Wiebe van der Hoek
Michael Wooldridge

Cooperation, Knowledge, and Time: Alternating-time Temporal Epistemic Logic and its Applications

Abstract. Branching-time temporal logics have proved to be an extraordinarily successful tool in the formal specification and verification of distributed systems. Much of their success stems from the tractability of the model checking problem for the branching time logic CTL, which has made it possible to implement tools that allow designers to automatically verify that systems satisfy requirements expressed in CTL. Recently, CTL was generalised by Alur, Henzinger, and Kupferman in a logic known as "Alternating-time Temporal Logic" (ATL). The key insight in ATL is that the path quantifiers of CTL could be replaced by "cooperation modalities", of the form $\langle\langle\Gamma\rangle\rangle$, where Γ is a set of agents. The intended interpretation of an ATL formula $\langle\langle\Gamma\rangle\rangle\varphi$ is that the agents Γ can cooperate to ensure that φ holds (equivalently, that Γ have a *winning strategy* for φ). In this paper, we extend ATL with *knowledge modalities*, of the kind made popular in the work of Fagin, Halpern, Moses, Vardi and colleagues. Combining these knowledge modalities with ATL, it becomes possible to express such properties as "group Γ can cooperate to bring about φ iff it is common knowledge in Γ that ψ". The resulting logic — Alternating-time Temporal Epistemic Logic (ATEL) — shares the tractability of model checking with its ATL parent, and is a succinct and expressive language for reasoning about game-like multiagent systems.

Keywords: Cooperation logic, epistemic logic, game theory, model checking.

1. Introduction

Perhaps the most successful approach to reasoning about distributed computer systems involves the use of *branching time logics*, of which Computation Tree Logic (CTL) is the best known example [13]. CTL is a temporal logic that is interpreted over tree-like structures, in which nodes represent time points and arcs represent transitions between time points. In distributed systems applications, the set of all paths through a tree structure is assumed to correspond to the set of all possible computations of a system. CTL combines *path quantifiers* "A" and "E" for expressing that a certain series of events

Special Issue The Dynamics of Knowledge
Edited by **Wiebe van der Hoek and Michael Wooldridge**

will happen on all paths and on some path respectively, with *tense modalities* for expressing that something will happen eventually on some path ($\Diamond$), always on some path ($\Box$) and so on. Thus, for example, by using CTL-like logics, one may express properties such as "on all possible computations, the system never enters a fail state", which is represented by the CTL formula $\mathsf{A}\Box\neg fail$. Although the computational complexity of the deductive proof problem for CTL is prohibitively expensive (it is EXPTIME-complete [13, p.1037]), the *model checking* problem for CTL — the problem of determining whether a given formula of CTL is satisfied in a particular CTL model — is computationally rather easy: it can be solved in time $O(|M| \times |\varphi|)$, where $|M|$ is the size of the model and $|\varphi|$ is the size of the formula to be checked [13, p.1044]. The tractability of CTL model checking has led to the development of a range of CTL model checking tools, which have been widely used in the verification of hardware and software systems [10].

Recently, *multiagent systems* have emerged as a new paradigm for understanding distributed systems [35]. In multiagent systems, computer processes are viewed as economic entities — *agents* — in the sense that interactions between them may not be cooperative, and may even be adversarial. Interactions between agents in multiagent systems can thus be understood as *games* in the sense of game theory [7, 29]. Branching time temporal logics of the CTL genus prove to be of limited value when applied to multiagent systems: given the game-like nature of multiagent systems, the kinds of properties we wish to express of them are the *powers* that the system components have. For example, we might wish to express the fact that "agent (process) 1 has the power to ensure that the system never fails". Alternatively, we might express this property as "agent 1 has a *winning strategy* for ensuring that the system never enters a fail state". Similarly, we might wish to express *cooperative powers*: "agents 1 and 2 can cooperate to ensure that the system never enters a fail state". It is not possible to capture such statements using CTL-like logics. The best one can do is either state that something will inevitably happen, or else that it may possibly happen; CTL-like logics have *no notion of agency.*

In 1997, Alur, Henzinger, and Kupferman introduced a logic intended to make good this deficit [3]. Alternating-time Temporal Logic (ATL) is a novel generalisation of CTL, in which path quantifiers are replaced by *cooperation modalities*: the ATL expression $\langle\langle\Gamma\rangle\rangle\varphi$, where Γ is a group of agents, expresses the fact that the group Γ can cooperate to ensure that φ. Thus, for example, the fact that agents 1 and 2 can ensure that the system never enters a fail state may be captured in ATL by the following formula.

$$\langle\langle 1,2\rangle\rangle\Box\neg fail$$

ATL generalises CTL because the path quantifiers A ("on all paths...") and E ("on some paths...") can be simulated in ATL by the cooperation modalities $\langle\langle\emptyset\rangle\rangle$ ("the emptyset of agents can cooperate to...") and $\langle\langle\Sigma\rangle\rangle$ ("the grand coalition of all agents can cooperate to..."). One of the fascinating aspects of ATL is that it shares with CTL the computational tractability of its model checking problem. This has led to the development of an ATL model checking system called MOCHA [4, 2].

Since the mid-1980s, *epistemic logics* — modal logics of knowledge — have found a wide range of applications in computer science and artificial intelligence [15, 27]. They have proved to be particularly useful for reasoning about multiagent systems, where it is frequently necessary to express statements such as "if agent a sends agent b message m, then eventually, b will know m" [20, 36].

In this paper, we show how ATL may be extended with knowledge modalities. The resulting logic is called *Alternating-time Temporal Epistemic Logic* (ATEL). As well as operators for representing the knowledge of individual agents, ATEL includes modalities for representing what "everyone knows" and common knowledge [15, 27]. ATEL is a succinct and very powerful language for expressing complex properties of multiagent systems. For example, the following formula expresses the fact that if it is common knowledge in group of agents Γ that φ, then Γ can cooperate to ensure ψ.

$$C_\Gamma\varphi \rightarrow \langle\langle\Gamma\rangle\rangle\Diamond\psi$$

As another example, the following ATEL formula says that, if a knows that φ, then a has a strategy to ensure that b knows φ — in other words, a can communicate what it knows to other agents.

$$K_a\varphi \rightarrow \langle\langle a\rangle\rangle\Diamond K_b\varphi$$

The remainder of the paper is structured as follows. We begin in the following section by presenting *Alternating Epistemic Transition Systems*, the semantic structures that underpin ATEL. We then introduce the logic ATEL, giving its semantics in terms of these structures. We discuss the axiomatic basis of ATEL in section 4, and in section 5 we focus in particular on the possible interactions between knowledge, ability, and cooperation. We demonstrate how ATEL can be used to succinctly express a range of desirable (and undesirable) properties of multiagent systems. We present a model checking algorithm for ATEL in section 6, and show that the complexity of the ATEL model checking problem is PTIME-complete, and hence no worse than that of ATL [3]. In section 6, we give a detailed model checking case study, in which we show how ATEL properties of a system can be verified using freely available ATL model checking tools [4, 2]. We conclude with a brief discussion on related work, and present some conclusions and possible avenues for future research.

