Verifying Existence of Resource-Bounded Coalition Uniform Strategies

Natasha Alechina  
University of Nottingham  
nza@cs.nott.ac.uk

Mehdi Dastani  
University of Utrecht  
m.m.dastani@uu.nl

Brian Logan  
University of Nottingham  
bsl@cs.nott.ac.uk

Abstract

We consider the problem of whether a coalition of agents has a knowledge-based strategy to ensure some outcome under a resource bound. We extend previous work on verification of multi-agent systems, where actions of agents produce and consume resources, by adding epistemic pre- and postconditions to actions. This allows us to model scenarios where agents perform both actions which change the world, and actions which change their knowledge about the world, such as observation and communication. To avoid logical omniscience and obtain a compact model of the system, our model of agents’ knowledge is syntactic. We define a class of coalition-uniform strategies with respect to any (decidable) notion of coalition knowledge. We show that the model-checking problem for the resulting logic is decidable for any notion of coalition-uniform strategies in these classes.

1 Introduction

We propose a new logical formalism, RB±ATSEL, for modelling and verifying multi-agent systems where agents execute both ontic actions (actions that change the world) and epistemic actions (actions that change their knowledge). This is a common situation in many multi-agent systems where agents have to explore and change their environment; for example, knowledge-based planning, diagnosis, etc. As an example, we focus on multi-agent systems where some agents monitor the behaviour of other agents to detect norm violations [Álvarez-Napagao et al., 2011]. We would like to be able to automatically verify properties of such systems using model-checking; for example, to check whether monitoring agents have a strategy to detect all norm violations.

There has been considerable work on Alternating Time Temporal Logic (ATL) extended with epistemic operators and on the model-checking problem for the resulting logics, e.g., [van der Hoek and Wooldridge, 2002; Lomuscio et al., 2009]. The motivation of this paper is closer to the work on Dynamic Epistemic Logic (DEL) e.g., [Baltag et al., 1998; van Ditmarsch and Kooi, 2008], and epistemic planning, e.g., [Andersen et al., 2012], where we can reason about how epistemic actions change the agents’ epistemic states, which is impossible in epistemic ATL.

Our approach differs from previous work in two main respects: the adoption of syntactic knowledge, and considering costs of both ontic and epistemic actions. We interpret epistemic modalities syntactically rather than using an indistinguishability relation. This allows us to use simpler models, and to model different (non-omniscient) reasoning procedures for different agents. We also consider the costs of both ontic and epistemic actions, such as observation and communication. Clearly, ontic actions (e.g., moving from one location to another) have costs (e.g., energy). However, observations often have non-trivial costs (e.g., an agent may need to use costly equipment, or pay some authority for verified information [Jamroga and Tabatabaei, 2013; Naumov and Tao, 2015]). Exchanging messages also has costs, for example, energy, or money. This is particularly relevant for norm monitoring scenarios: a successful monitoring strategy may exist, but could be prohibitively expensive and not practically feasible. For this reason, we chose as the basis for our formalism the logic RB±ATL, where actions produce and consume resources [Alechina et al., 2014]. (If observations have a cost, we need to model resource production if monitoring is to be performed indefinitely.) Using RB±ATL allows us to check whether a strategy that requires less than a given amount of resources exists. However Alechina et al. [2014] consider only the resource consumption of ontic actions.

The notion of strategies we consider are perfect recall strategies, where the choice of the next action by an agent depends on all previously encountered states. Perfect recall strategies make more sense than memoryless strategies in our setting, as actions both produce and consume resources. (Intuitively, this is because an agent may need to ‘loop’ several times making some resource in order to execute an action that consumes the resource, e.g., recharging a battery for several timesteps.) In addition, strategies should also be uniform; that is, if an agent has the same knowledge at each point in two histories, then it should choose the same action in both of them. However, model-checking epistemic ATL with uniform perfect recall strategies and more than one agent is undecidable [Dima and Tiplea, 2011]. This result does not change for syntactic epistemics. We therefore propose a notion of coalition-uniform strategies for which the model-checking problem is decidable. A strategy is coalition-uniform for a coalition $A$ if...
for any two histories indistinguishable for $A$ (wrt some notion of indistinguishability), it chooses the same action. We call the resulting logic Resource-Bounded Alternating-Time Syntactic Epistemic Logic (RB±ATSEL). The main contribution of this paper is a decidable model-checking procedure for RB±ATSEL with coalition-uniform strategies (wrt any decidable notion of indistinguishability).

