditions are simple, are decidable [31]. In addition, the game on asymmetric VASS with reachability of a control state (a slight variant of single-sided VASS in [1] or alternating VASS in [13]) has been shown to be 2EXPTIME-complete [13], and decidable with parity conditions [1,24]. The non-termination problem for asymmetric games is 2EXPTIME-complete (the upper bound is from [26] and the lower bound is from [13]).

In this paper, we establish formal relationships between model-checking problems for resource-bounded logics and decision problems for alternating VASS (also known as single-sided VASS). We then use these relationships to show new results for the decidability and complexity of model-checking resource-bounded logics. Ours is not the first work in this direction. There are clear similarities between resource values and counter values, and the semantics of resource-bounded logics are inherently game-based. Previous work has explored the connections with counter machines, either to obtain undecidability, or to obtain lower bounds on complexity, e.g., [11,2].

We give optimal complexity upper bounds and new decidability results, including for resource-bounded logics with enriched path formulae as those in CTL* [16]. First, we show that the model-checking problem for RB±ATL is 2EXPTIME-complete (Theorem 1 and Theorem 2), and that RB±ATL restricted to a bounded number of resources is in EXPTIME. The 2EXPTIME lower bound is obtained by a reduction from the state reachability problem for alternating VASS (AVASS) [13], whereas the upper bound is established by a reduction to the state reachability and the termination problems for AVASS (both problems are needed). These results are obtained by using formal relationships between strategies in concurrent game structures and proofs in AVASS, and the key observation is that only asymmetric VASS are needed. The formal relationships also allow us to show that the model-checking problem for RB±ATL* (a new logic naturally extending RB±ATL) is decidable by a reduction to the parity game problem for AVASS [1] (Theorem 4). To the best of our knowledge, the complexity of the parity game problem for AVASS is still open. We also show that resource parameters can be effectively computed in the parameterised version of RB±ATL* (Theorem 5), due to the fact that the Pareto frontier for any parity game on single-sided VASS is computable [1, Theorem 4]. As far as we know, this is the first time that resource values are synthesised in resource-bounded logics (see also [25]). Lastly, we show that the model-checking problems for RBTL [10] and its extension RBTL* are EXPSPACE-complete, and that RBTL restricted to a bounded number of resources is in PSPACE.

2 Alternating VASS Preliminaries

We write $\mathbb{N}$ (resp. $\mathbb{Z}$) for the set of natural numbers (resp. integers) and $[m, m']$ with $m, m' \in \mathbb{Z}$ to denote the set $\{j \in \mathbb{Z} : m \leq j \leq m'\}$. Given a dimension $r \geq 1$ and $a \in \mathbb{Z}$, we write $\boldsymbol{a} \in \mathbb{Z}^r$ to denote the vector of dimension r with all components equal to a. For each $\boldsymbol{x} \in \mathbb{Z}^r$, we write $\boldsymbol{x}(1), \ldots, \boldsymbol{x}(r)$ for the entries of $\boldsymbol{x}$. For each $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^r$, $\boldsymbol{x} \preceq \boldsymbol{y} \stackrel{\text{def}}{\Leftrightarrow}$ for every $i \in [1, r]$, we have $\boldsymbol{x}(i) \leq \boldsymbol{y}(i)$. We also write $\boldsymbol{x} \prec \boldsymbol{y}$ when $\boldsymbol{x} \preceq \boldsymbol{y}$ and $\boldsymbol{x} \neq \boldsymbol{y}$.

A binary tree $\mathfrak{T}$, which may contain nodes with one child, is a non-empty subset of $\{1,2\}^*$ such that, for all $\mathfrak{n} \in \{1,2\}^*$ and $i \in \{1,2\}$, $\mathfrak{n} \cdot i \in \mathfrak{T}$ implies $\mathfrak{n} \in \mathfrak{T}$ and, $\mathfrak{n} \cdot 2 \in \mathfrak{T}$ implies $\mathfrak{n} \cdot 1 \in \mathfrak{T}$. The nodes of $\mathfrak{T}$ are its *elements*. The root of $\mathfrak{T}$ is ε, the empty word. All notions such as parent, first child, second child, subtree and leaf, have their standard meanings. The height of $\mathfrak{T}$ is the length, i.e. the number of nodes, of the longest simple path from the root to a leaf.

An *alternating VASS* (AVASS) [13] is a tuple $\mathcal{A} = (Q, r, R_1, R_2)$ such that (1) Q is a finite set of *locations* (a.k.a. *control states*) and $r \geq 0$ is the number of resource values, (2) R_1 is a finite subset of $Q \times \mathbb{Z}^r \times Q$ (*unary rules*) and (3) R_2 is a finite subset of Q^3 (*fork rules*). A *derivation skeleton* of $\mathcal{A}$ is a labelling $\mathcal{D} : \mathfrak{T} \to (R_1 \cup R_2 \cup \{\bot\})$ such that: (1) $\mathfrak{T}$ is a (possibly infinite) binary tree (subset of $\{1,2\}^*$ with standard conditions), (2) if $\mathfrak{n}$ has one child in $\mathfrak{T}$, then $\mathcal{D}(\mathfrak{n}) \in R_1$, (3) if $\mathfrak{n}$ has two children in $\mathfrak{T}$, then $\mathcal{D}(\mathfrak{n}) \in R_2$ and (4) if $\mathfrak{n}$ is a leaf in $\mathfrak{T}$, then $\mathcal{D}(\mathfrak{n}) = \bot$. A *derivation* of $\mathcal{A}$ based on $\mathcal{D}$ is a labelling $\hat{\mathcal{D}} : \mathfrak{T} \to Q \times \mathbb{Z}^r$ such that: (1) if $\mathfrak{n}$ has one child $\mathfrak{n}'$ in $\mathfrak{T}$, $\mathcal{D}(\mathfrak{n}) = (q, \boldsymbol{u}, q')$ and $\hat{\mathcal{D}}(\mathfrak{n}) = (q, \boldsymbol{v})$, then $\hat{\mathcal{D}}(\mathfrak{n}') = (q', \boldsymbol{v} + \boldsymbol{u})$ and (2) if $\mathfrak{n}$ has two children $\mathfrak{n}'$ and $\mathfrak{n}''$ in $\mathfrak{T}$, $\mathcal{D}(\mathfrak{n}) = (q, q_1, q_2)$ and $\hat{\mathcal{D}}(\mathfrak{n}) = (q, \boldsymbol{v})$, then $\hat{\mathcal{D}}(\mathfrak{n}') = (q_1, \boldsymbol{v})$ and $\hat{\mathcal{D}}(\mathfrak{n}'') = (q_2, \boldsymbol{v})$. So, fork rules do not update the resources and whence, there is an asymmetry between unary rules and fork rules (this makes a difference with branching VASS, see e.g. [33,15]). This is a useful feature when dealing with the proponent restriction condition in RB±ATL. A derivation $\hat{\mathcal{D}}$ is *admissible* whenever $\hat{\mathcal{D}} : \mathfrak{T} \to Q \times \mathbb{N}^r$, i.e. only natural numbers occur in it. An admissible derivation is also called a *proof*.