2. Alternating Epistemic Transition Systems

We begin by introducing the semantic structures used to represent our domains. These structures are a straightforward extension of the alternating transition systems used by Alur and colleagues to give a semantics to ATL. Formally, an *alternating epistemic transition system* (AETS) is a tuple

$$\langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle,$$

where:

- Π is a finite, non-empty set of *atomic propositions*;
- $\Sigma = \{a_1, \ldots, a_n\}$ is a finite, non-empty set of *agents*;
- Q is a finite, non-empty set of *states*;
- $\sim_a \subseteq Q \times Q$ is an *epistemic accessibility relation* for each agent $a \in \Sigma$ — we usually require that each $\sim_a$ is an equivalence relation;
- $\pi : Q \to 2^{\Pi}$ gives the set of primitive propositions satisfied in each state;
- $\delta : Q \times \Sigma \to 2^{2^Q}$ is the system transition function, which maps states and agents to the choices available to these agents. Thus $\delta(q, a)$ is the set of choices available to agent a when the system is in state q. We require that this function satisfy the constraint that the system is completely controlled by its component agents: for every state $q \in Q$ and every set $Q_1, \ldots, Q_n$ of choices $Q_a \in \delta(q, a)$, the intersection $Q_1 \cap \cdots \cap Q_n$ is a singleton. This means that if every agent has made his choice, the system is completely determined.

We denote the set of sequences over Q by Q^*, and the set of non-empty sequences over Q by Q^+.

Epistemic Relations

If $\Gamma \subseteq \Sigma$, we denote the union of Γ's accessibility relations by $\sim_\Gamma^E$, so $\sim_\Gamma^E = (\bigcup_{a \in \Gamma} \sim_a)$. Also, $\sim_\Gamma^C$ denotes the transitive closure of $\sim_\Gamma^E$. We will later use

$\sim^C_\Gamma$ and $\sim^E_\Gamma$ to give a semantics to the "common knowledge" and "everyone knows" modalities in our logic [15].

Computations

For two states $q, q' \in Q$ and an agent $a \in \Sigma$, we say that state q' is an *a-successor* of q if there exists a set $Q' \in \delta(q, a)$ such that $q' \in Q'$. Intuitively, if q' is an a-successor of q, then q' is a possible outcome of one of the choices available to a when the system is in state q. We denote by $succ(q, a)$ the set of a successors to state q. We say that q' is simply a *successor* of q if for all agents $a \in \Sigma$, we have $q' \in succ(q, a)$; intuitively, if q' is a successor to q, then when the system is in state q, the agents Σ can cooperate to ensure that q' is the next state the system enters.

A *computation* of an AETS $\langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle$ is an infinite sequence of states $\lambda = q_0, q_1, \ldots$ such that for all $u > 0$, the state q_u is a successor of q_{u-1}. A computation λ starting in state q is referred to as a *q-computation*; if $u \in \mathbb{N} = \{0, 1, \ldots\}$, then we denote by $\lambda[u]$ the u'th state in λ; similarly, we denote by $\lambda[0, u]$ and $\lambda[u, \infty]$ the finite prefix $q_0, \ldots, q_u$ and the infinite suffix $q_u, q_{u+1}, \ldots$ of λ respectively.

Strategies and Their Outcomes

Intuitively, a *strategy* is an abstract model of an agents decision-making process; a strategy may be thought of as a kind of plan for an agent. By *following* a strategy, an agent can bring about certain states of affairs. Formally, a strategy f_a for an agent $a \in \Sigma$ is a total function $f_a : Q^+ \to 2^Q$, which must satisfy the constraint that $f_a(\lambda \cdot q) \in \delta(q, a)$ for all $\lambda \in Q^*$ and $q \in Q$. Given a set $\Gamma \subseteq \Sigma$ of agents, and an indexed set of strategies $F_\Gamma = \{f_a \mid a \in \Gamma\}$, one for each agent $a \in \Gamma$, we define $out(q, F_\Gamma)$ to be the set of possible outcomes that may occur if every agent $a \in \Gamma$ follows the corresponding strategy f_a, starting when the system is in state $q \in Q$. That is, the set $out(q, F_\Gamma)$ will contain all possible q-computations that the agents Γ can "enforce" by cooperating and following the strategies in F_Γ. Note that the "grand coalition" of all agents in the system can cooperate to uniquely determine the future state of the system, and so $out(q, F_\Sigma)$ is a singleton. Similarly, the set $out(q, F_\emptyset)$ is the set of all possible q-computations of the system.

3. Alternating Temporal Epistemic Logic

Alternating epistemic transition systems are the structures we use to model the systems of interest to us. We now introduce a language to represent and reason about these structures. This language — alternating temporal epistemic logic (ATEL) — is an extension of the alternating temporal logic (ATL) of Alur, Henzinger, and Kupferman [3], which in turn takes its inspiration from the branching temporal logics CTL and CTL* [13]. Just as formulae of alternating temporal logic are interpreted with respect to alternating transition systems, formulae of ATEL are interpreted with respect to the alternating epistemic transition systems introduced above.

Before presenting the detailed syntax of ATEL, we give an overview of the intuition behind its key constructs. ATEL is an extension of classical propositional logic, and so it contains all the conventional connectives that one would expect to find: $\wedge$ ("and"), $\vee$ ("or"), $\neg$ ("not"), $\rightarrow$ ("implies"), and so on. In addition, ATEL contains the temporal cooperation modalities of ATL, as follows. The formula $\langle\langle\Gamma\rangle\rangle\Box\varphi$, where Γ is a group of agents, and φ is a formula of ATEL, means that the agents Γ can work together (cooperate) to ensure that φ is always true. Similarly, $\langle\langle\Gamma\rangle\rangle\bigcirc\varphi$ means that Γ can cooperate to ensure that φ is true in the *next* state. The formula $\langle\langle\Gamma\rangle\rangle\varphi\mathcal{U}\psi$ means that Γ can cooperate to ensure that φ remains true *until* such time as ψ is true — and moreover, ψ *will* be true at some time in the future.

An ATEL formula, formed with respect to an alternating epistemic transition system $S = \langle\Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta\rangle$, is one of the following:

(S0) $\top$

(S1) p, where $p \in \Pi$ is a primitive proposition;

(S2) $\neg\varphi$ or $\varphi \vee \psi$, where φ and ψ are formulae of ATEL;

(S3) $\langle\langle\Gamma\rangle\rangle\bigcirc\varphi$, $\langle\langle\Gamma\rangle\rangle\Box\varphi$, or $\langle\langle\Gamma\rangle\rangle\varphi\mathcal{U}\psi$, where $\Gamma \subseteq \Sigma$ is a set of agents, and φ and ψ are formulae of ATEL;

(S4) $K_a\varphi$, where $a \in \Sigma$ is an agent, and φ is a formula of ATEL;

(S5) $C_\Gamma\varphi$ or $E_\Gamma\varphi$, where $\Gamma \subseteq \Sigma$ is a set of agents, and φ is a formula of ATEL.