2 Syntax and Semantics of RB±ATSEL

We adopt the approach to epistemic logic that interprets agents’ knowledge syntactically, as a (finite) set of formulas, as in, e.g., [Konolige, 1986]. An agent knows that $\phi$ if, and only if, $\phi$ is in its knowledge base or is derivable from it by some simple terminating procedure (e.g., closure under modus ponens). This approach is very close to the notion of algorithmic knowledge of Fagin et al. [1995]. In what follows, to decide whether the agent knows $\phi$, we simply check whether a formula $\phi$ is in agent $i$’s state $s_i$, but this can be trivially replaced with a check for $\mathit{alg}(s_i, \phi) = \text{true}$, where $\mathit{alg}$ is a terminating procedure that takes a set of formulas $s_i$ and a formula $\phi$ and checks whether $\phi$ follows from $s_i$.

Syntactic knowledge provides a convenient and compact way of modelling knowledge change, compared to, for example, DEL. In DEL, the update mechanism involves combining models to produce new models, and requires considerably more space to represent and more computation to reason about. In the syntactic approach, we can simply specify post-conditions of actions which add and remove formulas from the agent’s state. In DEL, we need to essentially associate an automaton with each action that can transform an epistemic model into a new epistemic model. Finally, it is worth noting that many epistemic planners use what are essentially syntactic knowledge bases (and as a result solve a decidable planning problem), e.g., [Petrick and Bacchus, 2004]. This contrasts with the undecidability of DEL-based epistemic planning [Aucher and Bolander, 2013].

The language of RB±ATSEL is built from the following components: $\mathsf{Agt} = \{a_1, \ldots, a_n\}$ a set of $n$ agents; $\mathsf{Res} = \{r_{s_1}, \ldots, r_{s_r}\}$ a set of $r$ resources; and $\Pi$ a set of propositions. $B = \mathsf{Agt} \times \mathsf{Res} \rightarrow \mathbb{N}$ is a set of resource bounds, where $\mathbb{N} = \mathbb{N} \cup \{\infty\}$. (Note that the definition of bound and related definitions differ from those in [Aucher et al., 2014] as we assume resources can’t be transferred between agents.)

Formulas of the language $\mathcal{L}$ of RB±ATSEL are defined by the following syntax $\varphi, \psi ::= p \mid \neg \varphi \mid \varphi \lor \psi \mid \langle\langle A \rangle\rangle \varphi \mid \langle\langle A \rangle\rangle \varphi \psi \mid \langle\langle A \rangle\rangle \square \varphi \mid K_a \varphi$ where $p \in \Pi$ is a proposition, $A \subseteq \mathsf{Agt}$, $b \in B$ is a resource bound and $a \in \mathsf{Agt}$.

The meaning of RB±ATSEL formulas is as follows: $\langle\langle A \rangle\rangle \varphi \psi$ means that a coalition $A$ has a strategy executable within resource bound $b$ to ensure that the next state satisfies $\varphi$; $\langle\langle A \rangle\rangle \varphi \psi$ means that $A$ has a strategy executable within resource bound $b$ to ensure $\psi$ while maintaining the truth of $\varphi$; $\langle\langle A \rangle\rangle \square \varphi$ means that $A$ has a strategy executable within resource bound $b$ to ensure that $\varphi$ is always true; and $K_a \varphi$ means that formula $\varphi$ is in agent $a$’s knowledge base.

**Definition 1.** A model of RB±ATSEL is a structure $M = (\Phi, \mathsf{Agt}, \mathsf{Res}, S, \Pi, \varphi, d, c, \delta)$ where:

- $\Phi$ is a finite set of formulas of $\mathcal{L}$.
- $S$ is a set of tuples $(s_1, \ldots, s_n, s_c)$ where $s_c \subseteq \Pi$ and for each $a \in \mathsf{Agt}$, $s_a \subseteq \Phi$.
- $\mathsf{Agt}$ is a non-empty set of $n$ agents, $\mathsf{Res}$ is a non-empty set of $r$ resources.
- $\Pi$ is a finite set of propositional variables; $p \in \Pi$ is true in $s \in S$ iff $p \in s_c$.
- $\delta$ is a non-empty set of actions which includes idle, and $d : S \times \mathsf{Agt} \rightarrow \phi(\mathcal{Act}) \setminus \{\emptyset\}$ is a function which assigns to each $s \in S$ a non-empty set of actions available to each agent $a \in \mathsf{Agt}$. We assume that for every $s \in S$ and $a \in \mathsf{Agt}$, idle $\in \mathit{d}(s,a)$. We denote joint actions by all agents in $\mathsf{Agt}$ available at $s$ by $D(s) = \mathit{d}(s,a_1) \times \cdots \times \mathit{d}(s,a_n)$.
- for every $s, s' \in S, a \in \mathsf{Agt}, d(s,a) = d(s',a)$ if $s_a = s'_a$.
- $c : \mathcal{Act} \times \mathsf{Res} \rightarrow \mathbb{Z}$ is the function which models consumption and production of resources by actions (a positive integer means consumption, a negative one production). Let $\mathit{cons}_{res}(\sigma) = \max(0, c(a, res))$ and $\mathit{prod}_{res}(\sigma) = -\min(0, c(a, res))$. We stipulate that $c(idle, res) = 0$ for all $res \in \mathsf{Res}$.

