Symbolic Model Checking for One-Resource RB±ATL[†]

Natasha Alechina[1] and **Brian Logan**[1] and **Hoang Nga Nguyen**[1] and **Franco Raimondi**[2]
[1] School of Computer Science, University of Nottingham, UK
[2] Department of Computer Science, Middlesex University, UK
[1] {nza,bsl,hnn}@cs.nott.ac.uk, [2] f.raimondi@mdx.ac.uk

Abstract

RB±ATL is an extension of ATL where it is possible to model consumption and production of several resources by a set of agents. The model-checking problem for RB±ATL is known to be decidable. However the only available model-checking algorithm for RB±ATL uses a forward search of the state space, and hence does not have an efficient symbolic implementation. In this paper, we consider a fragment of RB±ATL, 1RB±ATL, that allows only one resource type. We give a symbolic model-checking algorithm for this fragment of RB±ATL, and evaluate the performance of an MCMAS-based implementation of the algorithm on an example problem that can be scaled to large state spaces.

1 Introduction

Alternating Time Temporal Logic (ATL) [Alur *et al.*, 2002] is widely used to characterise coalitional abilities in multi-agent systems. Recently, several extensions of ATL have been proposed that can express coalitional ability under resource constraints, for example [Alechina *et al.*, 2009; Bulling and Farwer, 2010; Alechina *et al.*, 2010; Della Monica *et al.*, 2011; 2013; Bulling and Goranko, 2013; Alechina *et al.*, 2014]. These logics allow reasoning about coalitional ability under resource constraints such as: 'given resources b, coalition A has a strategy to achieve φ'. There are many different variants of ATL with resources. If resource production is allowed, the model-checking problem for many of these variants becomes undecidable [Bulling and Farwer, 2010; Bulling and Goranko, 2013]. For resource logics where the model-checking problem is known to be decidable, the only symbolic model-checking algorithm that has been proposed to date is that given in [Alechina *et al.*, 2015], which describes an implementation of symbolic model-checking for the resource-consumption only logic RB-ATL. Symbolic model-checking uses Binary Decision Diagrams (BDDs) [Bryant, 1986] to represent sets of states symbolically, and over the last 20 years it has been successfully employed as a powerful optimisation technique in a range of model checkers such as MCK [Gammie and Van Der Meyden, 2004] and MCMAS [Lomuscio *et al.*, 2009].

In this paper we provide a symbolic model-checking algorithm for a fragment of the logic RB±ATL. RB±ATL does have a decidable model-checking problem, however it is EXPSPACE-hard [Alechina *et al.*, 2014]. The model-checking algorithm proposed by Alechina et al. [2014] uses a forward search of the state space, and hence does not have an efficient symbolic implementation. In this paper, we consider a fragment of RB±ATL, 1RB±ATL, that allows only one resource type, for example, money. This is a natural special case of resource logics, since in many situations only one resource is of interest, or multiple resources are mutually convertible. For example, in [Della Monica *et al.*, 2013] it is assumed that all resources can be converted into money. The main contribution of this paper is a fixed point characterisation of 1RB±ATL modalities. This makes it possible to provide a symbolic model-checking algorithm for 1RB±ATL. We analyse the complexity of the model-checking algorithm, and also evaluate the performance of an MCMAS-based [Lomuscio *et al.*, 2009] implementation of the algorithm on an example problem that can be scaled to large state spaces. The evaluation suggests that the symbolic implementation is scalable and performs much better than the algorithm from [Alechina *et al.*, 2014].

2 RB±ATL

The logic RB±ATL was introduced in [Alechina *et al.*, 2014] as a generalisation of RB-ATL [Alechina *et al.*, 2010]. In contrast to RB-ATL which allows only consumption of resources, RB±ATL allows agents to both produce and consume resources. In this section, we briefly recall the syntax and semantics of RB±ATL.

Let $Agt = \{a_1, \ldots, a_n\}$ be a set of n agents, $Res = \{res_1, \ldots, res_r\}$ be a set of r resources, Π be a set of propositions and $B = \mathbb{N}_\infty^r$ be a set of resource bounds where $\mathbb{N}_\infty = \mathbb{N}_0 \cup \{\infty\}$. We denote by $\bar{0}$ the smallest bound $(0, \ldots, 0)$ and $\bar{\infty}$ the greatest bound $(\infty, \ldots, \infty)$.

[†]**Acknowledgements:** This work was supported by the Engineering and Physical Sciences Research Council [grants EP/K033905/1 and EP/K033921/1]. We would also like to thank the anonymous IJCAI'15 reviewers whose comments and suggestions helped to improve the paper.

Formulas of RB±ATL are defined by the following syntax

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid \langle\langle A^b \rangle\rangle \bigcirc \varphi \mid \langle\langle A^b \rangle\rangle \Box \varphi \mid \langle\langle A^b \rangle\rangle \varphi \mathcal{U} \psi$$

where $p \in \Pi$ is a proposition, $A \subseteq Agt$, and $b \in B$ is a resource bound. Here, $\langle\langle A^b \rangle\rangle \bigcirc \varphi$ means that a coalition A can ensure that the next state satisfies φ under resource bound b. $\langle\langle A^b \rangle\rangle \Box \varphi$ means that A has a strategy to make sure that φ is always true, and the cost of this strategy is at most b. Similarly, $\langle\langle A^b \rangle\rangle \varphi \mathcal{U} \psi$ means that A has a strategy to enforce ψ while maintaining the truth of φ, and the cost of this strategy is at most b.

To interpret this language, the definition of concurrent game structures [Alur *et al.*, 2002] is extended with resource consumption and production.

Definition 1. *A resource-bounded concurrent game structure (RB-CGS) is a tuple $M = (Agt, Res, S, \Pi, \pi, Act, d, c, \delta)$ where:*

- *Agt is a non-empty set of n agents, Res is a non-empty set of r resources and S is a non-empty set of states.*
- *Π is a finite set of propositional variables and $\pi : \Pi \to \wp(S)$ is a truth assignment that associates each proposition in Π with a subset of states where it is true.*
- *Act is a non-empty set of actions which includes $idle$, and $d : S \times Agt \to \wp(Act) \setminus \{\emptyset\}$ is a function that assigns to each $s \in S$ a non-empty set of actions available to each agent $a \in Agt$. For every $s \in S$ and $a \in Agt$, $idle \in d(s, a)$. We denote joint actions by all agents in Agt available at s by $D(s) = d(s, a_1) \times \cdots \times d(s, a_n)$.*
- *$c : S \times Agt \times Act \to \mathbb{Z}^r$ is a partial function that maps a state s, an agent a and an action $\alpha \in d(s, a)$ to a vector of integers where the integer in position i indicates consumption or production of resource res_i by the action (positive value for consumption and negative value for production). We stipulate that $c(s, a, idle) = \bar{0}$ for all $s \in S$ and $a \in Agt$.*
- *$\delta : (s, \sigma) \mapsto S$ is a function that, for every $s \in S$ and joint action $\sigma \in D(s)$, gives the state resulting from executing σ in s.*

Given a RB-CGS M, we denote the set of all infinite sequences of states (computations) by S^ω and the set of non-empty finite sequences of states by S^+. For a computation $\lambda = s_0 s_1 \ldots \in S^+ \cup S^\omega$, we use the notation $\lambda[i] = s_i$ and $\lambda[i, j] = s_i \ldots s_j \;\forall\, j \geq i \geq 0$.

