Symbolic Model-checking for Resource-Bounded ATL

(Extended Abstract)

Natasha Alechina, Brian Logan, Hoang Nga Nguyen
School of Computer Science
University of Nottingham
{nza,bsl,hhn}@cs.nott.ac.uk

Franco Raimondi
Department of Computer Science
Middlesex University
f.raimondi@mdx.ac.uk

Leonardo Mostarda
Scuola di Scienze e Tecnologie
Universitá degli Studi di Camerino
leonardo.mostarda@unicam.it

ABSTRACT

In this paper we present a symbolic implementation of a model checking algorithm for the verification of properties expressed in Resource-Bounded Alternating Time Temporal Logic (RB-ATL). The implementation is based on the model checker MCMAS. We evaluate the performance of our implementation using simple multi-agent model checking problems of increasing complexity.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multi-agent systems*

Keywords

Model-checking; resources

1. RESOURCE-BOUNDED ATL

Resource availability is an important characteristic of many multi-agent systems, since agents require resources in order to execute actions. For example, the nodes in a sensor network require energy to exchange messages. A number of *resource logics* for reasoning about and verifying the resource requirements of MAS have been proposed in the literature. However, the computational complexity of the model-checking problem for these logics is typically very high, and some are undecidable. In this paper, we focus on Resource-Bounded ATL (RB-ATL) [1], as the model checking problem for RB-ATL is polynomial in the size of the formula (if resource bounds in the formula are encoded in unary) and the model (if the number of resource types is treated as a constant). A model checking algorithm for RB-ATL was given in [1]. The key contribution of this paper is a symbolic encoding of the algorithm given in [1] which employs boolean variables to encode states, action costs, and resource bounds. We have implemented our symbolic algorithm by extending the ATL model checking capabilities of the MCMAS model checker [2], and present preliminary results of experiments comparing the performance of our symbolic implementation with a hybrid algorithm.

We generalise the version of RB-ATL presented in [1] by extending the language with infinite resource bounds that allow us to express standard ATL modalities and to ignore restrictions on some resource types.

Let $Agt = \{a_1, \ldots, a_n\}$ be a set of n agents, $Res = \{res_1, \ldots, res_r\}$ be a set of r resource types, Π denote a set of propositions and $B = \mathbb{N}^r_\infty$ denote the set of resource bounds where $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$. Formulas of RB-ATL are then 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. $\langle\langle A^b \rangle\rangle \bigcirc \varphi$ 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, and $\langle\langle A^b \rangle\rangle \varphi \,\mathcal{U}\, \psi$ 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.

RB-ATL is interpreted on concurrent game structures extended with resource requirements for executing actions $M = (Agt, Res, S, \Pi, \pi, Act, d, c, \delta)$ where $\pi : \Pi \to \wp(S)$ is a truth assignment, Act is a non-empty set of actions which includes a special action $idle$, $d : S \times Agt \to \wp(Act) \setminus \{\emptyset\}$ is a function that assigns to each $s \in S$ a non-empty set of actions available to each agent $a \in Agt$, $c : S \times Agt \times Act \to \mathbb{N}^r$ is a partial function that maps a state s, agent a and an action $\alpha \in d(s, a)$ to a vector of integers, where a positive integer in position i indicates consumption of resource res_i by the action, and $\delta : (s, \sigma) \mapsto S$ 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. We stipulate that $c(s, a, idle) = 0^r$ for all $s \in S$ and $a \in Agt$ ($idle$ does not consume resources). See [1] for details of the truth definitions. The presence of $idle$ simplifies the model-checking problem, since any finite strategy has a zero-cost infinite extension where all agents chose $idle$. It also means that the coalition monotonicity property holds: if A can enforce property ϕ under resource bound b, then a larger coalition B ($A \subseteq B$) can enforce ϕ under the same resource bound (intuitively, agents in $B \setminus A$ can chose the $idle$ action).