We denote by $D_A(s)$ the set of all joint actions by agents in coalition $A$ at $s$. Let $\sigma$ be a joint action by agents in $A$. The set of outcomes of this joint action in $s$ is the set of states reached when $A$ executes $\sigma$: $\mathit{out}(s, \sigma) = \{s' \in S \mid \exists \sigma' \in D(s) : \sigma = \sigma'_A \land s' = \delta(s, \sigma')\}$ (where $s'_A$ is the restriction of $\sigma'$ to $A$). A strategy for a coalition $A \subseteq \mathsf{Agt}$ is a mapping $F_A : S^+ \rightarrow \mathcal{Act}^{|A|}$ (from finite non-empty sequences of states to joint actions by $A$) such that, for every $\lambda \in S^+$, $F_A(\lambda) \in D_A(s)$. A computation $\lambda \in \mathsf{Ares}$ is consistent with a strategy $F_A$ iff, for all $i \geq 0$, $\lambda_{i+1} \in \mathit{out}(\lambda_i, F_A(\lambda_{i}))$. Overloading notation, we denote the set of all computations consistent with $F_A$ that start from $s$ by $\mathit{out}(s, F_A)$. Given a bound $b \in B$, a computation $\lambda \in \mathit{out}(s, F_A)$ is $b$-consistent with $F_A$ iff, for every $i \geq 0$, for every $a \in \mathcal{A}$,

$$\sum_{j=0}^{j=i-1} \mathit{tot}(F_a(\lambda[0, j])) + b_a \geq \mathit{cons}(F_a(\lambda[0, i]))$$

where $F_a(\lambda[0, j])$ is $a$’s action as part of the joint action returned by $F_A$ for the sequence of states $\lambda[0, j]$; $\mathit{tot}(\sigma) = \mathit{prod}(\sigma) - \mathit{cons}(\sigma)$ is the (vector) difference between the vector $\mathit{prod}(\sigma) = (\mathit{prod}_1(\sigma), \ldots, \mathit{prod}_n(\sigma))$ of resource amounts action $\sigma$ produces and the vector of resource amounts $\mathit{cons}(\sigma)$ it consumes; $b_a$ is $a$’s resource bound in $b$.

This condition requires that the amount of resources $a$ accumulated on the path so far, plus the original bound, is greater than or equal to the cost of executing the next action by $a$ in the strategy. $F_A$ is a $b$-strategy if all $\lambda \in \mathit{out}(s, F_A)$ are $b$-consistent.

In the presence of imperfect information, it makes sense to consider only uniform strategies rather than arbitrary ones.
A strategy is uniform if after epistemically indistinguishable histories, agents select the same actions. Two states $s$ and $t$ are epistemically indistinguishable by agent $a$, denoted by $s \sim_A t$, if $a$ has the same local state (knows the same formulas) in $s$ and $t$: $s \sim_A t$ iff $s_a = t_a$. For a coalition $A$, indistinguishability $s \sim_A s'$ means that $A$ as a whole has the same knowledge in the two states. Various notions of coalitional knowledge can be defined to $\sim_A$. For example, $s \sim_A t$ iff $\bigcup_{a \in A} s_a = \bigcup_{a \in A} t_a$ (the distributed knowledge of $A$ in $s$ and $t$ is the same). Another possible definition of $s \sim_A t$ is $\forall a \in A (s_a = t_a)$. $\sim_A$ can be lifted to histories in the obvious way: $s_1, \ldots, s_k \sim_A t_1, \ldots, t_k$ iff for all $j \in [1, k]$, $s_j \sim_A t_j$.

**Definition 2.** A strategy $F_A$ for $A$ is coalition-uniform with respect to $\sim_A$ if for all $s \sim_A t$, $F_A(s) = F_A(t)$.