We interpret the formulae of ATEL with respect to AETS, as introduced in the preceding section. Formally, if S is an AETS, q is a state in S, and φ is a formula of AETL over S, then we write $S, q \models \varphi$ to mean that φ is satisfied (equivalently, true) at state q in system S. The rules defining the satisfaction relation $\models$ are as follows:

- $S, q \models \top$
- $S, q \models p$ iff $p \in \pi(q)$ (where $p \in \Pi$);

- $S, q \models \neg\varphi$ iff $S, q \not\models \varphi$;
- $S, q \models \varphi \vee \psi$ iff $S, q \models \varphi$ or $S, q \models \psi$;
- $S, q \models \langle\langle\Gamma\rangle\rangle \bigcirc \varphi$ iff there exists a set of strategies F_Γ, one for each $a \in \Gamma$, such that for all $\lambda \in out(q, F_\Gamma)$, we have $S, \lambda[1] \models \varphi$;
- $S, q \models \langle\langle\Gamma\rangle\rangle \Box \varphi$ iff there exists a set of strategies F_Γ, one for each $a \in \Gamma$, such that for all $\lambda \in out(q, F_\Gamma)$, we have $S, \lambda[u] \models \varphi$ for all $u \in \mathbb{N}$;
- $S, q \models \langle\langle\Gamma\rangle\rangle \varphi \mathcal{U} \psi$ iff there exists a set of strategies F_Γ, one for each $a \in \Gamma$, such that for all $\lambda \in out(q, F_\Gamma)$, there exists some $u \in \mathbb{N}$ such that $S, \lambda[u] \models \psi$, and for all $0 \leq v < u$, we have $S, \lambda[v] \models \varphi$;
- $S, q \models K_a\varphi$ iff for all q' such that $q \sim_a q'$: $S, q' \models \varphi$;
- $S, q \models E_\Gamma\varphi$ iff for all q' such that $q \sim_\Gamma^E q'$: $S, q' \models \varphi$;
- $S, q \models C_\Gamma\varphi$ iff for all q' such that $q \sim_\Gamma^C q'$: $S, q' \models \varphi$.

Before proceeding, we introduce some derived connectives: these include the remaining connectives of classical propositional logic ($\bot \hat{=} \neg\top$, $\varphi \to \psi \hat{=} \neg\varphi \vee \psi$ and $\varphi \leftrightarrow \psi \hat{=} (\varphi \to \psi) \wedge (\psi \to \varphi)$), together with some other useful connectives of temporal logic.

$$\begin{array}{rcl} \langle\langle\Gamma\rangle\rangle \Diamond \varphi & \hat{=} & \langle\langle\Gamma\rangle\rangle \top \mathcal{U} \varphi \\ \langle\langle\Gamma\rangle\rangle \Box^+ \varphi & \hat{=} & \langle\langle\Gamma\rangle\rangle \bigcirc \langle\langle\Gamma\rangle\rangle \Box \varphi \\ \langle\langle\Gamma\rangle\rangle \Diamond^+ \varphi & \hat{=} & \langle\langle\Gamma\rangle\rangle \bigcirc \langle\langle\Gamma\rangle\rangle \Diamond \varphi \end{array}$$

As well as asserting that some collection of agents is able to bring about some state of affairs, we can use the dual "$[\![\ldots]\!]$" to express the fact that a group of agents cannot *avoid* some state of affairs. Thus $[\![\Gamma]\!] \Diamond \varphi$ expresses the fact that the group Γ cannot cooperate to ensure that φ will never be true; the remaining agents in the system have a collection of strategies such that, if they follow them, φ may eventually be achieved. Formally, $[\![\Gamma]\!] \bigcirc \varphi$ is defined as an abbreviation for $\neg\langle\langle\Gamma\rangle\rangle \bigcirc \neg\varphi$, while $[\![\Gamma]\!] \Diamond \varphi$ is defined as an abbreviation for $\neg\langle\langle\Gamma\rangle\rangle \Box \neg\varphi$, and so on. Finally, we generally omit set brackets inside cooperation modalities, (i.e., writing $\langle\langle a, b\rangle\rangle$ instead of $\langle\langle\{a, b\}\rangle\rangle$), and we will usually write $\langle\langle\rangle\rangle$ rather than $\langle\langle\emptyset\rangle\rangle$.

Some Informal Examples

To get a feel for the kinds of properties that may be expressed using ATEL, consider the following informal examples of ATEL formulae.

$$\langle\langle m, w\rangle\rangle \Diamond readerIsBored$$

This formula asserts that Mike and Wiebe have a collective strategy for ensuring that, eventually, the reader is bored. That is, they can “force” the reader to be bored if they so choose, in the sense that we say a player in a game is able to “force a win”. The following says that m and w cannot ensure that the reader is always excited.

$$\neg\langle\langle m, w\rangle\rangle\Box\, excited$$

We can also write this previous example as follows.

$$[\![m, w]\!]\Diamond\neg excited$$

The next (optimistic!) example says that m and w can ensure that eventually, the reader has the capability to eventually understand.

$$\langle\langle m, w\rangle\rangle\Diamond\langle\langle reader\rangle\rangle\Diamond understand$$

By combining ATL cooperation modalities with knowledge modalities, we can express much richer properties. The following formula says that w can ensure that eventually, m knows that the Earth is round.

$$\langle\langle w\rangle\rangle\Diamond K_m earthIsRound$$

The final example — a formula scheme this time, rather than a formula — says that if m can cause w to know something, then w knows it already (in other words, m cannot tell w anything that he does not already know).

$$\langle\langle m\rangle\rangle\Diamond K_w\varphi \rightarrow K_w\varphi$$

4. Axioms for ATEL

The aim of this paper is not to give a complete axiomatisation of ATEL. In this section, we rather give some typical properties. Goranko [16] uses the axiomatisation for Pauly’s coalition logic [30] to obtain an axiomatisation for a fragment of ATL with only the temporal operator for tomorrow ($\bigcirc$), and some properties of ATL with the operator $\Box$ are given. But, to the best of our knowledge, there is as yet no completeness result for full ATL in the literature.

It should be clear that ATEL inherits the $S5$ axioms of normal modal logic for knowledge modalities, as well as the associated axioms for common knowledge and everyone knowing — as these axioms are by now well-known, we will not describe them here (see e.g., [19, 15] for a detailed account).

Also, at this stage of research, we have not investigated yet what kind of *interaction axioms* would appear in natural extensions of ATEL, and whether they would complicate a completeness proof. We will give examples of such interaction properties in Section 5.

An obvious first question is the extent to which CTL axioms transfer to ATL/ATEL (see e.g., [13, p.1040]).

Ax1	All validities of propositional logic
Ax2	$\langle\langle\Sigma\rangle\rangle\Diamond\varphi \leftrightarrow \langle\langle\Sigma\rangle\rangle(\top\,\mathcal{U}\,\varphi)$
Ax2b	$\langle\langle\emptyset\rangle\rangle\Box\varphi \leftrightarrow \neg\langle\langle\Sigma\rangle\rangle\Diamond\neg\varphi$
Ax3	$\langle\langle\emptyset\rangle\rangle\Diamond\varphi \leftrightarrow \langle\langle\emptyset\rangle\rangle(\top\,\mathcal{U}\,\varphi)$
Ax3b	$\langle\langle\Sigma\rangle\rangle\Box\varphi \leftrightarrow \neg\langle\langle\Sigma\rangle\rangle\Diamond\neg\varphi$
Ax4	$(\langle\langle\Sigma\rangle\rangle\bigcirc\varphi \vee \langle\langle\Sigma\rangle\rangle\bigcirc\psi) \leftrightarrow \langle\langle\Sigma\rangle\rangle\bigcirc(\varphi\vee\psi)$
Ax5	$\langle\langle\emptyset\rangle\rangle\bigcirc\varphi \leftrightarrow \neg\langle\langle\Sigma\rangle\rangle\bigcirc\neg\varphi$
Ax8	$\langle\langle\emptyset\rangle\rangle\bigcirc\top \wedge [\![\Sigma]\!]\bigcirc\top$
Ax9	$\langle\langle\emptyset\rangle\rangle\Box(\chi\to(\neg\psi\wedge\langle\langle\Sigma\rangle\rangle\bigcirc\chi)) \to (\chi\to\neg\langle\langle\emptyset\rangle\rangle(\varphi\,\mathcal{U}\,\psi))$
Ax9b	$\langle\langle\emptyset\rangle\rangle\Box(\chi\to(\neg\psi\wedge\langle\langle\Sigma\rangle\rangle\bigcirc\chi)) \to (\chi\to\neg\langle\langle\emptyset\rangle\rangle\Diamond\psi)$
Ax10	$\langle\langle\emptyset\rangle\rangle\Box(\chi\to(\neg\psi\wedge(\varphi\to\langle\langle\emptyset\rangle\rangle\bigcirc\chi))) \to (\chi\to\neg\langle\langle\Sigma\rangle\rangle(\varphi\,\mathcal{U}\,\psi))$
Ax10b	$\langle\langle\emptyset\rangle\rangle\Box(\chi\to(\neg\psi\wedge\langle\langle\emptyset\rangle\rangle\bigcirc\chi)) \to (\chi\to\neg\langle\langle\Sigma\rangle\rangle\Diamond\psi)$
Ax11	$\langle\langle\emptyset\rangle\rangle\Box(\varphi\to\psi) \to (\langle\langle\Sigma\rangle\rangle\bigcirc\varphi\to\langle\langle\Sigma\rangle\rangle\bigcirc\psi)$