Below, we use the usual point-wise notation for vector comparison and addition. In particular, $(b_1, \ldots, b_r) \leq (d_1, \ldots, d_r)$ iff $b_i \leq d_i \;\forall\, i \in \{1, \ldots, r\}$, and $(b_1, \ldots, b_r) + (d_1, \ldots, d_r) = (b_1 + d_1, \ldots, b_r + d_r)$. We assume that for any $b \in \mathbb{Z}$, $b \leq \infty$ and $b + \infty$ and $\infty - b = \infty$. Given a function f returning a vector, we denote by f_i the function that returns the i-th component of the vector returned by f.

Given a RB-CGS M and a state $s \in S$, a *joint action by a coalition* $A \subseteq Agt$ is a tuple $\sigma_A = (\sigma_a)_{a \in A}$ such that $\sigma_a \in d(s, a)$. The set of all joint actions for A at state s is denoted by $D_A(s)$. Given a joint action by the grand coalition $\sigma \in D(s)$, σ_A denotes the joint action executed by A: $\sigma_A = (\sigma_a)_{a \in A}$. The set of all possible outcomes of a joint action $\sigma_A \in D_A(s)$ at state s is: $out(s, \sigma_A) = \{s' \in S \mid \exists \sigma' \in D(s) : \sigma_A = \sigma'_A \wedge s' = \delta(s, \sigma')\}$. The cost of a joint action $\sigma_A \in D_A(s)$ is defined as $cost_A(s, \sigma_A) = \sum_{a \in A} c(s, a, \sigma_a)$.

Given a RB-CGS M, a *strategy for a coalition* $A \subseteq Agt$ is a mapping $F_A : S^+ \to Act$ such that, for every $\lambda s \in S^+$, $F_A(\lambda s) \in D_A(s)$. A computation $\lambda \in S^\omega$ is consistent with a strategy F_A iff, for all $i \geq 0$, $\lambda[i+1] \in out(\lambda[i], F_A(\lambda[0, i]))$. We denote by $out(s, F_A)$ the set of all consistent computations λ of F_A that start from s.

Given a bound $b \in B$, a computation $\lambda \in out(s, F_A)$ is b-consistent with F_A iff, for every $i \geq 0$,

$$\sum_{j=0}^{i} cost_A(\lambda[j], F_A(\lambda[0, j])) \leq b$$

Note that this definition implies that the cost of every prefix of the computation is below b. The set of all b-consistent computations of F_A starting from state s is denoted by $out(s, F_A, b)$. F_A is a b-strategy iff $out(s, F_A) = out(s, F_A, b)$ for any state s.

Given a RB-CGS M and a state s, the truth of a RB±ATL formula φ with respect to M and s is defined inductively on the structure of φ as follows (the atomic case and the Boolean connectives are defined in the standard way):

- $M, s \models \langle\langle A^b \rangle\rangle \bigcirc \varphi$ iff $\exists$ b-strategy F_A such that for all $\lambda \in out(s, F_A)$: $M, \lambda[1] \models \varphi$;
- $M, s \models \langle\langle A^b \rangle\rangle \Box \varphi$ iff $\exists$ b-strategy F_A such that for all $\lambda \in out(s, F_A)$ and $i \geq 0$: $M, \lambda[i] \models \varphi$; and
- $M, s \models \langle\langle A^b \rangle\rangle \varphi \mathcal{U} \psi$ iff $\exists$ b-strategy F_A such that for all $\lambda \in out(s, F_A)$, $\exists i \geq 0$: $M, \lambda[i] \models \psi$ and $M, \lambda[j] \models \varphi$ for all $j \in \{0, \ldots, i-1\}$.

Since the infinite resource bound version of RB±ATL modalities correspond to the standard ATL modalities, we write $\langle\langle A^{\bar{\infty}} \rangle\rangle \gamma = \langle\langle A \rangle\rangle \gamma$.

Note that although we only consider infinite paths, the condition that the $idle$ action with cost $\bar{0}$ is always available makes the model-checking problem easier (we only need to find a strategy with a finite prefix under bound b to satisfy formulas of the form $\langle\langle A^b \rangle\rangle \bigcirc \varphi$ and $\langle\langle A^b \rangle\rangle \varphi \mathcal{U} \psi$, and then the strategy can make the $idle$ choice forever). This makes the logic closer to the finitary semantics in [Bulling and Farwer, 2010].

3 Hybrid Model Checking RB±ATL

The model-checking problem for RB±ATL is the question whether, for a given RB-CGS structure M, state s and RB±ATL formula φ, $M, s \models \varphi$. An algorithm for solving the model-checking problem of RB±ATL was presented in [Alechina *et al.*, 2014], and is restated in Algorithm 1. Given a RB-CGS structure M and a formula φ_0, the algorithm returns the set $[\varphi_0]_M$ of states in M that satisfy φ, i.e., $[\varphi_0]_M = \{s \mid M, s \models \varphi_0\}$.