Note that any notion of action choice based on coalition knowledge presupposes that agents in the coalition share knowledge for the purpose of action selection. In other words, there is a "silent step" before action selection when agents in the coalition communicate with each other instantaneously and without any cost. The only explicit and potentially resource consuming communication actions which may be necessary for a successful strategy are actions communicating with agents outside of the coalition.

The truth definition for $\mathbf{RB}^\pm \mathbf{ATSEL}$ with coalition-uniform strategies (parameterised by $\sim_A$) is as follows:

- $M, s \models p$ iff $p \in s$.
- boolean connectives have standard truth definitions.
- $M, s \models (\mathbb{A}^b) \phi$ iff $\exists$ coalition-uniform $b$-strategy $F_A$ such that for all $\lambda \in \text{out}(s, F_A)$: $M, \lambda[1] \models \phi$.
- $M, s \models (\mathbb{A}^b) \phi \land \psi$ iff $\exists$ coalition-uniform $b$-strategy $F_A$ such that for all $\lambda \in \text{out}(s, F_A)$, $\exists \geq 0$: $M, \lambda[i] \models \psi$ and $M, \lambda[j] \models \phi$ for all $j \in \{0, \ldots, i - 1\}$.
- $M, s \models (\mathbb{A}^b) \phi \land \psi$ iff $\exists$ coalition-uniform $b$-strategy $F_A$ such that for all $\lambda \in \text{out}(s, F_A)$ and $i \geq 0$: $M, \lambda[i] \models \phi$.
- $M, s \models K_a \phi$ iff $\phi \in s_a$.

Note that we do not impose any conditions on the syntactic knowledge (not consistency, not veracity etc.). Of course, in a particular modelling scenario such conditions may be imposed. The general results for decidability of model-checking stated below hold for such special cases too. They also hold for strong coalition uniformity where the truth definition for coalition modalities requires the existence of a coalition-uniform strategy from every indistinguishable state. For example, for $M, s \models (\mathbb{A}^b) \phi$ strong coalition uniformity requires that $\forall s' \sim_A s, \exists$ coalition-uniform $b$-strategy $F_A$ such that for all $\lambda \in \text{out}(s', F_A)$: $M, \lambda[1] \models \phi$.

3 Model-Checking $\mathbf{RB}^\pm \mathbf{ATSEL}$

In this section, we prove the following general result:

**Theorem 1.** The model-checking problem for $\mathbf{RB}^\pm \mathbf{ATSEL}$ with coalition-uniform strategies, with respect to any decidable notion of $\sim_A$, is decidable.

To prove decidability we give an algorithm which, given a structure $M = (\Phi, \text{Agt}, \text{Res}, S, \Pi, \text{Act}, d, \epsilon, \delta)$ and a formula $\phi_0$, returns the set of states $[\phi_0]_M$ satisfying $\phi_0$: $[\phi_0]_M = \{ s \mid M, s \models \phi_0 \}$. The theorem follows from Lemmas 1 and 2 which establish termination and correctness of the algorithm respectively.

Algorithm 1 Labelling $\phi_0$

1: function $\mathbf{RB}^\pm \mathbf{ATSEL}$-LABEL($M, \phi_0$)
2: for $\phi' \in \text{Sub}(\phi_0)$ do
3: case $\phi' = p$, $\neg \phi$, $\phi \lor \psi$ standard, see [Alur et al., 2002]
4: case $\phi' = K_a \phi$
5: $[\phi']_M' \leftarrow \{ s \mid s \in S \land \phi \in s_a \}$
6: case $\phi' = (\mathbb{A}^b) \phi$
7: $[\phi']_M' \leftarrow \text{Pre}(A, [\phi]_M, b)$
8: case $\phi' = (\mathbb{A}^b)^{\mathbb{A}^b} \phi$
9: $[\phi']_M' \leftarrow \{ s \mid s \in S \land \text{U}n\tilde{t}il\text{ }(\text{node}(a, b), \{ \}, (\mathbb{A}^b)^{\mathbb{A}^b} \phi) \}$
10: case $\phi' = (\mathbb{A}^b) \Box \phi$
11: $[\phi']_M' \leftarrow \{ s \mid s \in S \land \text{Box}(\text{node}(a, b), \{ \}, (\mathbb{A}^b) \Box \phi) \}$
12: return $[\phi_0]_M$
Algorithm 2 Labelling $\langle A^B \rangle \diamond U \psi$