We now discuss how these axioms generalise to properties of arbitrary coalitions Γ, where Γ ranges over 2^{Σ}. Ax2 and Ax3 can be combined in the general:

$$\text{Ax2\&3}\quad \langle\langle\Gamma\rangle\rangle\Diamond\varphi \leftrightarrow \langle\langle\Gamma\rangle\rangle(\top\,\mathcal{U}\,\varphi)$$

Several axioms are not sound for arbitrary $\Gamma\subseteq\Sigma$. An example of such an axiom is the generalisation of Ax2b to Ax2b′: $[\![\Sigma\setminus\Gamma]\!]\Box\varphi \leftrightarrow \neg\langle\langle\Gamma\rangle\rangle\Diamond\neg\varphi$, which is equivalent to $\neg\langle\langle\Sigma\setminus\Gamma\rangle\rangle\Box\neg\varphi \leftrightarrow \neg\langle\langle\Gamma\rangle\rangle\Diamond\neg\varphi$, or, as a scheme, to $\langle\langle\Sigma\setminus\Gamma\rangle\rangle\Box\psi \leftrightarrow \langle\langle\Gamma\rangle\rangle\Diamond\psi$, which is obviously not valid. Also, the following generalisation of Ax3b is not valid:

$$\langle\langle\Gamma\rangle\rangle\Box\varphi \leftrightarrow \neg\langle\langle\Sigma\setminus\Gamma\rangle\rangle\Diamond\neg\varphi \tag{i}$$

It is the "$\leftarrow$" direction that is invalid here: if a subgroup $\Sigma\setminus\Gamma$ does not have a strategy to ensure that sometime the system is not okay, it does not mean that Γ can ensure that the system is always okay. Note that (Ax4) is not sound for arbitrary $\Gamma\subseteq\Sigma$: see Example 1 below. A similar remark holds for (Ax5): for subgroups Γ the implication $\neg\langle\langle\Gamma\rangle\rangle\bigcirc\neg\varphi\to\langle\langle\Sigma\setminus\Gamma\rangle\rangle\bigcirc\varphi$

is not sound (for instance, let $\Sigma = \{a_1, a_2\}$, $\Gamma = \{a_1\}$ and let φ be the statement that a_1 and a_2 throw the same number with a dice. The we have $\neg\langle\langle\Gamma\rangle\rangle\bigcirc\neg\varphi$ but also $\neg\langle\langle\Sigma\setminus\Gamma\rangle\rangle\bigcirc\varphi$). Also note that the axioms (Ax6) and (Ax7) of [13, p.1040] have no counterpart in ATEL. Ax8 can be reduced to simply $\langle\langle\emptyset\rangle\rangle\bigcirc\top$ by using the supergroup property below.

Thus, new axioms are:

Seriality	$\langle\langle a\rangle\rangle\bigcirc\top$
Supergroup($\bigcirc$)	$\langle\langle\Gamma\rangle\rangle\bigcirc\varphi \to \langle\langle\Gamma\cup\{a\}\rangle\rangle\bigcirc\varphi$
Complement($\bigcirc$)	$\langle\langle\Gamma\rangle\rangle\bigcirc\varphi \to \neg\langle\langle\Sigma\setminus\Gamma\rangle\rangle\bigcirc\neg\varphi$
Complement($\Box$)	$\langle\langle\Gamma\rangle\rangle\Box\varphi \to \neg\langle\langle\Sigma\setminus\Gamma\rangle\rangle\neg\Diamond\varphi$
Ax4a	$(\langle\langle\Sigma\rangle\rangle\bigcirc\varphi \vee \langle\langle\Sigma\rangle\rangle\bigcirc\psi) \to \langle\langle\Sigma\rangle\rangle\bigcirc(\varphi\vee\psi)$
Ax5b	$[\![\Sigma]\!]\bigcirc\varphi \to \neg\langle\langle\Sigma\rangle\rangle\bigcirc\neg\varphi$

In fact, the Supergroup($\bigcirc$) and Complement($\bigcirc$) axioms follow from the following, stronger axiom.

$$\langle\langle\Gamma_1\rangle\rangle\bigcirc\varphi_1 \wedge \langle\langle\Gamma_2\rangle\rangle\bigcirc\varphi_2 \to \langle\langle\Gamma_1\cup\Gamma_2\rangle\rangle\bigcirc(\varphi_1\wedge\varphi_2) \quad (\Gamma_1\cap\Gamma_2=\emptyset) \tag{ii}$$

The only two obvious generalisations of Ax9 and Ax9b are those in which we replace $\langle\langle\Sigma\rangle\rangle$ by an arbitrary $\langle\langle\Gamma\rangle\rangle$, but such a positive occurence of $\langle\langle\Sigma\rangle\rangle$ may already be replaced by $\langle\langle\Gamma\rangle\rangle$ by the supergroup property, mentioned above. A similar remark applies to Ax10 and Ax10b. Let us finally observe that Ax11 can be generalised to

$$\text{Ax11'} \quad \langle\langle\emptyset\rangle\rangle\Box(\varphi\to\psi) \to (\langle\langle\Gamma\rangle\rangle\bigcirc\varphi \to \langle\langle\Gamma\rangle\rangle\bigcirc\psi)$$

but not to

$$\langle\langle\Gamma\rangle\rangle\Box(\varphi\to\psi) \to (\langle\langle\Gamma\rangle\rangle\bigcirc\varphi \to \langle\langle\Gamma\rangle\rangle\bigcirc\psi) \tag{iii}$$

which is seen as follows; if the coalition Γ has a stratey to guarantee that $(\varphi\to\psi)$ is always true, then, if it has at the same time a strategy (which may well be different) to make φ true in the next state, it does not necessarily mean it has a strategy to guarantee ψ in the next state.

It is worth asking whether there is a property in the language reflecting the requirement that the intersection of all the choices of the agents is a singleton. Where the formula $\Diamond\varphi \leftrightarrow \Box\varphi$ in modal logic expresses that there is exactly one successor, the fact that the whole group Σ determines a unique outcome is not captured by:

$$\langle\langle\Sigma\rangle\rangle\bigcirc\varphi \leftrightarrow [\![\Sigma]\!]\bigcirc\varphi \tag{iv}$$

Instead, we need the following two conditions. The first is that the intersection of choices is non-empty, which is given by the following axiom.

$$\langle\!\langle \Sigma \rangle\!\rangle \bigcirc \top \tag{v}$$

(Remarkably, (v) already follows from the seemingly weaker Seriality axiom: $\langle\!\langle a \rangle\!\rangle \bigcirc \top$.) The second property we place on system transition functions (i.e., that, if all the agents have chosen, there is at most one successor), is trickier. First note that (vi) is not valid (Γ is an arbitrary set of agents).

$$\langle\!\langle \Gamma \rangle\!\rangle \bigcirc (\varphi \vee \psi) \rightarrow (\langle\!\langle \Gamma \rangle\!\rangle \bigcirc \varphi \vee \langle\!\langle \Gamma \rangle\!\rangle \bigcirc \psi) \tag{vi}$$

The following example illustrates why.

