Alternating-time Temporal Epistemic Logic and its Application in AI

Mitch Lobbes (2627692), Yuyu Bai (2732696), Loïc Macken (2611927)

Vrije Universiteit Amsterdam

Abstract. The purpose of this study is to do a survey of Alternating-time Temporal Epistemic Logic (ATEL). ATEL is a concise and expressive language for reasoning about multi-agent systems. The syntax and semantics of ATEL are introduced in this survey. There are also a few AI applications mentioned. Finally, the model checking algorithm is described in detail in this study.

Keywords: multiagent systems, temporal logic, epistemic, model checking

1 Introduction

Alternating-time Temporal Epistemic Logic (ATEL) is an expansion upon Alternating-time Temporal Logic (ATL)[1], which is again based on Computation Tree Logic (CTL)[2]. ATL generalizes CTL and replaces path quantifiers by cooperation modalities, this allows it to express situations where agents cooperate and form a coalition in order to achieve a certain goal. Incorporating epistemic logic (the logic of knowledge) into ATL then allows for the modelling of knowledge of agents, together with their coalition. The resulting language of logic became known as ATEL[6].

A computerized system built of several interacting intelligent agents is known as a multi-agent system (MAS or "self-organized system"). Multi-agent systems can address problems that a solo agent or a monolithic system would find difficult or impossible to solve. ATEl is widely used in multi-agent systems[8]. There are many applications such as using ATEL to address knowledge forgetting problems[7]. In this paperwe briefly introduced some applications for ATEL such as WhatsApp encryption and games.

The structure of this paper is as follows. We begin with the syntax and semantics of ATEL in the next section. After that, some applications of ATEL in AI field are discussed. This section also includes a detailed introduction of a model checking algorithm. Finally, we make some conclusions.

2 Syntax and Semantics

This section describes the syntax and semantics of ATEL, providing insight into the inner workings of the language.

2.1 Syntax

The syntax of ATEL is an expansion of the syntax of ATL and thus includes the operators from CTL and propositional logic. Starting from the basis of propositional logic, the following operators and connectives are defined in ATEL:

- Propositional Logic:
  $\wedge$: Logical AND
  $\vee$: Logical OR
  $\neg$: Logical NOT
  $\rightarrow$: If the left-hand side holds then the right-hand side must also hold
  $\leftrightarrow$: The right-hand side holds if and only if the left-hand side holds
- Alternating-time Temporal Epistemic Logic:
  $\langle\langle \Gamma \rangle\rangle \phi$: The group of agents Γ cooperates to ensure that ϕ holds
  $[[\Gamma]]\phi$: The group of agents Γ cannot prevent that ϕ occurs
  $\langle\langle \Gamma \rangle\rangle \bigcirc \phi$: The group of agents Γ can cooperate to ensure that ϕ is true at the next state
  $\langle\langle \Gamma \rangle\rangle \Box \phi$: The group of agents Γ can cooperate to ensure that ϕ is always true
  $\langle\langle \Gamma \rangle\rangle \Diamond \phi$: The group of agents Γ can cooperate to ensure that ϕ is true at some state along the path
  $\langle\langle \Gamma \rangle\rangle \phi U \psi$: The group of agents Γ can cooperate to ensure that ϕ is true at all states until ψ is true
  $K_a \phi$: Agent a has individual knowledge that ϕ.
  $C_\Gamma \phi$: The group of agents Γ have common knowledge that ϕ
  $E_\Gamma \phi$: Every agent in the group Γ knows that ϕ

2.2 Semantics

The semantics of ATL is defined by Alur et al.(2002)[3] using concurrent game structures (CGSs). The following definition, which is used throughout the work, differs slightly from the original in that actions are designated by arbitrary labels instead of natural integers. The difference is minor in terms of form, but the following model better fits the semantic assumption. Now let us dive into how ATEL can be applied to solve logical problems, by first describing the systems to which it is applied and then elaborating upon the computations.