The algorithm makes use of the primitive operator $Sub(\varphi_0)$ which computes all sub-formulas of φ_0 in the usual way, and, in addition, if $\langle\langle A^b \rangle\rangle \gamma \in Sub(\varphi)$, its infinite resource version $\langle\langle A \rangle\rangle \gamma$ is also added to $Sub(\varphi)$. $Sub(\varphi)$ is ordered in increasing order of complexity, and the infinite

Algorithm 1 Labelling φ_0

```
function RB±ATL-LABEL($M, \varphi_0$)
    for $\varphi' \in Sub(\varphi_0)$ do
        case $\varphi' = p,\ \neg\varphi,\ \varphi \wedge \psi,$
            $\langle\langle A\rangle\rangle \bigcirc \varphi,\ \langle\langle A\rangle\rangle \varphi\,\mathcal{U}\,\psi,\ \langle\langle A\rangle\rangle \Box\varphi$
            standard, see [Alur et al., 2002]
        case $\varphi' = \langle\langle A^b\rangle\rangle \bigcirc \varphi$
            $[\varphi']_M \leftarrow Pre(A, [\varphi]_M, b)$
        case $\varphi' = \langle\langle A^b\rangle\rangle \varphi\,\mathcal{U}\,\psi$
            $[\varphi']_M \leftarrow \{\, s \mid s \in S \wedge$
            UNTIL-STRATEGY($node_0(s,b), \langle\langle A^b\rangle\rangle \varphi\,\mathcal{U}\,\psi$)$\}$
        case $\varphi' = \langle\langle A^b\rangle\rangle \Box\varphi$
            $[\varphi']_M \leftarrow \{\, s \mid s \in S \wedge$
            BOX-STRATEGY($node_0(s,b), \langle\langle A^b\rangle\rangle \Box\varphi$)$\}$
    return $[\varphi_0]_M$
```

resource version of each modal formula comes before the bounded version. Note that if a state s is not annotated with $\langle\langle A\rangle\rangle\gamma$ then s cannot satisfy the bounded resource version $\langle\langle A^b\rangle\rangle\gamma$.

The algorithm then proceeds by cases. For all formulas in $Sub(\varphi)$ apart from $\langle\langle A^b\rangle\rangle\bigcirc\varphi$, $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$ and $\langle\langle A^b\rangle\rangle\Box\varphi$, the standard ATL model-checking algorithm [Alur *et al.*, 2002] is used. The implementation of these cases can therefore be done symbolically. Labelling states with $\langle\langle A^b\rangle\rangle\bigcirc\varphi$ makes use of a function $Pre(A,\rho,b)$, which, given a coalition A, a set $\rho \subseteq S$ and a bound b, returns a set of states s in which A has a joint action σ_A with $cost(s,\sigma_A) \le b$ such that $out(s,\sigma_A) \subseteq \rho$. Labelling states with $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$ and $\langle\langle A^b\rangle\rangle\Box\varphi$ is more complex, and requires two auxiliary functions: UNTIL-STRATEGY for $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$ and BOX-STRATEGY for $\langle\langle A^b\rangle\rangle\Box\varphi$. Essentially, these functions search the existence of a strategy for A which satisfies the corresponding formula by depth-first and-or search of M. For a detailed description of these functions, see [Alechina *et al.*, 2014].

While the function $Pre(A,\rho,b)$ can be implemented symbolically as shown in [Alechina *et al.*, 2015], the last two cases call auxiliary functions that search state by state. Therefore, they can only be implemented non-symbolically, resulting in a hybrid (partially symbolic, partially non-symbolic) implementation.

4 Symbolic Model Checking 1RB±ATL

1RB±ATL is a fragment of RB±ATL with $|Res| = 1$. Resource bounds in 1RB±ATL are therefore in $\mathbb{N}_\infty$ and action costs in $\mathbb{Z}$. In this section, we present a symbolic implementation for the cases of until formulas $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$ and box formulas $\langle\langle A^b\rangle\rangle\Box\varphi$ in Algorithm 1 by providing a fixed point characterisation of these formulas. This will enable us to provide a symbolic implementation for the model-checking problem of 1RB±ATL. The of question whether a similar approach would work with more than one resource type is open. In particular it is not clear how to generalise the functions used for the fixed point characterisation of until and box formulas below. It is also not clear how, given a coalition A and a model M, to estimate the maximal possible cost of a strategy by A to satisfy an until formula (this estimate is required for the model-checking algorithm). Intuitively, if there is a cycle where resource r_1 is produced and resource r_2 is consumed, and a cycle where r_1 is consumed and r_2 is produced, it is not obvious what are the maximal amounts of r_1 and r_2 that may be needed to satisfy an until formula. This is a hard problem related to the upper bound on reachability in Petri nets (which is also an open problem, [Leroux, 2013]).

Below, we use the usual point-wise notation for comparison, intersection and union of vectors of sets. In particular, for some $k \ge 1$, $(b_1,\ldots,b_k) \subseteq (d_1,\ldots,d_k)$ iff $b_i \subseteq d_i\ \forall\, i \in \{1,\ldots,k\}$, $(b_1,\ldots,b_r) \cap (d_1,\ldots,d_r) = (b_1 \cap d_1,\ldots,b_r \cap d_r)$ and $(b_1,\ldots,b_r) \cup (d_1,\ldots,d_r) = (b_1 \cup d_1,\ldots,b_r \cup d_r)$. We drop the subscript M in $[\varphi]_M$ when this does not lead to ambiguity.

4.1 Until Formulas

In this section, we show that $[\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi]_M$ can be computed from the least fixed point of a function $U_{A,\varphi,\psi}$ which does not depend on b (only on costs of actions in M).

For each agent $i \in Agt$, we denote by $minc_i = \min\{c(s,i,a) \mid a \in Act_i, s \in S\}$ the minimal cost of any action by an agent i. Obviously, $minc_i \le 0$ due to the presence of $idle$, and the fact that there is no action for i that costs less that $minc_i$. Similarly, we denote by $minc_A = \sum_{i\in A} minc_i$ the minimal cost of any joint action by the non-empty coalition A. Again, $minc_A \le 0$ and there is no joint action for A that costs less than $minc_A$.

We extend the definition of $[\langle\langle A^b\rangle\rangle\bigcirc] : \wp(S) \to \wp(S)$ in [Alechina *et al.*, 2010] with $b \in \mathbb{Z} \cup \{\infty\}$ as $[\langle\langle A^b\rangle\rangle\bigcirc](X) = \{s \in S \mid \exists\sigma \in D_A(s) : cost(\sigma) \le b \wedge out(s,\sigma) \subseteq X\}$. Obviously, $[\langle\langle A^b\rangle\rangle\bigcirc](X) = \emptyset$ for any $b < minc_A$ and $X \subseteq S$.