The *state reachability problem* for AVASS is as follows: given an AVASS $\mathcal{A}$ and control states q_0 and q_f, is there a finite proof of AVASS whose root is equal to $(q_0, \mathbf{0})$ and each leaf belongs to $\{q_f\} \times \mathbb{N}^r$? When $\mathcal{A}$ has no fork rules, $\mathcal{A}$ is essentially a VASS [27] and the above problem is an instance of the coverability problem, which is known to be EXPSPACE-complete [28,30] (see also [7,14]). The *non-termination problem* for AVASS is as follows: given an AVASS $\mathcal{A}$ and a control state q_0, is there a proof whose root is equal to $(q_0, \mathbf{0})$ and all the maximal branches are infinite?

Proposition 1. [13,26] *The state reachability and non-termination problems for AVASS are* 2EXPTIME*-complete.*

Decidability of these problems were first established in [31] by using monotonicity of the games. The 2EXPTIME upper bound is preserved if we assume that the root is labelled by $(q_0, \boldsymbol{b})$ with $\boldsymbol{b} \in \mathbb{N}^r$ encoded with a binary representation (see Lemma 1).

In the sequel, we shall also admit fork rules of any arity $\alpha \geq 1$, and therefore, in such slightly extended AVASS the set of fork rules R_2 is a finite subset of $\bigcup_{\beta \geq 2} Q^\beta$.

Lemma 1. *The following extension of the state reachability and non-termination problems for AVASS remains in* 2EXPTIME*:*

- *Fork rules can be α-ary for any $\alpha \geq 1$ (but there are only a finite amount of them).*
- *Reachability is related to a subset $Q_f \subseteq Q$ (instead of a singleton set).*
- *The initial configuration is $(q_0, \boldsymbol{b})$ with $\boldsymbol{b} \in \mathbb{N}^r$ instead of the fixed tuple $\mathbf{0}$.*
- *The value ω is allowed in $\boldsymbol{b}$ in the initial configuration $(q_0, \boldsymbol{b})$, where for all $n \in \mathbb{Z}$, we have $\omega = n + \omega = \omega + n$.*

The rather standard proof consists in using Proposition 1 by simulating a non-binary fork by a linear-size gadget made of unary and binary fork rules and by adding binary fork rules from states in Q_f to a new single final state (alternatively, one could add unary rules with effect $\mathbf{0}$). The third item in Lemma 1 can be handled by adding a new unary rule with effect $\boldsymbol{b}$ whereas the fourth one amounts to ignore the components with the root value ω.

The notions of derivation skeleton, derivation and proof are also extended to general trees $\mathfrak{T} \subseteq (\mathbb{N} \setminus \{0\})^*$. The set of finite words $\mathfrak{T} \subseteq (\mathbb{N} \setminus \{0\})^*$ is a (not necessarily binary) *tree* iff for all $\mathfrak{n} \in (\mathbb{N} \setminus \{0\})^*$ and $i \in (\mathbb{N} \setminus \{0\})$, $\mathfrak{n} \cdot i \in \mathfrak{T}$ implies $\mathfrak{n} \in \mathfrak{T}$ and, $\mathfrak{n} \cdot i \in \mathfrak{T}$ and $i > 1$ imply $\mathfrak{n} \cdot (i-1) \in \mathfrak{T}$. Such AVASS correspond to a *single-sided VASS* [6,1].

In what follows, by a VASS we mean an alternating VASS without any fork rule and write it as $\mathcal{V} = (Q, r, R)$ where R is a finite set of unary rules. Given a VASS $\mathcal{V}$, its transition system $\mathfrak{TS}(\mathcal{V}) \stackrel{\text{def}}{=} (\mathfrak{W}, \rightarrow, L)$ is such that: (1) $\mathfrak{W} \stackrel{\text{def}}{=} Q \times \mathbb{N}^r$, (2) L is a truth assigment with elements of Q also understood as propositional variables and $L(q) \stackrel{\text{def}}{=} \{q\} \times \mathbb{N}^r$ and (3) $\rightarrow$ is a binary relation on $\mathfrak{W}$ such that $(q, \boldsymbol{v}) \rightarrow (q', \boldsymbol{v}')$ iff there is a unary rule $(q, \boldsymbol{u}, q')$ in R such that $\boldsymbol{v}' = \boldsymbol{v} + \boldsymbol{u}$ where '+' is the component-wise addition on $\mathbb{N}^r$. As usual, we also write $\stackrel{*}{\rightarrow}$ to denote the reflexive and transitive closure of $\rightarrow$. Since $\mathfrak{TS}(\mathcal{V})$ is a Kripke-style structure, it can be used to interpret modal or temporal formulae (e.g., LTL or CTL formulae) where atomic formulae refer to control states. Since alternating-time temporal logics such as ATL or ATL* are strict extensions of CTL or CTL* respectively, complexity hardness results for temporal logics can be lifted to such logics. A known result which will be useful in the sequel is that the model-checking problem for LTL on VASS is EXPSPACE-complete (the atomic formulae/propositions are control states) and it is PSPACE-complete for a fixed number of resources [21].

Below, we consider AVASS with a finite set of fork rules included in $\bigcup_{\beta \geq 2} Q^\beta$, and where the proofs are trees with nodes labelled by elements in $Q \times (\mathbb{N} \cup \{\omega\})^r$. Given an AVASS $\mathcal{A} = (Q, r, R_1, R_2)$, a *colouring* `col` is a map $Q \rightarrow [0, p]$ for some $p \geq 0$. The *parity game problem* for AVASS is as follows: given an AVASS $\mathcal{A}$, a control state q_0, $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$ and $\texttt{col} : Q \rightarrow [0, p]$, is there a proof the root of which is equal to $(q_0, \boldsymbol{b})$, all the maximal branches are infinite and the maximal colour that appears infinitely often along each branch is even (the colour of each node is induced by `col`)?

Proposition 2. [1, Corollary 2] *The parity game problem for AVASS is decidable.*

To be precise, [1, Corollary 2] states the result for single-sided VASS that can be viewed as AVASS such that the set Q of control states is partitioned into $Q = Q_1 \uplus Q_2$, unary rules start by states in Q_1, fork rules start by states in Q_2 and there is at most one fork rule starting in the same control state (necessarily, belonging to Q_2). The problem for AVASS can be reduced to that for single-sided VASS. It is not difficult to show that the state reachability and non-termination problems for AVASS can be understood as subproblems of the parity game problem and therefore their decidability also follows from [1]. However, the situation is different in the case of complexity. While the exact complexity of the parity game problem is unknown, the state reachability and non-termination problems for AVASS are shown to be 2EXPTIME-hard in [13], the state reachability problem is shown to be in 2EXPTIME in [13] and the non-termination

problem is proved to be in 2EXPTIME in [26]. It has been shown recently that the parity game problem is in TOWER [24].