Transition Systems An Alternating Epistemic Transition System is defined as a tuple

$$\langle \Pi, \Sigma, Q, \sim_a, ..., \sim_n, \pi, \delta \rangle \tag{1}$$

where:

- Π is a set of atomic propositions
- $\Sigma = \{a_1, ..., a_n\}$ is a set of agents
- Q is a set of states
- $\sim_a \subseteq Q \times Q$ is an epistemic accessibility relation for each agent $a \in \Sigma$
- $\pi : Q \rightarrow 2^{\Pi}$ is the set of primitive propositions satisfied at each state
- $\delta : Q \times \Sigma \rightarrow 2^{2^Q}$ is the system transition function

Epistemic Relations If $\Gamma \subseteq \Sigma$, then $\sim^E_\Gamma = (\cup_{a \in \Gamma} \sim_a)$. In addition, $\sim^C_\Gamma$ is defined as the transitive closure of $\sim^E_\Gamma$.

Computations A computation of an AETS is a sequence of successive states $\lambda = q_0, q_1, ...$ such that for all $u > 0$, q_u is a successor of q_{u-1}.

Strategies and their outcomes A strategy defines the plan of an agent or a group of agents. Each agent has goals they want to achieve and a defined strategy on how to achieve those goals. For a set of agents Γ starting at state $q \in Q$, we can define a set $out(q, F_\Gamma)$ that contains all possible q-computations that could emerge as a result of the agents in Γ following their strategies F_Γ. Note that this means that $out(q, F_\Sigma)$ contains only a single q-computation and $out(q, F_\emptyset)$ contains all possible q-computations.

Operators We define the following rules using S to represent an AETS and q to represent a state in S:

- $S, q \models \top$
- $S, q \models p$, if and only if p is part of the propositions at state q: $p \in \pi(q)$
- $S, q \models \neg\phi$ if and only if $S, q \not\models \phi$
- $S, q \models \phi \vee \psi$ if and only if $S, q \models \phi$ or $S, q \models \psi$
- $S, q \models \langle\langle\Gamma\rangle\rangle \bigcirc \phi$ if and only if the group of agents Γ has strategies for each agent to ensure that ϕ is true at the next state.
- $S, q \models \langle\langle\Gamma\rangle\rangle\Box\phi$ if and only if the group of agents Γ has strategies for each agent to ensure that ϕ is always true.
- $S, q \models \langle\langle\Gamma\rangle\rangle\phi U\psi$ if and only if the group of agents Γ has strategies for each agent to ensure that ϕ is true until ψ is true.
- $S, q \models K_a\phi$ if and only if for all states q' where $q \sim_a q'$: $S, q' \models \phi$
- $S, q \models E_\Gamma\phi$: if and only if for all states q' where $q \sim^E_\Gamma q'$: $S, q' \models \phi$
- $S, q \models C_\Gamma\phi$: if and only if for all states q' where $q \sim^C_\Gamma q'$: $S, q' \models \phi$

3 Applications

The next section explains ATEL's applications. Several examples are mentioned and they illustrate the flexible and expressive nature of ATEL in the area of Artificial Intelligence. Another interesting general example regarding games is also discussed.

3.1 Formulating Knowledge Preconditions

Currently one of the main issues with Artificial Planning is that of formulating predefined knowledge regarding actions and plans. ATEL has a natural way of interpreting this problem and all its variants, therefore the cooperative nature of ATEL allows it to be used for multi-agent plans.

Let the following ATEL formula show that in order for agent a to reach a state where φ holds, it must have the knowledge of ψ:

$$\langle\langle a\rangle\rangle\Diamond\varphi \longrightarrow K_a\psi$$

You could compare this to logging into a computer with a password. However, it is not exactly necessary that the knowledge of the password is present right *now*, as long as it *is* present when the agent wants to log in. Furthermore, this formula deems it necessary to have the knowledge of ψ to cause φ. This formulates that the requirement in the formula given is too strong and we could improve upon it, with the following formula:

$$\langle\langle a\rangle\rangle \bigcirc \varphi \longrightarrow K_a\psi$$

The new formula emphasises that the knowledge of the password needs to be present in order to log in. Although this is rather general and does not specify what the agent needs to know in order to log in at a specific time, we could again improve further:

$$\neg\langle\langle a\rangle\rangle \bigcirc \varphi U K_a\psi$$

Now we are at the point where our agent can not log into the computer without having the knowledge of the password. However, what would be preferred is formulating the knowledge that as soon as the agent has acquired the password knowledge it is able to log in.