EXAMPLE 1. Let $\Sigma = \{a_1, a_2\}$; let $\delta(q, a_1) = \{\{v_1\}, \{v_2\}\}$ and $\delta(q, a_2) = \{\{v_1, v_2\}\}$. Let p be only true in v_1 and r only in v_2. Then we have in $q \models \langle\!\langle a_2 \rangle\!\rangle \bigcirc (p \vee q)$, but $q \not\models \langle\!\langle a_2 \rangle\!\rangle \bigcirc p \vee \langle\!\langle a_2 \rangle\!\rangle \bigcirc q$.

Instead, the axiom we are after seems to be (vii).

$$\langle\!\langle \Sigma \rangle\!\rangle \bigcirc (\varphi \vee \psi) \rightarrow (\langle\!\langle \Sigma \rangle\!\rangle \bigcirc \varphi \vee \langle\!\langle \Sigma \rangle\!\rangle \bigcirc \psi) \tag{vii}$$

Still other properties relate $\langle\!\langle \Gamma \rangle\!\rangle$, $[\![\Gamma]\!]$, Γ and $\Sigma \setminus \Gamma$. The next two properties seem to hold for any temporal formula ψ, rather than for just $\bigcirc \varphi$:

$$\langle\!\langle \Gamma \rangle\!\rangle \bigcirc \varphi \rightarrow [\![\Sigma \setminus \Gamma]\!] \bigcirc \varphi \tag{viii}$$

$$\langle\!\langle \Gamma \rangle\!\rangle \bigcirc \varphi \rightarrow \langle\!\langle \Gamma' \rangle\!\rangle \bigcirc \varphi \qquad \text{(where } \Gamma \subseteq \Gamma'\text{)} \tag{ix}$$

As a corollary, we have the following.

$$\langle\!\langle \emptyset \rangle\!\rangle \bigcirc \varphi \rightarrow \langle\!\langle \Gamma \rangle\!\rangle \bigcirc \varphi \qquad \text{(for any } \Gamma \subseteq \Sigma\text{)} \tag{x}$$

Distinguishing Subsystems

In [3], Alur and colleagues identify a number of distinct sub-systems of ATL. We now consider axioms that distinguish these subsystems.

Turn-based Systems

The first subsystem we consider is the *turn-based synchronous* subsystem. In such a system, at every transition there is just one agent that is permitted to make a choice (and hence determine the future). This would model many "classical" games, in which players alternate to move. A characterizing property of a turn-based synchronous ATS is the following:

$$\langle\langle\Sigma\rangle\rangle \bigcirc \varphi \rightarrow \bigvee_{a\in\Sigma} \langle\langle a\rangle\rangle \bigcirc \varphi \tag{xi}$$

Property (xi) is the translation into ATEL of what Pauly calls "dictatorship" [30], and says that, if the grand coalition can establish φ to hold tomorrow, then there is an agent who can achieve this on his own already. Note that (xi) is equivalent to $\bigvee_{a\in\Sigma}(\langle\langle\Sigma\rangle\rangle \bigcirc \varphi \rightarrow \langle\langle a\rangle\rangle \bigcirc \varphi)$ and, since (ix) guarantees us that since we have in arbitrary ATEL systems $(\langle\langle\Sigma\rangle\rangle \bigcirc \varphi \leftarrow \langle\langle a\rangle\rangle \bigcirc \varphi)$, we know that in turn-based synchronous systems we have the following.

$$\bigvee_{a\in\Sigma} (\langle\langle a\rangle\rangle \bigcirc \varphi \leftrightarrow \langle\langle\Sigma\rangle\rangle \bigcirc \varphi) \tag{xii}$$

In fact, we can write (xii) to $\bigvee_{a\in\Sigma}(\neg\langle\langle a\rangle\rangle \bigcirc \varphi \leftrightarrow \neg\langle\langle\Sigma\rangle\rangle \bigcirc \varphi)$, then apply Ax5 to this scheme to obtain $\bigvee_{a\in\Sigma}([\![a]\!] \bigcirc \psi \leftrightarrow [\![\Sigma]\!] \bigcirc \psi)$. The latter property would read: "everything that the grand coalition cannot avoid is that which one of the agents cannot avoid".

Lock-step Systems

In a lock-step ATS, it is assumed that every state q is of the form $\langle q[a_1], \ldots, q[a_n]\rangle$, with $\{a_1, \ldots, a_n\} = \Gamma$. The idea is that every agent a_i has a local state, and in q this local state is $q[a_i]$. In a lock-step system, every agent a_i can only choose the effect on its own local state: agent a_i can only determine his next local state, i.e., every $Q_s \in \delta(q, a_i)$ is of the form $Q_s = \{q' \mid q'[a_i] = s\}$, where s is the local state that a_i can bring about. This kind of ATS becomes particularly interesting when adding knowledge to it, since one straightforward way to define the epistemic equivalence relations here is by $q \sim_{a_i} q \Leftrightarrow q[a_i] = q'[a_i]$, i.e., the agent knows exactly what its own state is. This brings us into the paradigm of *interpreted systems*, a notion of epistemic systems thoroughly studied in [15, 26]. A well-known property of such a system is for instance

$$C_\Sigma\varphi \rightarrow \langle\langle\rangle\rangle \Box C_\Sigma\varphi \tag{xiii}$$

saying that common knowledge is constant during every run in such a system where the whole state space is the cartesian product of the local state spaces (see also [27, p.61]).

In lock-step ATEL we also have $\langle\langle a\rangle\rangle \bigcirc \varphi \rightarrow \langle\langle a\rangle\rangle \bigcirc K_a\varphi$. This property is closely related to that of *perfect recall*, which in ATEL could read

$K_a\langle\langle a\rangle\rangle \bigcirc \varphi \rightarrow \langle\langle a\rangle\rangle \bigcirc K_a\varphi$. Restricting oneself to runs that satisfy such properties is a route undertaken by van de Meyden and Vardi in [32], when they perform *synthesis with incomplete information.*

Turn-based Asynchronous Systems

A turn-based asynchronous system is much like a turn based synchronous ATS, but now, there is a special agent, the scheduler *sch*, who determines which agent is to move, and hence the "dictator" is not determined by the state.

Without going into the full details here, we state two properties of such a system.

$$\langle\langle\Sigma\rangle\rangle \bigcirc \varphi \rightarrow \bigvee_{sch \neq a \in \Sigma} \langle\langle sch, a\rangle\rangle \bigcirc \varphi \tag{xiv}$$

$$\langle\langle sch\rangle\rangle \bigcirc \varphi \rightarrow \bigvee_{sch \neq a \in \Sigma} [\![a]\!] \bigcirc \varphi \tag{xv}$$

Property (xiv) expresses that if the grand coalition can guarantee that tomorrow φ holds, it can be achieved by *sch* choosing some agent that can guarantee φ. Then (xv) expresses the following. All that agent *sch* can do is to schedule one of the other agents. So if *sch* wants to guarantee φ in the next state, he should not just select an agent a that can guarantee φ (since a might, when given the turn, choose another alternative), but *sch* should pick an agent that *cannot avoid* φ.