This decidability result has been strenghtened in [1] in the following way. Given $\mathcal{A}$, q_0 and $\mathtt{col} : Q \to [0, p]$, the set of tuples $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$ for which there is a positive solution to the parity game problem is upward closed and computable. This means that it can be represented effectively by a Boolean combination of atomic constraints of the form either $\mathtt{x}_i \geq k$ where $i \in [1, r]$ and $k \in \mathbb{N}$ or $\mathtt{x}_i = \omega$. Indeed, since the set is upward closed, by Dickson's Lemma, it has a finite set of minimal elements (with respect to the well-quasi-ordering $\preceq$ slightly extended to accomodate the addition of the value ω) that allows one to define easily the symbolic representation in terms of atomic constraints of the form $\mathtt{x} \geq k$. The *Pareto frontier* of $\mathcal{A}$, q_0 and $\mathtt{col} : Q \to [0, p]$ is defined as the set of minimal elements in $(\mathbb{N} \cup \{\omega\})^r$ for which there is a positive solution to the parity game problem.

Proposition 3. [1, Theorem 4] *The Pareto frontier for any parity game on single-sided VASS is computable.*

3 The Logic RB±ATL and Variants

We consider the logics RB±ATL and RB±ATL*. The logic RB±ATL was introduced in [3,4], and extends ATL [5] with resources. RB±ATL* extends RB±ATL to allow path formulae to be any LTL-like formula.

Let PROP be a countably infinite set of atomic propositions. The models for the logics RB±ATL and RB±ATL* are the structures introduced in Definition 1 below. These are concurrent game structures from [5], but enriched with a cost function that specifies how resources are produced or consumed. At some abstract level, a structure is equipped with r counters and the transitions can perform increments and decrements.

Definition 1. *A* resource-bounded concurrent game structure $\mathfrak{M}$ *is a tuple of the form* $(Agt, S, Act, r, \mathtt{act}, \mathtt{cost}, \delta, L)$ *such that:*

- Agt *is a non-empty finite set of* agents *(by default* $Agt = [1, k]$ *for some* $k \geq 1$*).*
- S *is a set of states and* $r \geq 1$ *is the number of resources.*
- Act *is a non-empty set of actions with a distinguished action* $\mathtt{idle}$.
- $\mathtt{act} : Agt \times S \to \mathcal{P}(Act)$ *is the* action manager function *such that for all* a *and* s *we have* $\mathtt{idle} \in \mathtt{act}(a, s)$.
- $\mathtt{cost} : S \times Agt \times Act \to \mathbb{Z}^r$ *is the (partial) cost function so that* $\mathtt{cost}(s, a, \mathtt{a})$ *is defined exactly when* $\mathtt{a} \in \mathtt{act}(a, s)$. *Moreover,* $\mathtt{cost}(s, a, \mathtt{idle}) = \mathbf{0}$.
- $\delta : S \times (Agt \to Act) \to S$ *is the (partial) transition function such that* δ *is defined on* $(s, \mathtt{f})$ *whenever for all agents* $a \in Agt$*, we have* $\mathtt{f}(a) \in \mathtt{act}(a, s)$.
- $L : \text{PROP} \to \mathcal{P}(S)$ *is a truth assignment (the definition can be adapted when finite subsets of* PROP *are involved).*

The map δ is also viewed as a deterministic relation with transitions of the form $s \xrightarrow{(\mathtt{a}_1, \ldots, \mathtt{a}_k)} s'$ where $\delta(s, \mathtt{f}) = s'$ and for all $i \in [1, k] = Agt$, we have $\mathtt{f}(i) = \mathtt{a}_i$. We say that $\mathfrak{M}$ is finite whenever S and Act are finite sets and L is restricted to a finite subset of PROP.

For instance, the idle action is considered in [3,4], where motivations for considering such a distinguished action are given. Given a *coalition* $A \subseteq Agt$ and a state s, a *joint action* by A in s is a map $\mathfrak{f} : A \to Act$ such that, for all agents $a \in A$, we have $\mathfrak{f}(a) \in \mathtt{act}(a, s)$. The set of joint actions by A in s is denoted $D_A(s)$. Given a state s, the set of joint actions by Agt in s is denoted $D(s)$ (instead of $D_{Agt}(s)$) and the map δ is defined only for such joint actions. We write $\mathfrak{f} \sqsubseteq \mathfrak{g}$ whenever $\mathfrak{g}$ is a conservative extension of $\mathfrak{f}$, i.e. $\mathrm{dom}(\mathfrak{f}) \subseteq \mathrm{dom}(\mathfrak{g})$ and, $\mathfrak{f}$ and $\mathfrak{g}$ agree on $\mathrm{dom}(\mathfrak{f})$.

Given a joint action $\mathfrak{f} \in D_A(s)$, we write $\mathtt{out}(s, \mathfrak{f})$ to denote $\{s' \in S \mid \text{there is } \mathfrak{g} \in D(s) \text{ such that } \mathfrak{f} \sqsubseteq \mathfrak{g} \text{ and } s' = \delta(s, \mathfrak{g})\}$. For instance, $\mathtt{out}(s, \mathfrak{f})$ is a singleton set when $\mathfrak{f} \in D(s)$ since δ is a map and not a relation. Given a joint action $\mathfrak{f} \in D_A(s)$ and a state s, the *cost* of any transition fired from s following $\mathfrak{f}$ (restricted to A by definition) is as follows: $\mathtt{cost}_A(s, \mathfrak{f}) \stackrel{\text{def}}{=} \sum_{a \in A} \mathtt{cost}(s, a, \mathfrak{f}(a))$. In a sense, the value $\mathtt{cost}_A(s, \mathfrak{f})$ does not depend on the costs related to the agents in $(Agt \setminus A)$, or equivalently, the cost related to the agents in $(Agt \setminus A)$ is reduced to zero.

A *computation* λ is a finite sequence or an ω-sequence of the form $s_0 \xrightarrow{\mathfrak{f}_0} s_1 \xrightarrow{\mathfrak{f}_1} s_2 \ldots$ such that for all $i < |\lambda| - 1$, we have $s_{i+1} \in \delta(s_i, \mathfrak{f}_i)$. Here, $|\lambda|$ denotes the length of λ, each s_i is a state and each $\mathfrak{f}_i$ belongs to $D(s_i)$. For instance $|s_0 \xrightarrow{\mathfrak{f}_0} s_1 \cdots \xrightarrow{\mathfrak{f}_{n-1}} s_n| = n + 1$ and $|s_0 \xrightarrow{\mathfrak{f}_0} s_1 \cdots \xrightarrow{\mathfrak{f}_{n-1}} \cdots| = \omega$ for any infinite computation. So, in full generality, in a computation, a transition between two successive states is labelled by a joint action: this is not strictly needed for the forthcoming developments but it provides a more general notion that might be used in other contexts (for instance, if the winning condition of forthcoming strategies depends on the actions of all the agents and not only on those for the agents in A or on the visited states). A *strategy* F_A for the coalition A is a map from the set of finite computations to the set of joint actions by A such that $F_A(s_0 \xrightarrow{\mathfrak{f}_0} s_1 \cdots \xrightarrow{\mathfrak{f}_{n-1}} s_n) \in D_A(s_n)$. A computation $\lambda = s_0 \xrightarrow{\mathfrak{f}_0} s_1 \xrightarrow{\mathfrak{f}_1} s_2 \cdots$ *respects* the strategy F_A iff for all $i < |\lambda|$, we have, $s_{i+1} \in \mathtt{out}(s_i, F_A(s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \xrightarrow{\mathfrak{f}_{i-1}} s_i))$. A computation λ that respects F_A is *maximal* whenever it cannot be extended further while respecting the strategy. Note that maximal computations respecting F_A are infinite. The set of all maximal computations that respect the strategy F_A and that start at the state s is denoted by $\mathtt{out}(s, F_A)$. So far, no resource value has been involved in computations. Below, we shall quantify over maximal computations that respect a strategy and therefore for defining a strategy we can restrict ourselves to finite computations that respect it so far.