Lemma 1. $[\langle\langle A^b\rangle\rangle\bigcirc]$ *is monotone.*

Proof. Assume that $X \subseteq Y$. Then:

$$\begin{aligned} s \in [\langle\langle A^b\rangle\rangle\bigcirc](X) &\Rightarrow \exists\sigma \in D_A(s) : cost(\sigma) \le b \wedge \\ &\qquad\qquad out(s,\sigma) \subseteq X \\ &\Rightarrow \exists\sigma \in D_A(s) : cost(\sigma) \le b \wedge \\ &\qquad\qquad out(s,\sigma) \subseteq X \subseteq Y \\ &\Rightarrow s \in [\langle\langle A^b\rangle\rangle\bigcirc](Y) \end{aligned}$$

Therefore, $[\langle\langle A^b\rangle\rangle\bigcirc](X) \subseteq [\langle\langle A^b\rangle\rangle\bigcirc](Y)$. $\Box$

Next we define a predecessor to the function $U_{A,\varphi,\psi}$ that does depend on the bound b:

$$U_{A^b,\varphi,\psi}((X_0,\ldots,X_{b-minc_A})) = [\psi] \cup [\varphi] \cap \Big(\bigcup_{minc_A \le b_1 \le b} [\langle\langle A^{b_1}\rangle\rangle\bigcirc](X_{b-b_1}) \Big)$$

For convenience, we drop the subscripts φ,ψ in the remainder of this section. Note that $U_{A^b}(([\langle\langle A^0\rangle\rangle\varphi\,\mathcal{U}\,\psi],\ldots,[\langle\langle A^{b-minc_A}\rangle\rangle\varphi\,\mathcal{U}\,\psi])) = [\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi]$.

Observe that since $minc_A \le 0$ and $b \ge 0$, we always have $0 \le b - b_1 \le b - minc_A$ for all $b_1 \in \{minc_A,\ldots,b\}$.

Lemma 2. *U_{A^b} is monotone.*

Proof. Assume that $X = (X_0, \dots, X_{b-minc_A})$, $Y = (Y_0, \dots, Y_{b-minc_A})$ and $X \subseteq Y$, i.e., $X_i \subseteq Y_i$ for all $0 \leq i \leq b - minc_A$. Then,

$$\begin{aligned} U_{A^b}(X) &= [\psi] \cup ([\varphi] \cap \bigcup_{minc_A \leq b_1 \leq b} [\langle\langle A^{b_1}\rangle\rangle \bigcirc](X_{b-b_1})) \\ &\subseteq [\psi] \cup ([\varphi] \cap \bigcup_{minc_A \leq b_1 \leq b} [\langle\langle A^{b_1}\rangle\rangle \bigcirc](Y_{b-b_1})) \\ &\qquad \text{by } X_i \subseteq Y_i \text{ for all } 0 \leq i \leq b - minc_A \text{ and Lemma 1} \\ &= U_{A^b}(Y) \end{aligned}$$ ☐

Lemma 3. *There exists a bound* $\min \in \mathbb{N}$ *such that* $[\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi] \supseteq [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$ *for any bound* k.

Proof. Consider an infinite sequence of bounds $0, 1, \dots$. As $\wp(S)$ is finite, there is an infinite subsequence $0 \leq i_1 < i_2 < \dots$ such that $[\langle\langle A^{i_1}\rangle\rangle \varphi \,\mathcal{U}\, \psi] = [\langle\langle A^{i_2}\rangle\rangle \varphi \,\mathcal{U}\, \psi] = [\langle\langle A^{i_3}\rangle\rangle \varphi \,\mathcal{U}\, \psi] = \dots$

Let $\min = i_1$. For any k, there exists $j \geq 1$ such that $k \leq i_j$. Then $[\langle\langle A^{i_j}\rangle\rangle \varphi \,\mathcal{U}\, \psi] \supseteq [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$ As $[\langle\langle A^{i_1}\rangle\rangle \varphi \,\mathcal{U}\, \psi] = [\langle\langle A^{i_j}\rangle\rangle \varphi \,\mathcal{U}\, \psi]$, we have that $[\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi] \supseteq [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$. ☐

From Lemma 3, $[\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi] = [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$ for $k \geq \min$. As a result, we define the following function which is similar to U_{A^k} but accepts input vectors of the same size $min + 1$ for any $k \geq 0$:

$$\hat{U}_{A^k,\varphi,\psi}((X_0, \dots, X_{\min})) = [\psi] \cup ([\varphi] \cap \bigcup_{minc_A \leq k_1 \leq k} [\langle\langle A^{k_1}\rangle\rangle \bigcirc](\begin{cases} X_{k-k_1} & \text{if } k - k_1 \leq \min \\ X_{\min} & \text{otherwise} \end{cases}))$$

By Lemma 2, it is straightforward that $\hat{U}_{A^k}$ is also monotone.

Finally, we define the function used for fixed-point characterisation of Until:

$$\begin{aligned} &U_{A,\varphi,\psi}((X_0, \dots, X_{\min})) \\ &\quad = (\hat{U}_{A^k}((X_{A^0}, \dots, X_{\min})))_{0 \leq k \leq \min} \end{aligned}$$

Lemma 4. *U_A is monotone.*

Proof. This is straightforward as, by Lemma 2, $\hat{U}_{A^k}$ is monotone for any k. ☐

Lemma 5. *$([\langle\langle A^0\rangle\rangle \varphi \,\mathcal{U}\, \psi], \dots, [\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi])$ is the least fixed point of U_A.*

Proof. The Lemma follows from Claims 1 and 2 below by Knaster-Tarski's fixed-point theorem. ☐

Claim 1. *$X = ([\langle\langle A^0\rangle\rangle \varphi \,\mathcal{U}\, \psi], \dots, [\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi])$ is a prefixed point of U_A.*

Proof. To show that X is the prefixed point of U_A, we need to prove that $U_A(X) \subseteq X$, i.e., $\hat{U}_{A^k}(X) \subseteq X_k$ for all $0 \leq k \leq \min$:

$$\begin{aligned} s \in \hat{U}_{A_k}(X) &\Rightarrow s \in [\psi] \cup ([\varphi] \cap \\ &\bigcup_{minc_A \leq k_1 \leq k} [\langle\langle A^{k_1}\rangle\rangle \bigcirc](\begin{cases} X_{k-k_1} & \text{if } k - k_1 \leq \min \\ X_{\min} & \text{otherwise} \end{cases})) \\ &\Rightarrow s \in [\psi] \cup ([\varphi] \cap \\ &\qquad \bigcup_{minc_A \leq k_1 \leq k} [\langle\langle A^{k_1}\rangle\rangle \bigcirc](X_{k-k_1})) \\ &\qquad \text{as } X_{k-k_1} = X_{\min} \text{ for all } k - k_1 > \min \\ &\Rightarrow s \in [\psi] \text{ or } s \in ([\varphi] \cap \\ &\qquad \bigcup_{minc_A \leq k_1 \leq k} [\langle\langle A^{k_1}\rangle\rangle \bigcirc](X_{k-k_1})) \\ &\Rightarrow s \in [\psi] \text{ or } \exists k_1 \in \{minc_A, k\} : \\ &\qquad s \in [\varphi] \cap [\langle\langle A^{k_1}\rangle\rangle \bigcirc](X_{k-k_1}) \end{aligned}$$

If $s \in [\psi]$, then it is obvious that $s \in X_k = [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$.