$$K_a\psi \longrightarrow \langle\langle a\rangle\rangle\Diamond\varphi$$

Which sequentially, due to the ability that passwords are able to change, would result in

$$K_a\psi \longrightarrow \langle\langle a\rangle\rangle \bigcirc \varphi$$

3.2 WhatsApp Encryption

WhatsApp's end-to-end encryption is used when you chat with another person using WhatsApp Messenger. End-to-end encryption ensures only you and the person you're communicating with can read or listen to what is sent, and nobody in between, not even WhatsApp. This is an example of cryptographic protocols. In these protocols, it is necessary that a key is recognised to be able to decrypt or authorise the messages. ATEL has a very useful and intuitive way of formulating these protocols. Let's formulate a message that is being encrypted by a key as:

$$\{msg\}_S$$

Adding to that, when there are two agent a (recipient) and agent b (sender) that have the knowledge of the key:

$\{msg\}_{S_{a,b}}$

This means that a way of decrypting a received message by agent b would look like:

$\{msg\}_{S_{a,b}} \wedge K_a(is - key\langle(a,b)S_{ab}\rangle) \longrightarrow \langle\langle a\rangle\rangle \bigcirc K_a said_b msg$

If agent a receives the message $\{msg\}_{S_{ab}}$ and a has the knowledge that $S_a b$ is the shared key of a and of b, then a can use the key to encrypt the whole message, and safely conclude that b said msg. This concludes that ATEL's power is its usefulness regarding protocol formulations and the ability to describe if a message is even seen in a protocol.

3.3 Games

Another example of the relevance of ATEL can be found in simple knowledge games. The most well-known knowledge game around is a normal game of cards, where the aim of the player is to find out a particular deal d of cards. Where it is simultaneously the goal to gather as much information yourself as to reduce the information available to others. Using ATEL to formulate a winning strategy for this game results in the following formula:

$d \longrightarrow \langle\langle a\rangle\rangle \bigcirc (K_a d \wedge \bigwedge_{a \neq b} \neg K_b d)$

During a game of cards every player makes different decisions at every turn. However, all these decisions are based on gathered knowledge during the game. The requirement that applies here is that given agent a, it is necessary that the agent is aware of the choices that are available, and must assume that they're available in all equivalent $\ _a$ states. This creates the following semantic formulation:

$q \ _a q' \Rightarrow \delta(q,a) = \delta(q',a)$

Which results in the next syntatic property:

$AW\langle\langle a\rangle\rangle T\varphi \longleftrightarrow K_a\langle\langle a\rangle\rangle T\varphi$

Nevertheless, this does not require the agent to make the same choices in all equivalent states, this can be added with the stipulation of the following property:

$q \ _a q' \longrightarrow f_a(\lambda * q) = f_a(\lambda * q')$

The downside is that this formula lacks the ability to say, considering its expression that given agent a's ability to cause φ, it's possible to state if the agent a will do so.

3.4 Model checking of ATEL

Model checking is an automated method for verifying correctness properties of safety-critical reactive systems. This method has been used to detect minor flaws in complicated industrial designs including sequential circuits, communication protocols, and digital controllers.[4][5] In our paper, a domain of interest (a Multi-Agent system) is described by a semantic model. The desired property of the domain (e.g., a safety requirement) is described by ATEL formulas . The fact that a domain satisfies a desired property is determined by verification which consists of computing whether a model M satisfies ϕ. And this process is model checking. In this chapter, some model checking algorithm for ATEL formula is introduced.