Given a bound $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$, a computation $\lambda = s_0 \xrightarrow{\mathfrak{f}_0} s_1 \xrightarrow{\mathfrak{f}_1} s_2 \ldots$ in $\mathtt{out}(s, F_A)$ is $\boldsymbol{b}$*-consistent* iff for all $i < |\lambda|$, we have $\mathbf{0} \preceq (\sum_{j=0}^{i-1} \mathtt{cost}_A(s_j, F_A(s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \xrightarrow{\mathfrak{f}_{j-1}} s_j)) + \boldsymbol{b})$. Whenever $\boldsymbol{b}(i) = \omega$, this can be viewed as a means to disregard what happens on the ith resource (assuming that $n + \omega = \omega$ for any $n \in \mathbb{Z}$). Indeed, $\boldsymbol{b}(i) = \omega$ amounts to guarantee from the beginning of the computation that there is an infinite supply of resources on the ith component. Note also that the above condition is slightly different from the one in [4] but strictly equivalent. We have decided to adopt that notation in order to show more easily the relationships with VASS decision problems. So, for a computation $\lambda = s_0 \xrightarrow{\mathfrak{f}_0} s_1 \xrightarrow{\mathfrak{f}_1} s_2 \ldots$ and a coalition A, there is an underlying sequence $\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots$ of resource values so that $\boldsymbol{v}_0 \stackrel{\text{def}}{=} \boldsymbol{b}$ and for all $i < |\lambda| - 1$, we have $\boldsymbol{v}_{i+1} \stackrel{\text{def}}{=}$

$\boldsymbol{v}_i + \mathtt{cost}_A(s_i, F_A(s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \xrightarrow{\mathfrak{f}_{i-1}} s_i))$. The values of the sequence only depend on the agents in A, which is often called the *proponent restriction condition*.

The set of all the $\boldsymbol{b}$-consistent (infinite) computations is denoted by $\mathtt{out}(s, F_A, \boldsymbol{b})$. A *$\boldsymbol{b}$-strategy F_A with respect to s* is a strategy such that $\mathtt{out}(s, F_A) = \mathtt{out}(s, F_A, \boldsymbol{b})$. This definition also slightly differs from the one in [4] that is not relative to a given state and therefore in [4] the equality should hold for all the states.

So far, we have provided the main definitions about resource-bounded concurrent game structures and strategies. Let us present now the logic RB±ATL. Given a set of agents $Agt = \{a_1, \ldots, a_k\}$ and $r \geq 1$, we write RB±ATL(Agt, r) to denote the resource-bounded logic with k agents and r resources whose models are resource-bounded concurrent game structures with the same parameters. Formulae of RB±ATL(Agt, r) are defined according to the grammar:

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle A^{\boldsymbol{b}} \rangle\rangle \bigcirc \phi \mid \langle\langle A^{\boldsymbol{b}} \rangle\rangle \Box\phi \mid \langle\langle A^{\boldsymbol{b}} \rangle\rangle \phi \boldsymbol{\mathcal{U}} \phi,$$

where $p \in \text{PROP}$, $A \subseteq Agt$ and $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$. The size of a formula is computed from a DAG representation and the integers are encoded in binary. Note that forthcoming hardness results do not use the conciseness of the DAG representation (with respect to the tree representation).

The satisfaction relation $\models$ is defined inductively as follows assuming that $\mathfrak{M}$ is an RB±ATL(Agt, r) model (we omit the obvious cases for the Boolean connectives): $\mathfrak{M}, s \models p \overset{\text{def}}{\Leftrightarrow} s \in L(p)$ and,

- $\mathfrak{M}, s \models \langle\langle A^{\boldsymbol{b}} \rangle\rangle \bigcirc \phi \overset{\text{def}}{\Leftrightarrow}$ there is a $\boldsymbol{b}$-strategy F_A w.r.t. s such that for all $s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \in \mathtt{out}(s, F_A)$, we have $\mathfrak{M}, s_1 \models \phi$
- $\mathfrak{M}, s \models \langle\langle A^{\boldsymbol{b}} \rangle\rangle \Box\phi \overset{\text{def}}{\Leftrightarrow}$ there is a $\boldsymbol{b}$-strategy F_A w.r.t. s such that for all $\lambda = s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \in \mathtt{out}(s, F_A)$, for all $i < |\lambda|$, we have $\mathfrak{M}, s_i \models \phi$
- $\mathfrak{M}, s \models \langle\langle A^{\boldsymbol{b}} \rangle\rangle \phi_1 \boldsymbol{\mathcal{U}} \phi_2 \overset{\text{def}}{\Leftrightarrow}$ there is a $\boldsymbol{b}$-strategy F_A w.r.t. s such that for all $\lambda = s_0 \xrightarrow{\mathfrak{f}_0} s_1 \ldots \in \mathtt{out}(s, F_A)$, there is some $i < |\lambda|$ such that $\mathfrak{M}, s_i \models \phi_2$ and for all $j \in [0, i-1]$, we have $\mathfrak{M}, s_j \models \phi_1$.

Since all the maximal computations are infinite, the index i involved for clauses above related to $\langle\langle A^{\boldsymbol{b}} \rangle\rangle \Box$ or $\langle\langle A^{\boldsymbol{b}} \rangle\rangle \boldsymbol{\mathcal{U}}$ can take any value in $\mathbb{N}$. The presence of the idle action allows the extension of a strategy as soon as a given condition is satisfied along the computations. For instance, $\mathfrak{M}, s \models \langle\langle A^{\boldsymbol{b}} \rangle\rangle \bigcirc \phi$ is equivalent to the existence of $\mathfrak{f} \in D_A(s)$ such that for all $\mathfrak{g} \sqsupseteq \mathfrak{f}$, we have $\mathfrak{M}, s' \models \phi$ with $\delta(s, \mathfrak{g}) = s'$ and $\boldsymbol{0} \preceq \boldsymbol{b} + \mathtt{cost}_A(s, \mathfrak{f})$. Moreover, a strategy modality $\langle\langle A^{\boldsymbol{b}} \rangle\rangle$ reduces the impact of the function $\mathtt{cost}$ in two ways. If the ith component of $\boldsymbol{b}$ is equal to ω, then there are no constraints on the ith resource along the computation. The restriction of $\mathtt{cost}$ to opponent agents in $(Agt \setminus A)$ is also reduced to $\boldsymbol{0}$ (so without any impact on consistency).

Obviously, RB±ATL(Agt, r) is a quantitative variant of ATL [5] in which resource values are computed along the computations.

The *model-checking problem for RB±ATL* is as follows: given $k, r \geq 1$ (in unary), a formula ϕ in RB±ATL$([1, k], r)$, a finite RB±ATL$([1, k], r)$ model $\mathfrak{M}$ and a state s, is $\mathfrak{M}, s \models \phi$? The encoding of k and r in unary is unessential since the size of $\mathfrak{M}$ with an explicit representation of all the transitions is over $k + r$.