1: function $B \leftarrow \langle (\diamond A) \rangle U \psi$
2: if $B = [\ ]$ then
3: return true
4: $n \leftarrow \text{hd}(B)$
5: if $\exists n' \in p(n) : s(n') = s(n) \land (\forall i, k : e_{i,k}(n') \geq e_{i,k}(n))$ then
6: return false
7: for $i, k \in \{ i \in \text{Res}, k \in A \} \leftarrow \exists n' \in p(n) : s(n') = s(n) \land e_{i,k}(n') < e_{i,k}(n) \land (\forall j, m : e_{j,m}(n') \leq e_{j,m}(n))$ do
8: $e_{i,k}(n) \leftarrow \infty$
9: if $s(n) \in [\psi]_M$ then
10: return until until $(t(l)(B), C, \langle (\diamond A) \rangle U \psi)$
11: if $s(n) \notin [\phi]_M$ then
12: return false
13: if $\exists n' \in C : p(n) \cdot n \sim_A p(n')[1, p(n) \cdot n]$ then
14: $\sigma \leftarrow a(p(n))[p(n) \cdot n + 1$
15: if $\sigma \in D_{A}(s(n)) \land \text{cons}(\sigma) \leq e(n)$ then
16: $P \leftarrow \{\text{node}(n, s', s') \mid s' \in \text{out}(s(n), \sigma)\}$
17: until $(P \circ t(l)(B), C, \langle (\diamond A) \rangle U \psi)$
18: else
19: Act $A \leftarrow \{\sigma \in D_{A}(s(n)) \mid \text{cons}(\sigma) \leq e(n)\}$
20: for $\sigma \in \text{Act } A$ do
21: $P \leftarrow \{\text{node}(n, s, s') \mid s' \in \text{out}(s(n), \sigma)\}$
22: if until $(P \circ t(l)(B), C, \langle (\diamond A) \rangle U \psi)$ then
23: return true
24: return false

Algorithm 3 Labelling $\langle A^B \rangle \square \phi$

1: function $B \leftarrow \langle (\square A) \rangle \square \phi$
2: if $B = [\ ]$ then
3: return true
4: $n \leftarrow \text{hd}(B)$
5: if $s(n) \notin [\phi]_M$ then
6: return false
7: if $\exists n' \in p(n) : s(n') = s(n) \land (\forall j, k : e_{j,k}(n') \geq e_{j,k}(n)) \land (\exists j, k : e_{j,k}(n') > e_{j,k}(n))$ then
8: return false
9: if $\exists n' \in p(n) : s(n') = s(n) \land (\forall j, k : e_{j,k}(n') \leq e_{j,k}(n))$ then
10: return box $(t(l)(B), C, \langle (\square A) \rangle \square \phi)$
11: if $\exists n' \in C : p(n) \cdot n \sim_A p(n')[1, p(n) \cdot n]$ then
12: $\sigma' \leftarrow a(p(n))[p(n) \cdot n + 1$
13: if $\sigma \in D_{A}(s(n)) \land \text{cons}(\sigma) \leq e(n)$ then
14: $P \leftarrow \{\text{node}(n, s, s') \mid s' \in \text{out}(s(n), \sigma)\}$
15: return box $(P \circ t(l)(B), C, \langle (\square A) \rangle \square \phi)$
16: else
17: Act $A \leftarrow \{\sigma \in D_{A}(s(n)) \mid \text{cons}(\sigma) \leq e(n)\}$
18: for $\sigma \in \text{Act } A$ do
19: $P \leftarrow \{\text{node}(n, s, s') \mid s' \in \text{out}(s(n), \sigma)\}$
20: if box $(P \circ t(l)(B), C, \langle (\square A) \rangle \square \phi)$ then
21: return true
22: return false

$P(n') = [p(n) \cdot n], a(n') = \sigma,$ and for all resources $i$ and agents $k \in A,$ $e_{i,k}(n') = e_{i,k}(n) + \text{prod}_k(\sigma_k) - \text{cons}_i(\sigma_k).$ In addition, we assume functions $hd(u), tl(u)$ which return the head and tail of a list $u,$ and $u \circ v$ which concatenates the lists $u$ and $v.$ (We abuse notation slightly, and treat sets as lists, e.g., use $hd(u)$ where $u$ is a set, to return an arbitrary element of $u,$ and use $\circ$ between a set and a list.)
i.e., \( p(n) \) ends in a loop. For \( \text{box} \) the loop check is slightly different. If the loop decreases the amount of at least one resource for one agent without increasing the availability of any other resource, it cannot form part of a successful strategy, and the search terminates returning false. If a non-decreasing loop is found, then it is possible to maintain the invariant \( \phi \) forever without expending any resources, and the search terminates on the current branch and continues on a different branch by expanding the next open node in \( B \) and adding the current node \( n \) to the set of closed nodes. The remaining cases are similar to \( \text{until} \). If the current branch is not closed, search continues on the branch, first checking whether an action is required for the strategy to be coalition-uniform, and, if not, for each action that is possible in the current state given the current resource availability.