$||\langle\langle \Gamma \rangle\rangle \bigcirc \phi||_M$

For the formula $\langle\langle \Gamma \rangle\rangle \bigcirc \phi$ which means that a group of agents Γ has a winging strategy $F_{(\Gamma)}$ that makes sure to enter the next state which holds ϕ, the algorithm computes the set of states where this formula holds as follows:

$Q_1 = ||\phi||_M$

$Q_2 = \{s \in W : \exists s'(s \xrightarrow{F_{(\Gamma)}} s' \wedge s' \in Q_1)\}$

The first step is to find all states that hold ϕ. We call such set as Q_1. Then look back to former states. If in such state s, there exists a strategy $F_{(\Gamma)}$ of agent group Γ which can ensure to enter to a state s' and s' belongs to Q_1, s belongs to Q_2.

$||\langle\langle \Gamma \rangle\rangle \Box \phi||_M$

For the formula $\langle\langle \Gamma \rangle\rangle \Box \phi$ which means if the system in such states, a group of Γ has a winging strategy $F_{(\Gamma)}$ that makes sure every future state holds ϕ, the algorithm computes the set of states where this formula holds as follows:

$Q_1 = ||\phi||_M$

Q_2= Q_1 - $\{s \in W : \neg\exists s'(s \xrightarrow{F_{(\Gamma)}} s' \wedge s' \in Q_1)\}$

Q_3= Q_2 - $\{s \in W : \neg\exists s'(s \xrightarrow{F_{(\Gamma)}} s' \wedge s' \in Q_2)\}$

...

$Q_n = Q_{n-1}$ - $\{s \in W : \neg\exists s'(s \xrightarrow{F_{(\Gamma)}} s' \wedge s' \in Q_{n-1})\}$

Stops until $Q_n = Q_{n-1}$.

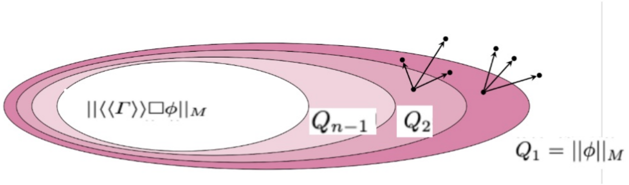

Fig. 1: Graphical representation of the algorithm

The first step is to find all states that hold ϕ. We call such set as Q_1. Then look back to former states. If in such state s, s also in set Q_1 and there exists a strategy $F_{(\Gamma)}$ of agent group Γ which can ensure to enter to a state s' and s' belongs to Q_1, s belongs to Q_2. Keep on doing this procedure and stop until Q_n is equal to Q_{n-1}.

$||\langle\langle\Gamma\rangle\rangle\phi U\psi||_M$

For the formula $\langle\langle\Gamma\rangle\rangle\phi U\psi$ which means if the system in such states, a group of Γ has a winging strategy $F_{(\Gamma)}$ that make sure a future state hold ψ and every intermediate state before that hold ϕ, the algorithm computes the set of states where this formula holds as follows:

$Q_1 = ||\psi||_M$
$Q_2 = Q_1 \cup \{s \in W : s \models \phi \land \exists s'(s \xrightarrow{F_{(\Gamma)}} s' \land s' \in Q_1)\}$
$Q_3 = Q_2 \cup \{s \in W : s \models \phi \land \exists s'(s \xrightarrow{F_{(\Gamma)}} s' \land s' \in Q_2)\}$
...
$Q_n = Q_{n-1} \cup \{s \in W : s \models \phi \land \exists s'(s \xrightarrow{F_{(\Gamma)}} s' \land s' \in Q_{n-1})\}$
Stops until $Q_n = Q_{n-1}$.

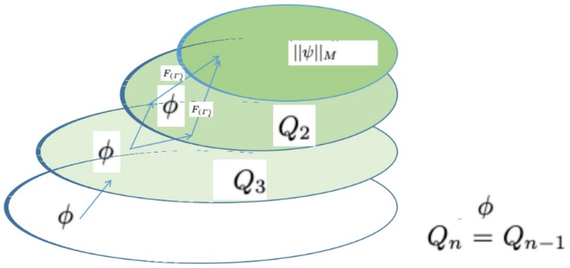

Fig. 2: Graphical representation of the algorithm

The first step is to find all states that hold ψ. We call such set as Q_1. Then look back to former states. If in such state s, there exists a strategy $F_{(\Gamma)}$ of agent group Γ which can ensure to enter to a state s' and s' belongs to Q_1 and s hold ϕ, s belongs to Q_2. Then enlarge Q_2 by also adding states in Q_1. Keep on doing this procedure and stop until Q_n is equal to Q_{n-1}.