Proposition 4. *[3, Theorem 1] The model-checking problem for RB±ATL is decidable.*

We also consider RB±ATL*, an extension of RB±ATL in which the path formulae can be any LTL-like formula, in particular a temporal operator may no longer be preceded by a cooperation modality. This is a new logic although its definition follows a classical schema for branching-time temporal logics. Given a set of agents $Agt = [1, k]$ and $r \geq 1$, we write RB±ATL$^*(Agt, r)$ to denote the resource-bounded logic with k agents and r resources whose models are resource-bounded concurrent game structures with the same parameters. The *parameterised version* of RB±ATL* denoted by ParRB±ATL* admits formulae as RB±ATL* except that the values $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$ are replaced by tuples of variables within $\mathtt{VAR} = \{\mathtt{x}_1, \mathtt{x}_2, \ldots\}$. Here is a typical formula in ParRB±ATL*:

$$\langle\langle\{1\}^{(\mathtt{x}_1,\mathtt{x}_2)}\rangle\rangle \top \mathcal{U} q_f \wedge \langle\langle\{2\}^{(\mathtt{x}_2,\mathtt{x}_3)}\rangle\rangle \top \mathcal{U} q'_f.$$

Given a parameterised (state or path) formula ϕ with variables $\mathtt{x}_1, \ldots, \mathtt{x}_n$ and a map $\mathfrak{v} : \{\mathtt{x}_1, \ldots, \mathtt{x}_n\} \to (\mathbb{N} \cup \{\omega\})$, we write $\mathfrak{v}(\phi)$ to denote the formula in RB±ATL* obtained from ϕ by replacing each occurrence of a variable $\mathtt{x}$ by $\mathfrak{v}(\mathtt{x})$. The *parameterised model-checking problem for ParRB±ATL** is as follows: given $k, r \geq 1$ (in unary), a state formula ϕ in ParRB±ATL$^*([1, k], r)$, a finite RB±ATL$^*([1, k], r)$ model $\mathfrak{M}$ and a state s, compute the set of maps $\mathfrak{v}$ such that $\mathfrak{M}, s \models_{\mathfrak{s}} \mathfrak{v}(\phi)$. Here, computing means to be able to characterise the set of maps $\mathfrak{v}$ with $\mathfrak{M}, s \models_{\mathfrak{s}} \mathfrak{v}(\phi)$, by using a symbolic representation with nice computational properties. We can show that we only need Boolean combinations of atomic formulae of the form either $\mathtt{x} \geq k$ where $k \in \mathbb{N}$ or $\mathtt{x} = \omega$.

4 The Complexity of RB±ATL

The results in this section are obtained by elaborating on correspondences between AVASS decision problems, and the existence of strategies in RB±ATL. We show a 2EXPTIME-hardness result by a reduction from the state reachability problem for AVASS. This improves the EXPSPACE-hardness result in [4]. It is also worth noting that in the proof of Theorem 1, the presence of the idle action requires a bit of work.

Theorem 1. *The model-checking problem for RB±ATL is* 2EXPTIME*-hard.*

The upper bound is proved by designing a labelling algorithm as done in [4] but the main difference with [4] rests on the fact that we explicitly call subroutines that solve decision problems on AVASS. Let $\mathfrak{M}$ be a finite resource-bounded concurrent game structure, $A \subseteq Agt$ be a coalition, F_A be a strategy and s be a state. We construct an AVASS $\mathcal{A}_{\mathfrak{M},A,s}$ such that the set of computations respecting F_A and starting from s corresponds to a derivation skeleton whose root is labelled by a unary rule with first state s. Moreover, if F_A is $\boldsymbol{b}$-strategy w.r.t. s, then the derivation skeleton can be turned into a proof whose root is labelled by $(s, \boldsymbol{b})$.

Given $\mathfrak{M} = (Agt, S, Act, r, \mathtt{act}, \mathtt{cost}, \delta, L)$, the AVASS $\mathcal{A}_{\mathfrak{M},A,s} \stackrel{\text{def}}{=} (Q, r, R_1, R_2)$ is built as follows:

$$Q \stackrel{\text{def}}{=} \{s\} \cup \{(s', \mathfrak{f}) \mid s' \in S,\ \mathfrak{f} \in D_A(s')\} \cup \{(\mathfrak{g}, s') \mid s', s'' \in S,\ \mathfrak{g} \in D(s''),\ \delta(s'', \mathfrak{g}) = s'\}.$$

– The set R_1 contains the following rules: (1) for all $\mathfrak{f} \in D_A(s)$, $(s, \mathtt{cost}_A(s,\mathfrak{f}), (s,\mathfrak{f}))$ and (2) for all $(\mathfrak{g}, s') \in Q$, for all $\mathfrak{f} \in D_A(s')$, $((\mathfrak{g}, s'), \mathtt{cost}_A(s',\mathfrak{f}), (s',\mathfrak{f}))$.
– The set R_2 contains the following rules. For all $(s',\mathfrak{f}) \in Q$, let $\{(\mathfrak{g}_1, s_1), \ldots, (\mathfrak{g}_\alpha, s_\alpha)\}$ $= \{(\mathfrak{g}, s'') \in Q \mid s'' \in \delta(s', \mathfrak{g}),\ \mathfrak{g} \in D(s'),\ \mathfrak{f} \sqsubseteq \mathfrak{g}\}$. The set is non-empty thanks to constraints on the idle action. We add the fork rule $((s',\mathfrak{f}), (\mathfrak{g}_1, s_1), \ldots, (\mathfrak{g}_\alpha, s_\alpha))$. In order to define unambiguously that rule, we assume an arbitrary ordering on Q.

It is worth noting that s has a special status in Q simply because any proof whose root configuration contains s has no predecessor configuration. Any derivation skeleton from $\mathcal{A}_{\mathfrak{M},A,s}$ has to alternate the rules in R_1 and the rules in R_2, by construction. For every $(s',\mathfrak{f})$ in Q, there is a unique fork rule starting by $(s',\mathfrak{f})$ and the construction applies also in the degenerate cases, i.e. when $A = Agt$ or when $A = \emptyset$ (assuming that $\mathtt{cost}(s',\mathfrak{f}) = \mathbf{0}$ for the unique $\mathfrak{f} \in D_\emptyset(s')$). The main property of $\mathcal{A}_{\mathfrak{M},A,s}$ is stated below.

Lemma 2. *There is a $\boldsymbol{b}$-strategy w.r.t. s in $\mathfrak{M}$ iff there is a proof in $\mathcal{A}_{\mathfrak{M},A,s}$ whose root is labelled by $(s,\boldsymbol{b})$ and every maximal branch is infinite.*

Theorem 2. *The model-checking problem for RB±ATL is in* 2EXPTIME.

An AVASS of the form $\mathcal{A}^{S'}_{\mathfrak{M},A,s}$ is defined as the restriction of $\mathcal{A}_{\mathfrak{M},A,s}$ in which the opponent coalition has no way to go out of S'. The algorithm is given below with the essential property: $GMC(\mathfrak{M},\psi) = \{s \mid \mathfrak{M}, s \models \psi\}$.