**Lemma 1** (Termination). Algorithm 1 terminates.

**Proof.** All the cases in Algorithm 1 apart from the calls to Algorithms 2 and 3 clearly terminate. It therefore suffices to show that the calls to Algorithms 2 and 3 terminate.

In order to prove termination, we first show (Claim 1) that on each path explored by Algorithm 2 or Algorithm 3 there is no infinite loop where nodes with the same state and incomparable \( e(n) \) occur. This implies that the tree explored for each element of \( B \) is of finite depth, since the number of states is finite and repeated states will necessarily occur in the search, and if the resource availability vectors are comparable the search will terminate for that node. Algorithm 2 returns false for a non-increasing loop on line 6, and resets resource availability to \( \infty \) on line 8 for an increasing loop; if the same resource-increasing loop is encountered again with all resources set to \( \infty \) or unchanged, the algorithm will return false on line 6. Algorithm 3 terminates returning false on line 8 if the loop is decreasing, and calls itself on the next member of \( B \) on line 10 if the loop is non-decreasing. Second, we show that (Claim 2) there cannot be infinitely many recursive calls generated by calls on line 10 of Algorithm 2 and of Algorithm 3. We do this by showing that the list \( B \) containing paths that must be checked with respect to a currently successful strategy, will eventually become empty. Together, these two claims provide the proof of the lemma, because they guarantee that after a finite number of recursive calls both algorithms terminate.

**Claim 1:** Algorithm 2 and Algorithm 3 cannot generate a path where nodes with the same state and incomparable \( e(n) \) occur infinitely often.

This part of the termination proof is similar to that in [Alechina et al., 2014] which in turn is similar to the proof of Lemma 1 in [Reisig, 1985, p.70], and proceeds by induction on the number of resource/agent pairs \( m \). For \( m = 1 \), since \( e(n) \) is always positive, the claim is immediate. Assume the claim holds for \( m \) and let us show it for \( m + 1 \). In other words, the first \( m \) positions in \( e(n) \) will eventually become comparable. Then the \( m + 1 \) position will become comparable since there are only finitely many positive integers which are smaller than a given \( c_{m+1}(n) \).

**Claim 2.** There can be only finitely many calls generated by the coalition uniformity check (line 13 of Algorithm 2 and line 11 of Algorithm 3).

Here we need to show that the open list \( B \) is used to explore a finite tree, hence \( B \) will eventually become empty. The depth of this tree is bounded by the depth of the longest possible path. Since the relation \( \sim \) only holds between the paths of the same length, Claim 1 is sufficient to limit the depth of the tree. The finite branching factor of the tree follows from the fact that the set \( \text{out}(s, \sigma_A) \) is always finite.

**Lemma 2** (Correctness). Given a model \( M \), a state \( s \) in \( M \) and a formula \( \phi \), Algorithm 1 labels \( s \) with \( \phi \) iff \( M, s \models \phi \).

**Proof.** The proof for all the cases in Algorithm 1 apart from the calls to Algorithms 2 and 3 is straightforward.

Let us look at the case for \( \langle \langle A^b \rangle \rangle \phi \cup \psi \). We need to show that a call to \( \text{until}([\text{node0}(s, b)], \{ \}, \langle \langle A^b \rangle \rangle \phi \cup \psi \) returns true if, and only if, \( M, s \models \langle \langle A^b \rangle \rangle \phi \cup \psi \) and similarly for \( \langle \langle A^b \rangle \rangle \square \phi \). By inductive hypothesis, the algorithm only explores paths where \( \phi \) holds (line 11) until \( \psi \) is encountered (line 9), so the pure temporal semantics of \( \mathcal{U} \) is respected. Note that it is enough to find a finite strategy which is guaranteed to achieve a state where \( \psi \) is true. After that, the agents can select the \( \text{idle} \) action in all subsequent histories, which both ensures coalition uniformity and does not require any resources. The proof as regards resource bounds (and whether it is safe to reset a bound to \( \infty \) when a productive loop is encountered, and explore a productive loop only once) is similar to the one for RB±ATL [Alechina et al., 2014]. However, in addition we need to show that the algorithms return true if and only if there is a satisfying coalition-uniform strategy. Assume that the algorithm returns true. We need to show that the strategy found is coalition-uniform. This is ensured by the check on line 13. A current successful strategy is kept in \( \text{current} \), and for all coalition-indistinguishable paths we check whether the same strategy returns true, and only then return true, otherwise we backtrack and try another strategy. For the other direction, assume that there is a coalition-uniform strategy for \( A \) to enforce \( \langle \langle A^b \rangle \rangle \phi \cup \psi \). An inspection of Algorithm 2 shows that if such a strategy exists, then there exists a sequence of recursive calls by the algorithm (corresponding to the choice of actions given by the strategy) which results in the algorithm returning true.