If $s \in [\varphi] \cap [\langle\langle A^{k_1}\rangle\rangle \bigcirc](X_{k-k_1}) = [\langle\langle A^{k_1}\rangle\rangle \bigcirc]([\langle\langle A^{k-k_1}\rangle\rangle \varphi \,\mathcal{U}\, \psi])$, then there exists a joint action $\sigma \in D_A(s)$ such that $cost(\sigma) \leq k_1$ and $out(s, \sigma) \subseteq [\langle\langle A^{k-k_1}\rangle\rangle \varphi \,\mathcal{U}\, \psi]$. Then, for all $s' \in out(s, \sigma)$, there exists a $k - k_1$-strategy $F_{s'}$ to satisfy $\langle\langle A^{k-k_1}\rangle\rangle \varphi \,\mathcal{U}\, \psi$. Let us consider the strategy F where $F(s) = \sigma$ and $F(ss'\lambda) = F_{s'}(s'\lambda)$ for all $\lambda \in S^*$. Obviously, F is a k-strategy to satisfy $\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi$. Therefore, $s \in [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi]$. ☐

Claim 2. *Let $X = ([\langle\langle A^0\rangle\rangle \varphi \,\mathcal{U}\, \psi], \dots, [\langle\langle A^{\min}\rangle\rangle \varphi \,\mathcal{U}\, \psi])$. For any pre-fixed point Y of U_A, $Y \supseteq X$.*

Proof. We have that $Y \supseteq U_A(Y)$; as U_A is monotone, it also implies $Y \supseteq U_A(Y) \supseteq U_A^2(Y) \supseteq \dots \supseteq U_A^i(Y) \supseteq \dots$ In order to prove $Y \supseteq X$, we show that for all $0 \leq k \leq \min$, there exists $i : X_k \subseteq \hat{U}^i_{A^k}(Y)$ where $\hat{U}^i_{A^k}(Y)$ is the shorthand for $\hat{U}_{A^k}(U_A^{i-1}(Y))$. First,

$$\begin{aligned} s \in X_k &\Rightarrow s \in [\langle\langle A^k\rangle\rangle \varphi \,\mathcal{U}\, \psi] \\ &\Rightarrow s \in \exists\ k\text{-strategy } F_A : \forall \lambda \in out(s, F_A) : \\ &\qquad \exists l_{s,\lambda} \geq 0 : \lambda[l_{s,\lambda}] \models \psi \wedge \forall l' < l_{s,\lambda} : \lambda[l'] \models \varphi \end{aligned}$$

Let $l_{s,k} = \max\{l_{s,\lambda} \mid \lambda \in out(s, F_A)\}$, we continue the proof by induction on $l_{s,k}$ that $s \in \hat{U}^l_{A^k}(Y)$ for all $l \geq l_{s,k}$.

Base case: If $l_{s,k} = 0$, then $s \in [\psi]$, hence $s \in \hat{U}^l_{A^k}(Y)$ for any $l \geq 1$ by definition of $\hat{U}_{A^k}$.

Induction step: If $l_{s,k} > 0$, then $s \in [\varphi] \cap [\langle\langle A^{k_1}\rangle\rangle \bigcirc](X_{k-k_1})$ for some $minc_A \leq k_1 \leq k$, i.e., $\exists \sigma \in D_A(s) : cost(s, \sigma) \leq k_1 \wedge out(s, \sigma) \subseteq X_{k-k_1}$. Then $l_{s',k-k_1} < l_{s,k}$ (i.e., $l_{s',k-k_1} \leq l_{s,k} - 1$) for all $s' \in out(s, \sigma) \subseteq X_{k-k_1}$. By the induction hypothesis, we have that $s' \in \hat{U}^l_{A^{k-k_1}}(Y)$ for all $l \geq l_{s',k-k_1}$; then, $s' \in \hat{U}^l_{A^{k-k_1}}(Y)$ for all $l \geq l_{s,k} - 1$ and $s' \in out(s, \sigma)$.

This means for all $l \geq l_{s,k} - 1$ we have:

$$\begin{aligned} out(s,\sigma) \subseteq \hat{U}^{l}_{A^{k-k_1}}(Y) &\Rightarrow [\langle\langle A^{k_1}\rangle\rangle\bigcirc](out(s,\sigma)) \subseteq \\ &\quad [\langle\langle A^{k_1}\rangle\rangle\bigcirc](\hat{U}^{l}_{A^{k-k_1}}(Y)) \\ &\quad \text{as } [\langle\langle A^{k_1}\rangle\rangle\bigcirc] \text{ is monotone} \\ &\Rightarrow s \in [\varphi] \cap \\ &\quad [\langle\langle A^{k_1}\rangle\rangle\bigcirc](\hat{U}^{l}_{A^{k-k_1}}(Y)) \\ &\Rightarrow s \in \hat{U}^{l+1}_{A^{k}}(Y) \end{aligned}$$

Therefore, $s \in \hat{U}^{l}_{A^k}(Y)$ for all $l \geq l_{s,k}$. Let $l_k = \max\{l_{s,k} \mid s \in X_k\}$, we have that $s \in \hat{U}^{l}_{A^k}(Y)$ for all $l \geq l_k$ and $s \in X_k$, i.e., $X_k \subseteq \hat{U}^{l_k}_{A^k}(Y)$. □

As usual, the least fixed point $(X_k)_{0 \leq k \leq \min}$ of U_A can be computed by repeatedly applying U_A to $(\emptyset, \ldots, \emptyset)$ until a fixed point is reached. Then,

$$[\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi] = \begin{cases} X_b & \text{if } b \leq \min \\ X_{\min} & \text{otherwise.} \end{cases}$$

It remains to find $\min$. A coarse estimation is that $\min = maxc_A \times |S|$ where $maxc_A = \sum_{i \in A} maxc_i$ with $maxc_i = \max\{c(s,i,a) \mid a \in Act_i, s \in S\}$, i.e. the maximal cost of any action of any agents in A. The idea behind this estimation is that any strategy to satisfy $\langle\langle A^{\min}\rangle\rangle\varphi\,\mathcal{U}\,\psi$ can be converted into a strategy that does not revisit any states (does not contain loops) on a path to a state where ψ is satisfied. This is made precise by the following lemma.