Algorithm 1 An algorithm for RB±ATL model checking.

procedure GMC($\mathfrak{M}, \phi$)
  case ϕ **of**
  p: **return** $\{s \in S \mid p \in L(s)\}$
  $\neg\psi$: **return** $S \setminus GMC(\mathfrak{M},\psi)$
  $\psi_1 \wedge \psi_2$: **return** $GMC(\mathfrak{M},\psi_1) \cap GMC(\mathfrak{M},\psi_2)$
  $\langle\langle A^b\rangle\rangle \bigcirc \psi$: **return** $\{s \mid \exists\, \mathfrak{f} \in D_A(s), \mathbf{0} \preceq \boldsymbol{b} + \mathtt{cost}_A(s,\mathfrak{f}),$ for all $\mathfrak{f} \sqsubseteq \mathfrak{g} \in D(s),\ \delta(s,\mathfrak{g}) \in GMC(\mathfrak{M},\psi)\}$
  $\langle\langle A^b\rangle\rangle \Box \psi$: $S_1 := GMC(\mathfrak{M},\psi)$
    if $s \in S_1$ **then return** $\{s \mid \mathcal{A}^{S_1}_{\mathfrak{M},A,s}, (s,\boldsymbol{b})$ is non-terminating$\}$ **end if**
    if $s \notin S_1$ **then return** $\emptyset$ **end if**
  $\langle\langle A^b\rangle\rangle \psi_1 \mathcal{U} \psi_2$: **return** $\{s \mid \mathcal{A}^{S_1\cup S_2}_{\mathfrak{M},A,s}, (s,\boldsymbol{b}), S'_2$ is a positive instance of state reachability$\}$ with $S_1 = GMC(\mathfrak{M},\psi_1)$, $S_2 = GMC(\mathfrak{M},\psi_2)$, $S'_2 = \{(\mathfrak{g},s') \in Q \mid s' \in S_2\} \cup \{s' \in Q \mid s' = s,\ s \in S_2\}$
  end case
end procedure

Corollary 1. *For any fixed $r \geq 1$, the model-checking problem for RB±ATL restricted to at most r resources is in* EXPTIME. *For $r \geq 4$, the problem is* EXPTIME*-hard.*

We invoke [26, Theorem 3.4] and [13, Theorem 3.1] since for a bound r, the state reachability and the non-termination problems can be solved in EXPTIME. EXPTIME-hardness ($r \geq 4$) is due to [13, Proposition 4.2] and to the proof of Theorem 1.

5 More Path Formulae While Preserving Decidability

In this section, we study the model-checking problem for resource-bounded logics where the path formulae can be any LTL-like formula. In doing so, we also illustrate the versatility of our formalisation, by showing how it can be used to establish complexity results for the model-checking problem for the logics RBTL* and RB$\pm$ATL*.

5.1 The Logic RBTL* and its Complexity

The models of the logic RBTL* are structures of the form (Q, r, R, L) where (Q, r, R) is a VASS and L is a truth assignment built on elements of Q understood as propositional variables, so that $L(q) = \{q\}$ (see e.g [10, Section 3]). In order to fit the usual terminology, below, an infinite proof in (Q, r, R) is called a *path* or *run* and it can be represented by $\lambda = (q_0, \boldsymbol{v}_0) \to (q_1, \boldsymbol{v}_1) \ldots$. We write $\lambda[i, +\infty)$ to denote the run starting from $(q_i, \boldsymbol{v}_i)$ taken from λ as a suffix and $\lambda(i)$ to denote the configuration $(q_i, \boldsymbol{v}_i)$.

The *state formulae* ϕ and the *path formulae* Φ of RBTL* are defined by mutual recursion with the grammar (relatively to Q and r)

$$\begin{array}{rcl} \phi & ::= & q \mid \neg\phi \mid (\phi \wedge \phi) \mid \langle \boldsymbol{b} \rangle \Phi \\ \Phi & ::= & \phi \mid \neg\Phi \mid (\Phi \wedge \Phi) \mid \bigcirc \Phi \mid (\Phi \mathcal{U} \Phi) \mid \Box\Phi \end{array}$$

where $q \in Q$ and $\boldsymbol{b} \in (\mathbb{N} \cup \{\omega\})^r$. Syntactically, every state formula is also a path formula according to this grammar, and this reflects the fact that a path uniquely identifies a control state in which a formula is interpreted: its starting control state. We present the semantics for RBTL* by distinguishing the state formulae from the path formulae. The two satisfaction relations $\models_{\mathrm{s}}$ and $\models_{\mathrm{p}}$ are defined as follows (clauses for the Boolean connectives are omitted).

$$\begin{array}{ll} \mathfrak{M}, q \models_{\mathrm{s}} q' & \text{iff } q' = q \\ \mathfrak{M}, q \models_{\mathrm{s}} \langle \boldsymbol{b} \rangle \Phi & \text{iff there is an infinite run } \lambda \text{ starting at } (q, \boldsymbol{b}) \text{ such that } \mathfrak{M}, \lambda \models_{\mathrm{p}} \Phi \\ \mathfrak{M}, \lambda \models_{\mathrm{p}} \phi & \text{iff } \mathfrak{M}, q_0 \models_{\mathrm{s}} \phi \text{ for state formulae } \phi \text{ with } \lambda(0) = (q_0, \boldsymbol{v}_0) \\ \mathfrak{M}, \lambda \models_{\mathrm{p}} \bigcirc\Phi & \text{iff } \mathfrak{M}, \lambda[1, +\infty) \models_{\mathrm{p}} \Phi \\ \mathfrak{M}, \lambda \models_{\mathrm{p}} \Phi \mathcal{U} \Psi & \text{iff there is } i \geq 0 \text{ such that } \mathfrak{M}, \lambda[i, +\infty) \models_{\mathrm{p}} \Psi \text{ and} \\ & \quad \text{for every } j \in [0, i-1], \text{ we have } \mathfrak{M}, \lambda[j, +\infty) \models_{\mathrm{p}} \Phi \\ \mathfrak{M}, \lambda \models_{\mathrm{p}} \Box\Phi & \text{iff for all } i \geq 0, \mathfrak{M}, \lambda[i, +\infty) \models_{\mathrm{p}} \Phi. \end{array}$$

The model-checking problem for RBTL* is as follows: given a model $\mathfrak{M}$, q and a state formula ϕ, is it $\mathfrak{M}, q \models_{\mathrm{s}} \phi$? The logic RBTL is the fragment of RBTL* in which any subformula whose outermost connective is in $\{\mathcal{U}, \bigcirc, \Box\}$, is preceded by some $\langle \boldsymbol{b} \rangle$. The problem for RBTL is already EXPSPACE-hard since the state reachability problem for VASS can be reduced easily to it. The EXPSPACE lower bound for the model-checking problem for RBTL can be matched with the upper bound for RBTL*.

Theorem 3. *The model-checking problems for RBTL and RBTL* are* EXPSPACE*-complete.*

We can obtain a improved complexity result if the number of resources is considered fixed.

Corollary 2. *For any fixed $r \geq 1$, the model-checking problem for RBTL* restricted to at most r resources is in* PSPACE.