The case of \( \langle \langle A^b \rangle \rangle \square \phi \) is similar. We ensure that Algorithm 3 returns a coalition-uniform strategy by an identical check on line 11.

**4 Verifying Norm Monitoring Strategies**

In this section, we show how RB±ATSEL can be used to reason about knowledge-based resource bounded strategies in a simple norm monitoring scenario. In the scenario, agents monitor and enforce a norm that visitors to a museum are prohibited from getting too close to the artwork on display: if a visitor approaches the artwork, s/he is warned; if s/he approaches again after being warned, s/he is required to leave the museum. For simplicity, we assume the museum has a single exhibition room, there are two monitoring agents 1 and 2, and one visitor 3. At each timestep, the visitor can perform an \( \text{idle} \) action or approach the artwork, \( \text{app} \). Agents 1 and 2 can perform an \( \text{idle} \) action, an observation action, \( \text{obs} \), issue a warning \( \text{warn} \), escort the visitor out of the museum \( \text{rem} \),
or recharge their battery \( gen \). The agents require a single resource, energy. The \( gen \) action produces energy; all other actions apart from \( idle \) consume energy.

We use propositions \( a \) to denote that the visitor has approached the artwork, \( c_i \) \((i \in \{1, 2\})\) to denote that agent \( i \) has just charged their battery, \( w \) to denote that the visitor has been warned, and \( r \) to denote that the visitor has been removed from the museum. The global system state is represented by \( s = (s_1, s_2, s_3, s_e) \), where \( s_i \) \((i \in \{1, 2, 3\})\) is the local state of \( i \), and \( s_e \) is the state of the environment. The set of formulas \( \Phi \) which constitute possible contents of agents’ states includes information on whether the agents have (just) charged, whether the visitor has approached the artwork, been warned, or removed from the museum.

The museum scenario can be modelled by the structure \( M = (\Phi, \text{Agt}, \text{Res}, S, \Pi, \text{Act}, d, c, \delta) \), where \( \Phi = \{a, c_1, c_2, r, w\}, \text{Agt} = \{1, 2, 3\}, \text{Res} = \{\text{energy}\}, S = 2^{(a,c_1,w)} \times 2^{(a,c_2,w)} \times 2^{(r,w)} \times 2_{\Pi}, \Pi = \{a, c_1, c_2, w, r\}, \text{Act} = \{\text{idle, app, obs, warn, rem, gen}\} \).

\( d \) is defined for all \( s \in S \) as follows:

1. \( \text{idle} \in d(s, i) \) for all \( i \in \{1, 2, 3\} \)
2. \( \text{app} \in d(s, 3) \) iff \( r \not\in s_3 \)
3. \( \text{obs} \in d(s, i) \) for all \( i \in \{1, 2\} \)
4. \( \text{gen} \in d(s, i) \) for all \( i \in \{1, 2\} \)
5. \( \text{warn} \in d(s, i) \) for all \( i \in \{1, 2\} \) iff \( a \in s_i \) (a warning is only issued if a monitor knows the visitor has approached the artwork)
6. \( \text{rem} \in d(s, i) \) for all \( i \in \{1, 2\} \) iff \( a, w \in s_i \) (the visitor is only removed if s/he approaches the artwork and a warning has been issued)

\( c(\text{idle, energy}) = 0, c(\text{gen, energy}) = -2, c(\alpha, energy) = 1 \) for \( \alpha \in \text{Act} \setminus \{\text{idle, gen}\} \)

\( \delta \) is defined based on the following post conditions of actions (action preconditions are given by \( d \)):