ATEL*

CTL* is a well-known, and much more expressive variant of CTL [13]. Put somewhat crudely, CTL* allows path quantifiers and tense modalities to be arbitrarily intermingled in formulae, rather than requiring that tense modalities be immediately preceded by a path quantifier, as in CTL. Thus, for example, $\mathsf{A}\Diamond\Box off$ is a formula of CTL* while it is not a formula of CTL. It is not hard to see that CTL* is strictly more expressive than CTL: there are properties of temporal tree structures that can be expressed using CTL*, which cannot be expressed using CTL [14]. Alur and colleagues defined a variant of ATL which plays a role analogous to that played of CTL*: in particular, it allows tense modalities and cooperation modalities to be intermingled in formulae. Thus $\langle\langle 1\rangle\rangle\Diamond\Box off$ is a formula of ATL*, but not of ATL. We will not give a full definition of ATL*, or its rather obvious generalisation ATEL*; see [3] for details.

The first obvious property we get is that every group Γ can obtain the current state: one has nothing to do to obtain it.

$$\varphi \rightarrow \langle\langle\Gamma\rangle\rangle\varphi \qquad \text{(for state formulae } \varphi) \tag{xvi}$$

Moreover, agents *cannot alter* the current state: it has been obtained already. Thus, we also have the following.

$$\varphi \rightarrow [\![\Gamma]\!]\varphi \qquad \text{(for state formulae } \varphi) \tag{xvii}$$

As a scheme, (xvii) is equivalent to $\langle\langle\Gamma\rangle\rangle\varphi \rightarrow \varphi$; and so, despite the suggestive existential notation, $\langle\langle\cdot\rangle\rangle$ does not take you further in time. From (xvi) and (xvii) we obtain the following.

$$\varphi \leftrightarrow \vec{X}\varphi \qquad \text{(for state formulae } \varphi) \tag{xviii}$$

where $\vec{X}$ is an arbitrary sequence of $\langle\langle\Gamma\rangle\rangle$ and $[\![\Gamma']\!]$ prefixes, with arbitrary group variables Γ, Γ'.

5. Knowledge, Ability, and Cooperation

In this section, we demonstrate the expressive power and flexibility of ATEL, by showing how it allows us to capture the many complex and subtle ways in which knowledge, ability, and cooperation can interact.

Knowledge Preconditions

Performing actions and knowledge interfere in at least two ways: for some actions, in order to be able to do them properly, some knowledge is required, and, on the other hand, actions may add to an agent's knowledge. A major ongoing research problem in AI planning is that of correctly formulating *knowledge preconditions* for actions and plans [1]. Intuitively, a knowledge pre-condition for a particular plan is the information that an agent must have in order to be able to successfully carry this plan out. Arguably the best known, and most influential attempt to develop a formalism for knowledge pre-conditions was that by Moore [28]. Moore used a formalism that combined aspects of both epistemic logic [15, 27] and dynamic logic [21] in order to capture what is needed to be known in order to carry out a plan. (Many formalisms have subsequently been developed for this problem — see [37] for a survey.)

We can formulate knowledge pre-conditions quite naturally using ATEL and its variants, and the cooperation modality naturally and elegantly allows us to consider knowledge pre-conditions for *multi*-agent plans. The requirement that, in order for an agent a to be able to eventually bring about state of affairs φ, it must know ψ, might, as a first attempt, be specified in ATEL as:

$$\langle\langle a\rangle\rangle\Diamond\varphi \rightarrow K_a\psi \tag{xix}$$

Formula (xix) intuitively says that knowing ψ is a *necessary* requirement for having the ability to bring about φ. However, this requirement is too strong. For instance, in order to be able to ever open the safe, I don't necessarily in general have to know the key right *now*; it sufficient that I know it when I am going to open it. A slightly better formulation might therefore be the following.

$$\langle\langle a\rangle\rangle\bigcirc\varphi \rightarrow K_a\psi \tag{xx}$$

If (xx) is given as an overall constraint of the system, it helps the agent to realize that he has to possess the right knowledge in order to achieve φ. But taken as a local formula, it does not tell us anything about what the agent should know if he wants to bring about φ the day after tomorrow, or "sometime" for that matter. Taken as a local constraint, a necessary knowledge condition to bring about φ might be

$$(\neg\langle\langle a\rangle\rangle\bigcirc\varphi)\,\mathcal{U}\,K_a\psi \tag{xxi}$$

Property (xxi) expresses that our agent is not able to open the safe until he knows its key. Of course, this is still a bit weak: one wants to express that as soon as a *does* know the key, he in fact *is* able to open the safe. The fact that knowing ψ is *sufficient* to be able to bring about φ is captured by the converse of the scheme (xix):

$$K_a\psi \rightarrow \langle\langle a\rangle\rangle\Diamond\varphi \tag{xxii}$$

This scheme presupposes that knowledge (of ψ) is not perishable, e.g., keys for safes do not change. A more cautious condition might be:

$$K_a\psi \rightarrow \langle\langle a\rangle\rangle\bigcirc\varphi \tag{xxiii}$$

Indeed, one might show that under the assumption that knowledge of a key is non-perishable ($K_a\psi \rightarrow \bigcirc K_a\psi$), equation (xxiii) implies (xxii).

Another example of such an ability is (xxiv), expressing that if Bob knows that the combination of the safe is s, then he is able to open it (o), as long as the combination remains unchanged.

$$K_b(c = s) \rightarrow \langle\langle b\rangle\rangle(\langle\langle b\rangle\rangle\bigcirc o)\,\mathcal{U}\,\neg(c = s) \tag{xxiv}$$

We have not yet explored knowledge-dependent abilities of groups. The simplest case would be $(K_a\varphi_a \wedge K_b\varphi_b) \rightarrow \langle\langle a, b\rangle\rangle\Diamond\psi$.

Resource-bounded Reasoners

One may also use ATEL-formulas to model limited reasoners, i.e., reasoners that do not obey all the $S5$-axioms in a blow, but who can approximate them over time (here, we are assuming of course that an agent's knowledge accessibility relation $\sim_a$ is not an equivalence relation).

$$K_a\psi \rightarrow \langle\langle a\rangle\rangle \bigcirc K_a K_a \psi \tag{xxv}$$

In fact, such reasoning bounded rules are proposed in [17], but there, the introspective step is done "automatically" during a time step; in ATEL, the agent may decide to apply it only when needed.

Cryptography and Security

Schemes like the above allow representing time consuming *computations* by the agent needed to acquire or use his knowledge. An example of the latter would be in cryptographic protocols, where the agents need to compute or recognize a key in order to decrypt or authorize a message. In such logics, one usually distinguishes between receiving a message (denoted by `sees`) and knowing its contents [9]. Let us denote by $\{msg\}_{S_{ab}}$ that the message msg is encrypted by a key S_{ab} only known by a and b, then a decryption rule would typically express that if agent a receives a message $\{msg\}_{S_{ab}}$ and a knows that S_{ab} is the shared key of a and b, then a can use this key to encrypt the whole message, and safely conclude that b said msg:

$$\texttt{sees}_a\{msg\}_{S_{ab}} \wedge K_a(\textit{is-key}\langle(a,b),S_{ab}\rangle) \rightarrow \langle\langle a\rangle\rangle \bigcirc K_a said_b msg \tag{xxvi}$$