The PSPACE upper bound is then a consequence of [21, Theorem 4.1]. Again, if r is fixed but greater than two, then the model-checking problem for RBTL* restricted to at most r resources is PSPACE-hard since the state reachability problem for VASS of dimension two is PSPACE-complete [8]. When $r = 1$, the model-checking problem for RBTL* restricted to at most one resource is NP-hard since the state reachability for VASS of dimension one is NP-complete [20].

5.2 Decidability of RB±ATL*

In order to illustrate the reduction from the model-checking problem for RB±ATL* into the parity game problem, we briefly present a notion of synchronisation. Let $\mathfrak{M} = (Agt, S, Act, r, \mathtt{act}, \mathtt{cost}, \delta, L)$ be a resource-bounded concurrent game structure. Given $p_1, \ldots, p_n$, we write Σ_n to denote $\mathcal{P}(\{p_1, \ldots, p_n\})$ and $L_n(s') \stackrel{\text{def}}{=} \{p_i \mid i \in [1,n],\ s' \in L(p_i)\}$ for all $s' \in S$. So, $L_n(s') \in \Sigma_n$.

Let $\mathcal{A}_{\mathfrak{M},A,s} = (Q, r, R_1, R_2)$ be the AVASS defined from $\mathfrak{M}$, A and s, and let $\mathbb{A} = (Q', q'_0, \delta' : Q' \times \Sigma_n \to Q', \mathtt{col} : Q' \to [0,p])$ be a deterministic parity automaton over Σ_n. The principle of the *synchronised product* $\mathcal{A}_{\mathfrak{M},A,s} \otimes \mathbb{A}$ defined below is the following. Any (infinite) branch of a proof of $\mathcal{A}_{\mathfrak{M},A,s}$ contains control states of the form s, $(s', \mathfrak{f})$ or $(\mathfrak{g}, s')$ where s is a distinguished state of $\mathfrak{M}$, s' is any state, $\mathfrak{f} \in D_A(s')$ and $\mathfrak{g}$ is a joint action in $D(s'')$ with $\delta(s'', \mathfrak{g}) = s'$. By construction, $(s', \mathfrak{f})$ is preceded by a state of the form either $(\mathfrak{g}, s')$ or s' (if $s' = s$). So an infinite branch of the form $(s_0, \boldsymbol{u}_0)\ ((s_0, \mathfrak{f}_0), \boldsymbol{u}_1)\ ((\mathfrak{g}_1, s_1), \boldsymbol{u}_1)\ ((s_1, \mathfrak{f}_1), \boldsymbol{u}_2)\ ((\mathfrak{g}_2, s_2), \boldsymbol{u}_2) \cdots$ leads to the ω-word $L_n(s_0)\ L_n(s_1)\ L_n(s_2) \cdots$ that admits a unique run in $\mathbb{A}$ (thanks to determinism). Above, we slightly abuse notation since we identify a branch with its label. Given an infinite branch $s_0 \xrightarrow{\boldsymbol{u}_0} (s_0, \mathfrak{f}_0) \to (\mathfrak{g}^1_{k_1}, s^1_{k_1}) \xrightarrow{\boldsymbol{u}_1} (s^1_{k_1}, \mathfrak{f}_1) \to (\mathfrak{g}^2_{k_2}, s^2_{k_2}) \xrightarrow{\boldsymbol{u}_2} (s^2_{k_2}, \mathfrak{f}_2) \to (\mathfrak{g}^3_{k_3}, s^3_{k_3}) \cdots$ in a proof of $\mathcal{A}_{\mathfrak{M},A,s}$, its *$L_n$-projection* is simply defined as the ω-word $L_n(s_0)\ L_n(s^1_{k_1})\ L_n(s^2_{k_2})\ L_n(s^3_{k_3}) \cdots$ in Σ_n^ω.

The control states of $\mathcal{A}_{\mathfrak{M},A,s} \otimes \mathbb{A}$ are pairs in $Q \times Q'$ and the second components are therefore control states in Q' as they appear for the unique run on $L_n(s_0)\ L_n(s_1)\ L_n(s_2) \cdots$.

Let us define the AVASS $\mathcal{A}_{\mathfrak{M},A,s} \otimes \mathbb{A} \stackrel{\text{def}}{=} (Q'', r, R'_1, R'_2)$ such that $Q'' \stackrel{\text{def}}{=} Q \times Q'$ and:

- For each $s \xrightarrow{\boldsymbol{u}} (s, \mathfrak{f}) \in R_1$, R'_1 contains the unary rule $(s, q'_0) \xrightarrow{\boldsymbol{u}} ((s, \mathfrak{f}), q'_0)$.
- For each $(\mathfrak{g}, s') \xrightarrow{\boldsymbol{u}} (s', \mathfrak{f}) \in R_1$, and for each $q \in Q'$, R'_1 contains the rule $((\mathfrak{g}, s'), q) \xrightarrow{\boldsymbol{u}} ((s', \mathfrak{f}), q)$. So, firing a unary rule from $\mathcal{A}_{\mathfrak{M},A,s}$ does not change the second component.
- For each $((s', \mathfrak{f}), (\mathfrak{g}_1, s_1), \ldots, (\mathfrak{g}_\alpha, s_\alpha)) \in R_2$ and for each $q \in Q'$, we add in R'_2 $(((s', \mathfrak{f}), q), ((\mathfrak{g}_1, s_1), \delta(q, L_n(s'))), \ldots, ((\mathfrak{g}_\alpha, s_\alpha), \delta(q, L_n(s'))))$. Firing a fork rule from $\mathcal{A}_{\mathfrak{M},A,s}$ changes the second component in a unique way depending on q and $L_n(s')$. Again, there is a unique fork rule starting by the control state $((s', \mathfrak{f}), q)$.

Let us define the colouring $\mathtt{col}' : Q'' \to [0,p]$ such that for all $(q, q') \in Q''$, we have $\mathtt{col}'((q, q')) \stackrel{\text{def}}{=} \mathtt{col}(q')$. The synchronised product satisfies the essential property for

the automata-based approach (as for temporal logics). This is the most natural way to inherit colours from $\mathbb{A}$ to $\mathcal{A}_{\mathfrak{M},A,s} \otimes \mathbb{A}$.

Lemma 3. *Let* $(s, \boldsymbol{b}) \in Q \times (\mathbb{N} \cup \{\omega\})^r$. *The statements below are equivalent:*

(I) $\mathcal{A}_{\mathfrak{M},A,s}$ *has a proof the root of which is equal to* $(s, \boldsymbol{b})$*, all the maximal branches are infinite and the* L_n*-projection of each infinite branch belongs to the language accepted by* $\mathbb{A}$ *(i.e. to* $\mathrm{L}(\mathbb{A})$*).*

(II) $\mathcal{A}_{\mathfrak{M},A,s} \otimes \mathbb{A}$ *has a proof the root of which is equal to* $((s, q'_0), \boldsymbol{b})$*, all the maximal branches are infinite and the maximal colour that appears infinitely often is even.*

Theorem 4. *The model-checking problem for RB±ATL* is decidable.*

Lemma 3 is essential to establish Theorem 4 since its proof uses the product between an alternating VASS and a deterministic parity automaton recognizing ω-words. This is reminiscent of the proof of [5, Theorem 5.6] about the 2EXPTIME upper bound for the ATL* model-checking problem. Rabin tree automata of the proof of [5, Theorem 5.6] are replaced by deterministic parity automata for encoding the LTL formulae and by alternating VASS (with counters) as outcome of the synchronisation.