1. \( \text{idle} \) performed by agent \( i \in \{1, 2\} \) removes \( c_i \) from \( s_i \) and \( s_e \); \( \text{idle} \) performed by agent 3 removes \( a \) from \( s_e \)
2. \( \text{app} \) performed by agent 3 adds \( a \) to the state of the environment
3. \( \text{obs} \) performed by agent \( i \in \{1, 2\} \) removes \( c_i \) from \( s_i \) and \( s_e \), and, if performed in a state where \( a \) is true (false), adds (removes) \( a \) to (from) \( i \)’s local state
4. \( \text{warn} \) performed by agent \( i \in \{1, 2\} \) removes \( c_i \) from \( s_i \) and \( s_e \), and adds \( w \) to \( s_1, s_2, s_3 \) and \( s_e \)
5. \( \text{rem} \) performed by agent \( i \in \{1, 2\} \) removes \( c_i \) from \( s_i \) and \( s_e \), and adds \( r \) to \( s_3, s_e \)
6. \( \text{gen} \) performed by agent \( i \in \{1, 2\} \) adds \( c_i \) to \( s_i \) and \( s_e \)

The following property states that if the visitor approaches the artwork, then this will be known by one of the monitoring agents in the next state:

\[ \langle\langle 1, 2 \rangle^1.0 \rangle \Box(a \rightarrow \langle\langle 1, 2 \rangle^0.0 \rangle \Diamond (K_1a \lor K_2a)). \]

This formula is true for a notion of coalition uniformity based on distributed knowledge of the coalition. The strategy is as follows: agents take turns charging and observing; agent 1 chooses \( obs \) in the state where both agents’ states don’t contain \( c_1 \), hence it needs 1 unit of energy to start with. The following properties state that the monitoring agents are able to warn the visitor in two steps after the visitor’s approach, and that after being warned, the visitor will be removed directly after another approach. They are true under the same notion of coalition uniformity.

\[ \phi_w = \langle\langle 1, 2 \rangle^1.0 \rangle \Box(a \rightarrow \langle\langle 1, 2 \rangle^0.0 \rangle \Diamond (\langle\langle 1, 2 \rangle^0.0 \rangle \Diamond w)) \]

\[ \langle\langle 1, 2 \rangle^0.0 \rangle \Box(w \rightarrow \phi_r), \text{where} \phi_r = \phi_w[w/r]. \]

5 Related Work

The motivation of work on epistemic logics where acquiring information requires resources [Jamroga and Tabatabaei, 2013; Naumov and Tao, 2015] is very similar to ours, however the technical approach is very different. In [Jamroga and Tabatabaei, 2013], a set of states an agent considers possible is updated by observations (which eliminate some states), and observations have resource costs. The logic introduced in the paper can express statements such as ‘\( i \) can potentially achieve knowledge of whether \( \phi \) is true under resource bound \( b \).’ In [Naumov and Tao, 2015], edges in an epistemic indistinguishability relation have weights corresponding to the costs of removing them (obtaining information which would distinguish the states). This allows the authors to define weighted knowledge operators which represent the costs of coming to know whether some proposition is true.

Other related work falls broadly into three categories: work on model-checking resource logics (without epistemics), work on model-checking epistemic ATL (under standard semantics for epistemics and without knowledge change), and work on model-checking DEL and epistemic planning. There exist several formalisms that extend Alternating Time Temporal Logic (ATL), [Alur et al., 2002] with reasoning about resources available to agents and production and consumption of resources by actions. When the production of resources is allowed, the model-checking problem for many (but not all) of these logics is undecidable (for a survey, see [Alechina et al., 2015]). Epistemic ATL has been studied extensively, see e.g., [Ivan der Hoek and Wooldridge, 2002; Ågotnes, 2006; Lomuscio et al., 2009; Guelev et al., 2011; Dima and Tiplea, 2011]. Its model-checking problem with perfect recall and uniform strategies was shown to be undecidable in the case of more than one agent in [Dima and Tiplea, 2011]. In [Guelev et al., 2011], it was shown that if uniform strategies are defined in terms of distributed knowledge of the coalition, the model-checking problem becomes decidable. The technique used to prove this is very different from the one used in this paper. Various notions of coalition uniformity were studied in [Ivan Ditmarsch and Knight, 2014], and justified for a setting where agents in a coalition share their information; the model-checking problem for the resulting logic was not considered. There is a large body of work on DEL. The model-checking problem for full DEL was shown to be undecidable in [Aucher and Bolander, 2013] and decidable for a fragment of DEL in [Aucher and Schwarzentruber, 2013]. DEL-based epistemic planning is also undecidable in general, but is tractable for some special cases [Yu et al., 2013; Bolander et al., 2015].
Acknowledgement. This work was supported by the Engineering and Physical Sciences Research Council [grant EP/K033905/1].

References