Lemma 6. *If $s \in \langle\langle A^k\rangle\rangle\varphi\,\mathcal{U}\,\psi$ for some $k \in \mathbb{N}$ then $s \in \langle\langle A^{maxc_A \times |S|}\rangle\rangle\varphi\,\mathcal{U}\,\psi$.*

Proof. As $s \in \langle\langle A^k\rangle\rangle\varphi\,\mathcal{U}\,\psi$, there exists a k-strategy F_A such that for all $\lambda \in out(s, F_A)$: $\exists l_\lambda \geq 0 : \lambda[l_\lambda] \models \psi$ and $\forall l' < l_\lambda : \lambda[l'] \models \varphi$.

Let us construct a simpler strategy F'_A based on F_A as follows: for an arbitrary $\lambda \in out(s, F_A)$ such that $\exists l_1 < l_2 \leq l_\lambda : \lambda[l_1] = \lambda[l_2] : F'_A(\lambda[0,l_1]\lambda') = F_A(\lambda[0,l_2]\lambda')$ for any $\lambda' \in S^*$; for others λ'', $F'_A(\lambda'') = F_A(\lambda)$. We repeat this construction until no further simplifications can be made. Let F''_A be the resulting strategy. We have that for all $\lambda \in out(s, F''_A)$: $\exists l_\lambda \geq 0 : \lambda[l_\lambda] \models \psi$ and $\forall l' < l_\lambda : \lambda[l'] \models \varphi$. Furthermore, no state is repeated along any $\lambda[0, l_\lambda]$ where $\lambda \in out(s, F''_A)$. This means $l_\lambda < |S|$. Furthermore, $cost_A(\lambda[i], F''_A(\lambda[0,i])) \leq maxc_A$ for all $i \leq l_\lambda$; therefore, the total cost along $\lambda[0, l_\lambda]$ is less than $maxc_A \times |S|$; i.e., F''_A is a $(maxc_A \times |S|)$-strategy; hence $s \in \langle\langle A^{maxc_A \times |S|}\rangle\rangle\varphi\,\mathcal{U}\,\psi$. □

4.2 Box Formulas

In this section, we show that $[\langle\langle A^b\rangle\rangle\Box\varphi]_M$ can be computed from the greatest fixed point of a function $B_{A,\varphi}$ defined below.

First, we define a function (dependent on the bound b)

$$B_{A^b,\varphi}((X_0, \ldots, X_{b-minc_A})) = [\varphi] \cap \left(\bigcup_{minc_A \leq b_1 \leq b} [\langle\langle b_1\rangle\rangle\bigcirc](X_{b-b_1}) \right)$$

We shall drop the subscript φ for convenience. Intuitively, $B_{A^b}(([\langle\langle A^0\rangle\rangle\Box\varphi], \ldots, [\langle\langle A^{b-minc_A}\rangle\rangle\Box\varphi])) = [\langle\langle A^b\rangle\rangle\Box\varphi]$. Then, we have the following result:

Lemma 7. *B_{A^b} is monotone.*

Proof. The proof is similar to that of Lemma 2. □

Lemma 8. *There exists a bound $\min \in \mathbb{N}$ such that $[\langle\langle A^{\min}\rangle\rangle\Box\varphi] \supseteq [\langle\langle A^k\rangle\rangle\Box\varphi]$ for any bound k.*

Proof. The proof is similar to that of Lemma 3. □

Again, the result of Lemma 8 means that $[\langle\langle A^{\min}\rangle\rangle\Box\varphi] = [\langle\langle A^k\rangle\rangle\Box\varphi]$ for $k \geq k_{\min}$. As a result, we define the following function, which is similar to B_{A^k} but accepts input vectors of the same size $min + 1$ for any $k \geq 0$:

$$\hat{B}_{A^k,\varphi}((X_0, \ldots, X_{\min})) = ([\varphi] \cap \left(\bigcup_{minc_A \leq k_1 \leq k} [\langle\langle A^{k_1}\rangle\rangle\bigcirc]\left(\begin{cases} X_{k-k_1} & \text{if } k - k_1 \leq \min \\ X_{\min} & \text{otherwise} \end{cases} \right)\right)$$

This function is also monotone.

Finally, we define the following function:

$$B_{A,\varphi}((X_0, \ldots, X_{\min})) = (\hat{B}_{A^k}((X_0, \ldots, X_{\min})))_{0 \leq k \leq \min}$$

Lemma 9. *B_A is monotone.*

Lemma 10. *$([\langle\langle A^0\rangle\rangle\Box\varphi], \ldots, [\langle\langle A^{\min}\rangle\rangle\Box\varphi])$ is the greatest fixed point of B_A.*

Proof. The proof is similar to that of Lemma 5. □

As usual, the greatest fixed point $(X_k)_{0 \leq k \leq \min}$ of B_A can be computed by repeatedly applying B_A to $(S, \ldots, S)$ until a fixed point is reached.

Then,

$$[\langle\langle A^b\rangle\rangle\Box\varphi] = \begin{cases} X_b & \text{if } b \leq \min \\ X_{\min} & \text{otherwise.} \end{cases}$$

It remains to find $\min$. Again, a coarse estimation is that $\min = maxc_A \times |S|$. The idea behind this estimation is that any strategy to satisfy $\langle\langle A^{\min}\rangle\rangle\Box\varphi$ will eventually maintain a zero-cost cycle. Then, it can be simplified so that the prefix leading to the zero-cost cycle does not repeat a state.

Lemma 11. *If $s \in \langle\langle A^k\rangle\rangle\Box\varphi$ for some $k \in \mathbb{N}$ then $s \in \langle\langle A^{maxc_A \times |S|}\rangle\rangle\Box\varphi$.*

Proof. The proof is similar to that of Lemma 3. □

4.3 Symbolic Algorithm for 1RB±ATL

The above results enable us to provide the symbolic model checking algorithm for 1RB±ATL shown in Algorithm 2.

In Algorithm 2, $\bar{\cdot}$ denotes a vector of size $maxc_A \times |S| + 1$ where $\bar{\cdot}_i$ denotes the i-th element of $\bar{\cdot}$ for $i \geq 0$. The algorithm differs from Algorithm 1 only for the cases of until formulas $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$ and box formulas $\langle\langle A^b\rangle\rangle\Box\varphi$. For until formulas $\langle\langle A^b\rangle\rangle\varphi\,\mathcal{U}\,\psi$, the algorithm computes the least fixed point of $U_{A,\varphi,\psi}$. It also makes use of the maximal cost $maxc_A$ of any action by agents in A which can be computed in advance.