2. SYMBOLIC ENCODING

In this section we present a symbolic encoding of the model checking algorithm for RB-ATL given in [1]. In particular, we show how the function $Pre^b(A, \rho)$ in [1], which returns the set of states from which the coalition A can enforce ρ in one step without spending more than b amount of resource, can be implemented symbolically using additional boolean variables to encode costs and bounds.

Given an agent $i \in Agt$ and a resource $r \in Res$, let $C_{r,i} = \{c_r(a_i) \mid a_i \in Act_i\}$ denote the costs of actions by i with respect to r. We therefore need $\sum_{r \in Res} |C_{r,i}|$ variables to encode

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

the costs and bounds of an action by i. We encode the costs of actions by all the agents in Agt so that the same encoding can be used to verify formulas referring to arbitrary coalitions. Let $C_i = \bigcup_{r \in Res} \{c_v^{r,i} \mid v \in C_{r,i}\}$ denote the set of such variables. Given a cost $c \in \mathbb{N}^{|Res|}$ of an action by agent i, i.e., $c_r \in C_{r,i}$ for all $r \in Res$, c_r is encoded as $f_{c_r}^{r,i} = \bigwedge_{v \in C_{r,i}} d_v^{r,i}$ where

$$d_v^{r,i} = \begin{cases} c_v^{r,i} & \text{if } v = c_r \\ \neg c_v^{r,i} & \text{otherwise.} \end{cases}$$

Then, c is encoded as $f_c^i(C_i) = \bigwedge_{r \in Res} f_{c_r}^{r,i}$. Similarly, the bound $b \in (\mathbb{N} \cup \{\infty\})^{|Res|}$ for an agent i, is encoded as $h_b^i(C_i) = \bigwedge_{r \in Res} \bigvee_{e \in C_{r,i}, e \leq b_r} c_e^{r,i}$. Then, we have $f_b^i(C_i) \wedge h_{b'}^i(C_i) = \bot$ if $b \leq b'$ and $f_b^i(C_i)$ otherwise. In general, a bound b for a coalition of agents A is encoded by:

$$h_b^A(\cup_{i \in A} C_i) = \bigvee_{(b_i)_{i \in A} \in \prod_{i \in A} C_i, \sum_{i \in A} b_i \leq b} \bigwedge_{i \in A} h_{b_i}$$

We extend the encoding of actions to include their costs. An action a_i with cost c performed by agent i is encoded by $f_{a_i,c}(A_i, C_i) = f_{a_i}(A_i) \wedge f_c^i(C_i)$. The encoding of d and o are also extended with binary variables for costs as $prot(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i)$ and $evol(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i, V')$, respectively.

$Pre^b(A, \rho)$ can then be obtained by extending the implementation of $Pre(A, \rho)$ as follows:

- $\neg\rho(V')$ encodes the set of next states that are not in ρ;
- $p_1(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i) = \exists V' : \neg\rho(V') \wedge evol(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i, V')$ encodes the set of states from which there is a transition to states not in ρ;
- $p_2(V, \cup_{i \in A} A_i) = \exists \cup_{i \in Agt \setminus A} A_i, \cup_{i \in Agt} C_i : p_1(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i) \wedge prot(V, \cup_{i \in Agt} A_i, \cup_{i \in Agt} C_i)$ encodes states and joint actions from these states by A such that $Agt \setminus A$ can prevent the next states from being in ρ;
- $p_3(V, \cup_{i \in A} A_i) = \neg p_2(V, \cup_{i \in A} A_i)$ encodes either states or joint actions by A such that $Agt \setminus A$ cannot prevent the next states from being in ρ;
- $p_4(V, \cup_{i \in A} A_i) = \exists \cup_{i \in Agt \setminus A} A_i, \cup_{i \in Agt} C_i : p_3(V, \cup_{i \in A} A_i) \wedge prot(V, \cup_{i \in Agt} A_i) \wedge h_b^A(\cup_{i \in A} C_i)$ encodes states and joint actions which cost at most b from these states by A such that $Agt \setminus A$ cannot prevent the next states from being in ρ;
- $p_5(V) = \exists \cup_{i \in A} A_i : p_4(V, \cup_{i \in A} A_i)$ encodes states from which A has a joint action which costs at most b such that $Agt \setminus A$ cannot prevent the next states from being in ρ.