The logic of [9] then has a rule to conclude when K_aK_bmsg holds: the message msg must be fresh, that is, may not be seen in the current run of the protocol (we feel that ATEL is especially useful when reasoning about runs in protocols, in particular, it will be possible to denote that a message has not been seen in a protocol yet):

$$(K_a fresh(msg) \wedge K_a said_b msg) \rightarrow K_a K_b msg \tag{xxvii}$$

Communication

We now consider some properties that are important when agents communicate, gather information or, more generally, bring about knowledge. How can a bring about that b learns something, i.e., how to guarantee $\langle\langle a\rangle\rangle\Diamond K_b\varphi$? There are many issues at stake here, dealing with the reliability of channels, the communication protocol used, the language that may be used in communication, etc. Let us have a look at a simple example, depicted in Figure 1. The left part of the picture is an $S5_2$ model: reflexive and transitive arrows are not drawn, the dotted lines are the accessibilities of agent 1, the straight ones are those of agent 2. In state s, we have

$$s \models \neg K_1 p \wedge \neg K_1 \neg p \wedge K_1(K_2 p \vee K_2 \neg p)$$

which says that 1 does not know whether p, but 1 knows that 2 knows whether p. Thus, 1 asking 2 whether p seems a reasonable thing to do. This question is modeled as $\delta(s,1) = \{\{s_1, s_2\}\}$. One easily sees that this gives us that $\langle\langle 1\rangle\rangle \bigcirc (K_1 p \vee K_1 \neg p)$ is true in s, that is, agent 1 can find out whether p is true.

Thus, the speech act of a yes/no question φ? by agent a is nothing more than a opening up his current state s in two alternatives, one in which he knows φ, and one in which he knows $\neg\varphi$. Since we know that the intersection of the results of all choices by the agents will be a unique state, we know that after the other agents have completed their moves, agent a knows whether φ.

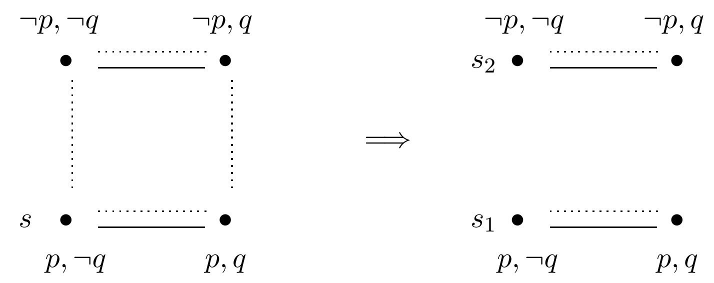

Figure 1. Asking a question

In the particular example here, if agent 2 is truthful, he will select s_1, that is, $\delta(s,2) = \{\{s_1\}\}$.

Note that we made a lot of assumptions in this example: the two agents are willing to cooperate, and the communication is known to be reliable, since in fact we have that $s \models \langle\langle 1,2\rangle\rangle \bigcirc C_{\{1,2\}} p$.

Let us now briefly examine a number of properties that may be important in systems of communicating agents.

FS	$K_a\varphi \to \langle\langle a\rangle\rangle \Diamond K_a K_b \varphi$
NH	$K_a\varphi \to [\![a]\!] \Diamond K_a K_b \varphi$
TT	$\neg K_a p \to [\![a]\!] \bigcirc \neg K_b K_a p$
MO	$K_a\varphi \to [\![\Gamma]\!] \Box K_a \varphi$

The first principle, Freedom of Speech (FS), says that an agent can always tell the truth to other agents. The second principle, No Hiding (NH), ensures that agents cannot but tell everything they know eventually to others. In many cases, this makes sense for specific facts p only. Next, Telling the Truth (TT): this says that if agent a does not know p now, he cannot avoid that b will not know that a knows p tomorrow. This property is restricted to propositional formulas p, for the following reason. Suppose we have that agent a does not know p now; then he cannot know that b knows p, hence we have $\neg K_a K_b p$. Suppose now b makes a public announcement p, then we have that, in the next state, $C_\Sigma p$ will be true, and in particular $K_b K_a K_b p$, which gives a violation of TT, since we have a situation in which $\neg K_a K_b p \wedge \langle\langle\Sigma\rangle\rangle \bigcirc K_b K_a K_b p$ holds! Finally, Monotony (MO) says that knowledge can only decrease. This is not a realistic property for arbitrary formulas either, for instance knowledge about $\bigcirc p$ generally does not persist, and this property is also seen to be undesirable if one takes $\varphi = \neg K_a p$. The validity of $(K_a \neg K_a p \leftrightarrow \neg K_a p)$ together with MO would then yield $\neg K_a p \rightarrow [\![\Gamma]\!] \Box \neg K_a p$ which would imply that a can never learn p.

As well as monotonicity of knowledge, persistence of ignorance may also be an issue, both as a pre- and as a post-condition. In security protocols for instance, where agents a and b share some common secret (a key S_{ab} for instance), what one typically wants is (xxviii), expressing that a can send private information to b, without revealing the message to another agent c:

$$K_a\varphi \wedge \neg K_b\varphi \wedge \neg K_c\varphi \wedge \langle\langle a,b\rangle\rangle \bigcirc (K_a\varphi \wedge K_b\varphi \wedge \neg K_c\varphi) \qquad \text{(xxviii)}$$

Let us conclude this subsection by looking at a typical group way of bringing about knowledge. Recall that distributed knowledge is the knowledge that is implicitly present in the group: the idea is that it is the knowledge that would become explicit if all the agents could communicate. More precisely, $D_\Gamma\varphi$ holds in group Γ iff there are $\psi_1, \ldots, \psi_n$ ($n = |\Gamma|$), such that one has $(K_1\psi_1 \wedge \cdots \wedge K_n\psi_n)$ and $\models (\psi_1 \wedge \cdots \wedge \psi_n) \rightarrow \varphi$ (cf. [15, 23]). Let us again look at a simple example, depicted in Figure 2. Again, 1 owns the dotted arrows, and 2 the straight ones. The valuations of the worlds are not repeated. Note that in s, we have

$$s \models K_1 p \wedge K_2(p \rightarrow q) \wedge K_1(q \rightarrow r)$$

and hence, in s, proposition r is distributed knowledge, i.e., $s \models D_{\{1,2\}} r$.

Figure 2 shows how the agents can establish that $E_{\{1,2\}} r$: agent 1 first communicates p to 2, who answers with $p \rightarrow q$ to which 1 concludes with $q \rightarrow r$. Thus, we obtain:

$$s \models \langle\langle 1,2\rangle\rangle \Diamond E r$$

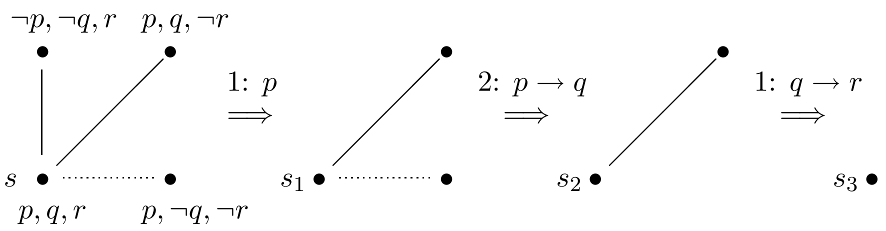

Figure 2. Making distributed knowledge explicit

Of course, semantically speaking, one could obtain the end result in one blow (i.e., one agent taking the system to s_3 in one step), but then the question arises whether there is a natural intuition for such an action. What we implicitly assumed is that agents can only communicate facts that they know:

$$\text{CF} \quad K_a\varphi \rightarrow \langle\langle a\rangle\rangle \bigcirc K_a K_b \varphi \wedge (\langle\langle a\rangle\rangle \bigcirc K_b\varphi \rightarrow (K_a\varphi \vee K_b\varphi))$$

Now, consider the following property.

$$\text{DE} \quad D_\Gamma\varphi \rightarrow \langle\langle \Gamma\rangle\rangle \Diamond E_\Gamma\varphi$$

DE states that agents can always cooperate to make distributed knowledge explicit (so that they never have to agree to disagree). If we assume CF, then from [23] we distill that DE is guaranteed iff the model is finite and for every state, a unique formula must be true. Axioms closely related to DE were used to characterise the soundness and completeness of distributed problem solving systems in [34].