Algorithm 2 Labelling φ_0

```
function 1RB±ATL-LABEL($M, \varphi_0$)
    for $\varphi' \in Sub(\varphi_0)$ do
        case $\varphi' = p,\ \neg\varphi,\ \varphi \wedge \psi,\ \langle\langle A\rangle\rangle \bigcirc \varphi,$
                $\langle\langle A^b\rangle\rangle \bigcirc \varphi,\ \langle\langle A\rangle\rangle \varphi \mathcal{U} \psi,\ \langle\langle A\rangle\rangle \Box \varphi$
            see Algorithm 1
        case $\varphi' = \langle\langle A^b\rangle\rangle \varphi \mathcal{U} \psi$
            $\bar{\rho} \leftarrow \bar{\emptyset}$ and $\bar{\tau} \leftarrow \overline{[\psi]_M}$
            while $\bar{\tau} \not\subseteq \bar{\rho}$ do
                $\bar{\rho} \leftarrow \bar{\rho} \cup \bar{\tau}$ and $\bar{\tau} \leftarrow U_{A,\varphi,\psi}(\bar{\rho})$
            if $b < maxc_A \times |S|$ then
                $[\varphi']_M \leftarrow \bar{\rho}_b$
            else
                $[\varphi']_M \leftarrow \bar{\rho}_{maxc_A \times |S|}$
        case $\varphi' = \langle\langle A^b\rangle\rangle \Box \varphi$
            $\bar{\rho} \leftarrow \bar{S}$ and $\bar{\tau} \leftarrow \overline{[\varphi]_M}$
            while $\bar{\rho} \not\subseteq \bar{\tau}$ do
                $\bar{\rho} \leftarrow \bar{\tau}$ and $\bar{\tau} \leftarrow B_{A,\varphi}(\bar{\rho})$
            if $b < maxc_A \times |S|$ then
                $[\varphi']_M \leftarrow \bar{\rho}_b$
            else
                $[\varphi']_M \leftarrow \bar{\rho}_{maxc_A \times |S|}$
    return $[\varphi_0]_M$
```

Both are defined in Section 4.1. For box formulas $\langle\langle A^b\rangle\rangle \Box \varphi$, the algorithm computes the greatest fixed point of $B_{A,\varphi}$ as defined in Section 4.2. The correctness of these two cases is due to Lemmas 5, 6, 10 and 11. Termination is guaranteed as the set S of states in M is finite.

Theorem 1. *Algorithm 2 runs in time* $O(|Act|^{|Agt|} \times |Agt|^3 \times |S|^4 \times (-m_p) \times m_c^2)$ *where* m_p *is the least cost of any action and* m_c *is the greatest cost of any action.*

Proof. We only discuss the three RB±ATL modalities.

First, observe that the function $Pre(A, [\varphi]_M, b)$ used in the computation of $[\langle\langle A^b\rangle\rangle \bigcirc \varphi]_M$ case involves evaluating a boolean expression which encodes all possible joint actions of A and their costs (one boolean per cost). The size of this expression is $O(|Act|^{|Agt|} \times |Agt|)$. Hence the operation takes time $O(|Act|^{|Agt|} \times |Agt|)$. For readability, we abbreviate $|Act|^{|Agt|} \times |Agt|$ as T_{Pre}.

For the next two cases, we will use the following abbreviations. $M_c = |Agt| \times m_c$ is the maximal cost of any joint action. $M_c \times |S|$ is an upper bound on $maxc_A \times |S|$ used in the algorithm. $M_p = |Agt| \times -m_p$ is the maximal production of resource by any joint action.

For Until formulas, the loop is executed at most $|S|$ times. During each iteration, $U_{A,\varphi,\psi}(\bar{\rho})$ is computed. Since $U_{A,\varphi,\psi}(\bar{\rho}) = (\hat{U}_{A^0}(\bar{\rho}), \ldots, \hat{U}_{A^{(M_c \times |S|)}}(\bar{\rho}))$, and the computation of $\hat{U}_{A^{(M_c \times |S|)}}(\bar{\rho})$ is the most expensive, let us compute the time required for that and multiply by $M_c \times |S|$.

$\hat{U}_{A^{(M_c \times |S|)}}(\bar{\rho}) = [\psi] \cup ([\varphi] \cap (Z_1 \cup \ldots \cup Z_{M_p + (M_c \times |S|)}))$, where each Z_j is of the form $[\langle\langle A^{k_j}\rangle\rangle \bigcirc](\bar{\rho}_j)$, so computing it takes time $O(T_{Pre})$. Then the computation of $\hat{U}_{A^{(M_c \times |S|)}}(\bar{\rho})$ is $O(T_{Pre} \times (M_p + (M_c \times |S|)))$ and the computation of $U_{A,\varphi,\psi}(\bar{\rho})$ is $O(T_{Pre} \times (M_p + (M_c \times |S|)) \times (M_c \times |S|))$. Finally, the loop for until formulas takes time

$$O(T_{Pre} \times (M_p + (M_c \times |S|)) \times (M_c \times |S|) \times |S|)$$

A similar analysis shows that the running time of the Box loop is the same, hence so is the running time of the whole algorithm. Substituting expressions for T_{Pre}, M_c and M_p gives $O(|Act|^{|Agt|} \times |Agt|^3 \times |S|^4 \times (-m_p) \times m_c^2)$. □

Note that the $|Act|^{|Agt|}$ component is the cost of computing Pre symbolically, and it corresponds to the number of all possible joint actions in the model. If the set of joint actions is considered to be part of the input, the algorithm runs in time polynomial in the size of the model.

5 Experimental Evaluation of Performance

In this section, we describe an experiment designed to show the scalability of the symbolic implementation of the model-checking algorithm for 1RB±ATL, and compare its performance with that of the hybrid model-checking algorithm for RB±ATL.

Consider a scenario in which three agents compete in an infinite sequence of car races by spending and earning money. Each car is equipped with one of three possible types of engine $T = \{1, 2, 3\}$. Initially, agent 1 has an engine of type 1, agent 2 of type 2, and agent 3 of type 3. Each race consists of 3 steps:

Set-up: Agents can choose whether to upgrade their engine; upgrading from engine type i to type $i+1$ costs \$1 ($i \leq |T| - 1$).

Race: Agents can choose to race (at a cost of \$1), or to remain idle (at a cost of \$0). The ranking of the race is decided as follows: idle agents have rank 4th. The other agents are partitioned in disjoint sets of participants with the same engine type. The agents in the set with engine type 3 have rank 1st; the agents in the next partition have a rank equal to the number of agents in the previous set plus the rank of those agents, and so on.

Award: Agents can execute an action to accept their ranking, or perform an idle action. The action of accepting their ranking produces \$3 if the agent ranks 1st, \$2 if they rank 2nd, and \$1 for 3rd.

The race is then repeated.