Notice that $p_4(V, \cup_{i \in A} A_i)$ includes the encoding of the bound b and states in $Pre^b(A, \rho)$ are encoded by $p_5(V)$.

3. EXPERIMENTAL EVALUATION

In this section, we compare the performance of a MCMAS-based implementation of the symbolic encoding with a hybrid implementation of RB-ATL model-checking. In the hybrid implementation, the underlying ATL model-checking is symbolic, but the function $Pre^b(A, \rho)$ is implemented by explicitly collecting all states which have a joint action σ_A with $c(\sigma_A) \leq b$ and all possible outcomes in ρ. In the experimental scenario, there is an array of spaces on which a block can be placed, and a number of arms (each controlled by an agent) that can travel in a straight line over these spaces. There are two resources: power to run the arms and the monetary cost of wear and tear. The actions and their costs on each resource are as follows:

Action	Cost	Description
move left	(2,1)	move to an empty space on the left
move right	(2,1)	move to an empty space on the right
pick up	(3,2)	pick a block from the space below
drop	(1,3)	drop a block to the empty space below
idle	(0,0)	do nothing

We assume that: the agents take turns to perform actions, there are at least two spaces, the number of agents is smaller than the number of spaces, and the number of blocks is at most half of the number of spaces.

We verify whether, starting in a state where all blocks are on the leftmost spaces, the agents can cooperate to move all blocks to the rightmost spaces within three different resource bounds $b_1 = (8, 7)$, $b_2 = (12, 18)$ and $b_3 = (24, 24)$. Specifically, we check if the formulas $\varphi_i = \langle\langle Agt^{b_i} \rangle\rangle \top \mathcal{U}\, dest$ for $i \in \{1, 2, 3\}$ are true in the initial state, where $dest$ is a propositional variable that is true only in states where all the blocks are on the rightmost spaces. The symbolic implementation requires $8 \times |Agt|$ additional boolean variables to encode costs and bounds compared to the hybrid implementation. The experiments were performed out on a 2.66 GHz quad-core 64-bit processor with 32GB of memory. The results are summarised in Table 1. For each instance, we show the truth of φ_i, the CPU time required by the symbolic and hybrid implementations, and the ratio of hybrid to symbolic times. As can be seen, in this scenario, the symbolic implementation outperforms the hybrid implementation in all cases, and is 3 times to 60 times faster depending on the problem.

Scenario			Formula	t?	Time (s)		Ratio
Spaces	$\|Agt\|$	Blocks			Symbolic	Hybrid	
2	1	1	φ_1	t	0.077	0.441	5.73
			φ_2	t	0.786	5.164	6.57
			φ_3	t	4.728	32.973	6.97
3	1	1	φ_1	f	0.099	0.764	7.72
			φ_2	t	1.039	9.911	9.54
			φ_3	t	6.199	67.695	10.92
3	2	1	φ_1	f	0.552	2.279	4.13
			φ_2	f	7.865	27.513	3.50
			φ_3	t	57.869	194.015	3.35
4	1	1	φ_1	f	0.101	0.996	9.86
			φ_2	f	1.080	13.923	12.89
			φ_3	t	6.669	101.812	15.27
4	1	2	φ_1	f	0.191	3.375	17.67
			φ_2	f	2.080	48.356	23.25
			φ_3	t	12.678	391.274	30.86
4	2	1	φ_1	f	0.616	4.488	7.29
			φ_2	f	10.489	60.038	5.72
			φ_3	t	80.596	453.046	5.62
4	2	2	φ_1	f	0.887	28.075	31.65
			φ_2	f	11.433	580.737	50.79
			φ_3	t	84.539	5859.010	60.31

Table 1: Experimental results

REFERENCES

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

[2] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: A model checker for the verification of multi-agent systems. In *Proceedings of the 21st International Conference on Computer Aided Verification, CAV 2009*, *LNCS* Volume 5643, pages 682–688. Springer, 2009.