$||K_\Gamma \phi||_M$
For the formula $K_\Gamma \phi$ which means ϕ is a group knowledge of agent group Γ, the algorithm computes the set of states where this formula holds as follows:

$Q_1 = ||\phi||_M$
Q_2= $\{s \in W : s(\bigcup_{a_i \in \Gamma} \sim a_i)s' \wedge s' \in Q_1\}$

The first step is to find all states that hold ϕ. We call such set as Q_1. We enlarge Q_1 by adding new elements to it. The enlarged set is called Q_2. If an agent from the group Γ can't distinguish between two states which one of them is in set Q_1, we include another state also in Q_2.

$||C_\Gamma \phi||_M$
For the formula $C_\Gamma \phi$ which means ϕ is common knowledge of agent group Γ, the algorithm computes the set of states where this formula holds as follows:

Q_1= $||\phi||_M$
Q_2= $\{s \in W$: There exist a computation $s = s0 \sim_{b1} s_1 \sim_{b2} s_2 ... \sim_{bn} s_n = s'$ from s to s' with n$\geq$ 0 for some agents $bn \in \Gamma$, and s' $\in$Q$_1$ $\}$

The first step is to find all states that hold ϕ. We call such set as Q_1. We enlarge Q_1 by adding new elements to it. The enlarged set is called Q_2. If there exists a computation $s = s0 \sim_{b1} s_1 \sim_{b2} s_2 ... \sim_{bn} s_n = s'$
from s to s' with n$\geq$ 0 for some agents $bn \in \Gamma$, and s' $\in Q_1$, we include s in Q_2.

4 Conclusion

In this paper ATEL's expressive and cooperative power is explained, described and illustrated. It's based upon the expansion of ATL, which on its own is based on CTL. Therefore this means its syntax is also mostly based upon, and consists of operators from the CTL and propositional logic. The expressiveness of ATEL's syntax is demonstrated with several examples, one of them regarding formulating knowledge preconditions, where the cooperative nature of the ATEL syntax makes it a very useful logic for multi-agent systems. Another example regarding encryption of WhatsApp messages was discussed and how ATEL is used to formulate seen and send, encrypted and decrypted messages and how in general protocol formulations are described. Furthermore the illustration of a simple card game is formulated using ATEL syntax to describe ATEL's relevance in a normal day-to-day application.

References

1. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. J. ACM **49**(5), 672–713 (sep 2002). https://doi.org/10.1145/585265.585270, https://doi.org/10.1145/585265.585270
2. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. J. ACM **49**(5), 672–713 (sep 2002). https://doi.org/10.1145/585265.585270, https://doi.org/10.1145/585265.585270
3. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. Journal of the ACM (JACM) **49**(5), 672–713 (2002)
4. Browne, M., Clarke, E., Dill, D.: Checking the correctness of sequential circuits. In: 1985 IEEE Proceedings of the International Conference on Computer Design. pp. 545–548 (1985)
5. Clarke, E.M., Emerson, E.A., Sistla, A.P.: Automatic verification of finite-state concurrent systems using temporal logic specifications. ACM Transactions on Programming Languages and Systems (TOPLAS) **8**(2), 244–263 (1986)
6. van der Hoek, W., Wooldridge, M.: Cooperation, Knowledge, and Time: Alternating-time Temporal Epistemic Logic and its Applications. Studia Logica **75**(1), 125–157 (2003). https://doi.org/10.1023/a:1026185103185
7. Lin, Y., Wang, X.: Semantics of Knowledge Forgetting in Alternating-Time Temporal Epistemic Logic. 2009 First International Conference on Information Science and Engineering (2009). https://doi.org/10.1109/icise.2009.1054
8. Wooldridge, M.: An introduction to multiagent systems. John wiley & sons (2009)