If all agents decide to race starting from the configuration above, agent 3 will win and produce \$3, agent 2 will produce \$2, while agent 1 will produce \$1. It is possible to verify that no agent can participate in the race with a bound of \$0. It is also possible to verify that agent 1 cannot be ranked 1st with a bound of \$2, but it can if the bound is \$3 (because it can decide to upgrade but not race in the first two rounds, and only race in the third one). More interestingly, it is possible to verify that if agents 1 and 2 cooperate, then they can *both* be ranked 1st under a bound \$1. To achieve this, one of the two agents has to idle (not race), until the other agent has accumulated enough money to upgrade its engine; when both agents have upgraded their engines they can both win the race.

This scenario can be scaled up to any number of agents and any number of engine types. We have implemented Algorithm 2 on top of the model checker MCMAS [Lomuscio *et al.*, 2009] and we have encoded examples in which the number of engine types $|T| \in \{3, 4, 5, 6\}$[1].

In the experiment, we verify, for each encoding with T types of engines, whether agent 1 can eventually win the race within the bounds $b \in \{|T| - 1, |T|\}$ and the coalition of agents 1 and 2 can cooperate so that eventually agent 1 wins the race. In other words, we want to check if the formulas $\varphi_1(b) = \langle\langle\{1\}^b\rangle\rangle \top \mathcal{U}\, 1\text{fst}$ for $b \in \{|T| - 1, |T|\}$ and $\varphi_2 = \langle\langle\{1, 2\}^1\rangle\rangle \top \mathcal{U}\, 1\text{fst}$ where 1fst is a proposition that is only true in states where agent 1 ranks first. The experiment was carried out on a quad-core 64-bit Processor running at 2.66 GHz with 32GB of memory. The results are summarised in Table 1 (Holds records whether the formula is true in the model). As it can be seen, in this example the symbolic algorithm always uses less memory. It also outperforms the hybrid algorithm when verifying φ_1 (from 2 to 1961 times faster) and catches up when verifying φ_2 (eventually 3 times faster).

Types	Fomulas	Holds	Time (s)			Memory (MB)		
			Hybrid	Symbolic	Ratio	Hybrid	Symbolic	Ratio
3	$\varphi_1(2)$	No	0.509	0.175	2.91	10.23	9.09	1.12
	$\varphi_1(3)$	Yes	0.595	0.175	3.40	10.23	9.09	1.12
	φ_2	Yes	0.210	0.745	0.28	10.01	9.16	1.09
4	$\varphi_1(3)$	No	37.300	0.826	45.16	11.40	9.31	1.22
	$\varphi_1(4)$	Yes	64.252	0.821	78.26	11.40	9.31	1.22
	φ_2	Yes	1.649	5.146	0.32	10.82	9.51	1.14
5	$\varphi_1(4)$	No	4582.360	3.543	1293.36	14.19	10.22	1.39
	$\varphi_1(5)$	Yes	6792.980	3.463	1961.59	15.60	10.22	1.53
	φ_2	Yes	14.678	19.154	0.77	13.52	10.33	1.31
6	$\varphi_1(5)$	No	time-out	6.233	-	time-out	10.15	-
	$\varphi_1(6)$	Yes	time-out	6.182	-	time-out	10.15	-
	φ_2	Yes	115.370	30.013	3.84	14.57	10.23	1.42

Table 1: Comparison between symbolic and hybrid implementations. Time-out means no termination within 3 hours.

6 Conclusion and future work

We have presented a fixed point characterisation of coalition modalities in the resource logic 1RB±ATL. This enabled us to state and implement a symbolic model-checking algorithm for 1RB±ATL. We gave an evaluation of the performance of the algorithm on a parameterised example, which shows that the symbolic algorithm is more scalable than the hybrid one.

We conjecture that full RB±ATL does not have a fixed point characterisation. In future work, we plan to settle this question and investigate other resource logics (or their fragments) with decidable model-checking problems.

References

[Alechina *et al.*, 2009] N. Alechina, B. Logan, H. N. Nguyen, and A. Rakib. A logic for coalitions with bounded resources. In *Proc. of the 21st International Joint Conference on Artificial Intelligence*, volume 2, pages 659–664. AAAI Press, 2009.

[Alechina *et al.*, 2010] N. Alechina, B. Logan, H. N. Nguyen, and A. Rakib. Resource-bounded alternating-time temporal logic. In *Proc. of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 481–488. IFAAMAS, 2010.

[Alechina *et al.*, 2014] N. Alechina, B. Logan, H. N. Nguyen, and F. Raimondi. Decidable Model-Checking for a Resource Logic with Production of Resources. In *21st European Conference on Artificial Intelligence*, pages 9–14, 2014.

[Alechina *et al.*, 2015] N. Alechina, B. Logan, H. N. Nguyen, F. Raimondi, and L. Mostarda. Symbolic model-checking for resource-bounded ATL. In *Proc. of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*, pages 1809–1810. IFAAMAS, 2015.

[Alur *et al.*, 2002] R. Alur, T. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[Bryant, 1986] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Computers*, 35(8):677–691, 1986.

[Bulling and Farwer, 2010] N. Bulling and B. Farwer. On the (un-)decidability of model checking resource-bounded agents. In *Proc. of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, volume 215, pages 567–572. IOS Press, 2010.

[Bulling and Goranko, 2013] 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 *Proc. 1st International Workshop on Strategic Reasoning, SR 2013*, volume 112 of *EPTCS*, pages 33–41, 2013.

[Della Monica *et al.*, 2011] D. Della Monica, M. Napoli, and M. Parente. On a logic for coalitional games with priced-resource agents. *Electr. Notes Theor. Comput. Sci.*, 278:215–228, 2011.

[Della Monica *et al.*, 2013] D. Della Monica, M. Napoli, and M. Parente. Model checking coalitional games in shortage resource scenarios. In *Proc. of the 4th International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2013*, volume 119 of *EPTCS*, pages 240–255, 2013.

[Gammie and Van Der Meyden, 2004] P. Gammie and R. Van Der Meyden. MCK: Model checking the logic of knowledge. In *Computer Aided Verification*, pages 479–483. Springer Berlin Heidelberg, 2004.

[Leroux, 2013] J. Leroux. Acceleration for Petri Nets. In *Proc. of the 11th International Symposium on Automated Technology for Verification and Analysis, ATVA 2013*, volume 8172 of *LNCS*, pages 1–4. Springer, 2013.

[Lomuscio *et al.*, 2009] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: A model checker for the verification of multi-agent systems. In *Proc. of the 21st International Conference on Computer Aided Verification*, CAV '09, pages 682–688, 2009.

[1]Available from `www.vrbmas.org/dl/ijcai15.zip`