Theorem 5. *The parameterised model-checking problem for ParRB±ATL* is decidable.*

The proof of Theorem 5 is based on a global model-checking algorithm that is essentially based on Lemma 3 and on [1, Theorem 4]. Synthesising resource values has been also considered in [25].

6 Concluding Remarks

We have related model-checking problems for resource-bounded logics and decision problems for AVASS. Though such relationships should not come as a complete surprise, we obtained new complexity and decidability results. We prove that the model-checking problem for RB±ATL introduced in [3,4] is 2EXPTIME-complete. No complexity upper bound was known so far. We have introduced the logic RB±ATL* that extends RB±ATL, and we have shown that the model-checking problem is decidable. The same hold for the parameterised version ParRB±ATL*, i.e. it is decidable to compute the set of resource bounds for which the given parameterised formula is satisfied. We have also shown that the model-checking problem for RBTL* introduced in [10] is EXPSPACE-complete. No complexity upper bound for RBTL was known so far as well as the decidability status for RBTL*.We believe that the simple framework we have proposed could be used to obtain further results for new resource-bounded logics.

Acknowledgements We would like to thank the anonymous reviewers for their numerous suggestions that helped us improve the quality of the paper.

References

1. Abdulla, P., Mayr, R., Sangnier, A., Sproston, J.: Solving Parity Games on Integer Vectors. In: CONCUR'13. LNCS, vol. 8052, pp. 106–120. Springer (2013)
2. Alechina, N., Bulling, N., Logan, B., Nguyen, H.: 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)
3. Alechina, N., Logan, B., Nguyen, H., Raimondi, F.: Decidable model-checking for a resource logic with production of resources. In: ECAI'14. pp. 9–14 (2014)
4. Alechina, N., Logan, B., Nguyen, H., Raimondi, F.: Technical report: Model-checking for resource-bounded ATL with production and consumption of resources. CoRR abs/1504.06766 (2015)
5. Alur, R., Henzinger, T., Kupferman, O.: Alternating-time temporal logic. JACM 49(5), 672–713 (2002)
6. Bérard, B., Haddad, S., Sassolas, M., Sznajder, N.: Concurrent games on VASS with inhibition. In: CONCUR'12. LNCS, vol. 7454, pp. 39–52. Springer (2012)
7. Blockelet, M., Schmitz, S.: Model-checking coverability graphs of vector addition systems. In: MFCS'11. LNCS, vol. 6907, pp. 108–119. Springer (2011)
8. Blondin, M., Finkel, A., Göller, S., Haase, C., McKenzie, P.: Reachability in two-dimensional vector addition systems with states is PSPACE-complete. In: LICS'15. pp. 32–43. ACM Press (2015)
9. Brázdil, T., Jančar, P., Kucera, A.: Reachability games on extended vector addition systems with states. In: ICALP'10. LNCS, vol. 6199, pp. 478–489. Springer (2010)
10. Bulling, N., Farwer, B.: Expressing properties of resource-bounded systems: The logics RBTL* and RBTL. In: CLIMA X. LNCS, vol. 6214, pp. 22–45. Springer (2009)
11. Bulling, N., Farwer, B.: On the (Un-)Decidability of Model-Checking Resource-Bounded Agents. In: ECAI'10. pp. 567–572 (2010)
12. Bulling, N., Nguyen, H.: Model checking resource bounded systems with shared resources via alternating Büchi pushdown systems. In: PRIMA'15. LNCS, vol. 9387, pp. 640–649 (2015)
13. Courtois, J., Schmitz, S.: Alternating vector addition systems with states. In: MFCS'14. LNCS, vol. 8634, pp. 220–231. Springer (2014)
14. Demri, S.: On selective unboundedness of VASS. JCSS 79(5), 689–713 (2013)
15. Demri, S., Jurdziński, M., Lachish, O., Lazić, R.: The covering and boundedness problems for branching vector addition systems. JCSS 79(1), 23–38 (2013)
16. Emerson, A.: Temporal and modal logic. In: Handbook of Theoretical Computer Science. pp. 996–1072. Elsevier (1990)
17. Esparza, J.: On the decidability of model checking for several μ-calculi and Petri nets. In: ICALP'94. LNCS, vol. 787, pp. 115–129. Springer (1994)
18. Esparza, J.: Decidability and complexity of Petri net problems — an introduction. In: Advances in Petri Nets 1998. LNCS, vol. 1491, pp. 374–428. Springer (1998)
19. Göller, S., Lohrey, M.: Branching-time model checking of one-counter processes and timed automata. SIAM Journal of Computing 42(3), 884–923 (2013)
20. Haase, C.: On the Complexity of Model Checking Counter Automata. Ph.D. thesis, University of Oxford (2012)
21. Habermehl, P.: On the complexity of the linear-time mu-calculus for Petri nets. LNCS, vol. 1248, pp. 102–116. Springer (1997)
22. Howell, R., Rosier, L.: Problems concerning fairness and temporal logic for conflict-free Petri nets. TCS 64, 305–329 (1989)
23. Jančar, P.: Decidability of a temporal logic problem for Petri nets. TCS 74(1), 71–93 (1990)

24. Jančar, P.: On reachability-related games on vector addition systems with states. In: RP'15. LNCS, vol. 9328, pp. 50–62. Springer (2015)
25. Juhl, L., Larsen, K., Raskin, J.F.: Optimal bounds for multiweighted and parametrised energy games. In: Theories of Programming and Formal Methods - Essays Dedicated to Jifeng He on the Occasion of His 70th Birthday. LNCS, vol. 8051, pp. 244–255. Springer (2013)
26. Jurdziński, M., Lazić, R., Schmitz, S.: Fixed-dimensional energy games are in pseudo-polynomial time. In: ICALP'15. LNCS, vol. 9135, pp. 260–272. Springer (2015)
27. Karp, R., Miller, R.: Parallel program schemata. JCSS 3(2), 147–195 (1969)
28. Lipton, R.: The reachability problem requires exponential space. Tech. Rep. 62, Department of Computer Science, Yale University (1976)
29. Monica, D.D., Napoli, M., Parente, M.: On a logic for coalitional games with priced-resource agents. ENTCS 278, 215–228 (2011)
30. Rackoff, C.: The covering and boundedness problems for vector addition systems. TCS 6(2), 223–231 (1978)
31. Raskin, J., Samuelides, M., Begin, L.V.: Games for counting abstractions. ENTCS 128(6), 69–85 (2005)
32. Serre, O.: Parity games played on transition graphs of one-counter processes. In: FOSSACS'06. LNCS, vol. 3921, pp. 337–351. Springer (2006)
33. Verma, K., Goubault-Larrecq, J.: Karp-Miller Trees for a Branching Extension of VASS. Discrete Mathematics and Theoretical Computer Science 7, 217–230 (2005)
34. Vester, S.: On the complexity of model-checking branching and alternating-time temporal logics in one-counter systems. In: ATVA'15. LNCS, vol. 9364, pp. 361–377. Springer (2015)