Common knowledge C_Γ of a group Γ is also important. In particular, one is interested in conditions that are sufficient to ensure that

$$C_\Gamma \langle\langle \Gamma\rangle\rangle T\varphi \qquad (T \text{ a temporal operator}) \qquad \text{(xxix)}$$

Schema (xxix) expresses that it is common knowledge in the group Γ that it can bring about (next, or sometime, or always) φ. Note that this is the kind of scenario made famous in the *coordinated attack problem*, and the associated negative result, to the effect that, if message delivery is not guaranteed, then no amount of message exchange will be sufficient to establish common knowledge [18]. In our case, it is not immediately clear that we have such a negative result about obtaining common knowledge, since it seems we can model actions stronger than communication. For instance, we may have knowledge-producing actions, and also common-knowledge producing actions, like making an announcement. If a can make an announcement

p, he can choose a set of worlds in which the transitive closure of all the accessibility relations only leads to p-worlds.

As a simple example, suppose we have three agents, of which agent 1 knows whether p, i.e., $K_1 p \vee K_1 \neg p$, and this is common knowledge; it is also common knowledge that 1 always tells the truth. Now, given that 1 knows p, we can model that 1 can tell the truth only to 2, or to 2 and 3 separately or he can announce p in public:

$$\langle\langle 1 \rangle\rangle \bigcirc [(K_2 p \wedge \neg K_3 p) \vee (K_2 p \wedge K_3 p \wedge \neg C_{\{2,3\}} p) \vee (C_{\{2,3\}} p)]$$

Games and Knowledge

The relevance of ATEL goes much further than communication. In [11], *Knowledge Games* are investigated as a particular way of learning in multiagent systems. Epistemic updates are interpreted in a simple card game, where the aim of the player is to find out a particular deal d of cards. Having a winning strategy then easily translates into

$$d \to \langle\langle a \rangle\rangle \bigcirc (K_a d \wedge \bigwedge_{a \neq b} \neg K_b d) \qquad \text{(xxx)}$$

Of course, when agents make strategic choices, both epistemic pre- and post-conditions are at stake: a rational agent bases his choices upon his knowledge, and will typically try to maximize his own knowledge, at the same time minimize that of his competitors.

A reasonable constraint on rational agents is that they make their choices depending on their knowledge. This requires first of all that an agent a must be aware of the choice he has which, in its turn, assumes that he has the same choices in all $\sim_a$ equivalent states. This leads to the semantic requirement (xxxi):

$$q \sim_a q' \Rightarrow \delta(q, a) = \delta(q', a) \qquad \text{(xxxi)}$$

This gives us the following syntactic property.

$$\text{AW} \quad \langle\langle a \rangle\rangle T \varphi \leftrightarrow K_a \langle\langle a \rangle\rangle T \varphi$$

However, this does not guarantee yet that the agent makes the same decision in $\sim_a$-equivalent states. For this, we have to stipulate the following semantic property.

$$q \sim_a q' \Rightarrow f_a(\lambda \cdot q) = f_a(\lambda \cdot q') \qquad \text{(xxxii)}$$

It seems that this property has no syntactic counterpart in ATEL: although we are able to express that agent a *can* bring about φ, we cannot say that he *will* do so.

6. Model Checking for ATEL

It is well-known that the branching temporal logic CTL lacks expressive power — fairness, for example, cannot be expressed in "vanilla" CTL (see e.g., [33] for a recent discussion on the relative merits of CTL *versus* other temporal logics). What makes CTL so attractive from the point of view of formal methods is that the *model checking* problem for CTL is computationally cheap: given a CTL model M of size m and a CTL formula φ of size ℓ, the problem of checking whether or not φ is valid in M can be solved in time $O(m\ell)$. This has made it possible to implement efficient, industrial strength formal verification tools for checking whether a given finite state system satisfies a CTL specification [10]. The attractive computational properties of CTL are known to carry across to the alternating temporal logic of Alur et al [3]. The fact that model checking for alternating temporal logic is tractable is particularly intriguing because this problem generalises several other interesting problems of interest. For example, the *realizability* problem —showing that it is possible to implement a system *sys* that satisfies a particular specification φ— involves model checking the formula $\langle\langle sys \rangle\rangle\varphi$ in a "maximal" model, which encodes all possible input/output relations. In this section, we show that the tractability of model checking for alternating temporal logic carries over to ATEL: we present a (deterministic) symbolic model algorithm for ATEL that runs in time polynomial in the size of the formula and the size of the system begin checked. The core of this algorithm is given in Figure 4: the function *eval*(...) takes as input a formula φ of ATEL and an alternating epistemic transition system S, and returns as output the set of states in S in which the formula is satisfied.

The *eval*(...) function is recursive, and makes use of several subsidiary definitions:

- The function $pre : 2^{\Sigma} \times 2^{Q} \to 2^{Q}$, which takes as input a set of agents Γ and a set of states Q_1 and returns as output the set of all states Q_2 such that when the system is in one of the states in Q_2, the agents Γ can cooperate and force the next state to be one of Q_1.
- The function $img : Q \times 2^{Q \times Q} \to 2^{Q}$, which takes as input a state q and a binary relation $R \subseteq Q \times Q$, and returns the set of states that are accessible from q via R. That is, $img(q, R) = \{q' \mid qRq'\}$.

Notice that for any given inputs, both of these functions may be easily computed in time polynomial in the size of the inputs and the structure against which they are being computed; the *pre* function involves a fixed point computation that may be carried out in linear time [3]. Given this, our main results with respect to model checking are as follows.

THEOREM 1. *The algorithm given in Figure 4 terminates and is correct, in the sense that it returns the set of states in which the input formula is satisfied.*

PROOF. We only do the non-trivial case $\varphi = \langle\langle\Gamma\rangle\rangle\Box\psi$: we have to show that with this input, the set Q_1 will eventually contain the set of states satisfying this formula. Let us say that in state q group Γ can maintain ψ for n steps, notation $q \models\uparrow^n (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$, $n \geq -1$, if the following holds:

- $q \models \top$, in case $n = -1$;
- $q \models \psi$, in case $n = 0$;
- $q \models \langle\langle\Gamma\rangle\rangle \bigcirc (\uparrow^{n-1} (\langle\langle\Gamma\rangle\rangle \bigcirc \psi) \wedge \psi) \wedge \psi$, else.

Intuitively, $\uparrow^n (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$ is true in q if ψ is true in q, and the group Γ has a strategy to maintain the truth of ψ for at least n steps. For convenience, for any $n \geq -1$, and a given system AETS, let Γ^n_ψ be the set of states $q \in Q$ for which $\uparrow^n (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$ is true. Note that for all n, $\Gamma^{n+1}_\psi \subseteq \Gamma^n_\psi$.

To prove the procedure correct, we use the notation of Hoare triples. A triple $\{P\}S\{R\}$ means that, if precondition P is true before statement S is executed, then afterwards R will hold. An *invariant* of a while-loop is a statement that is true before the while-loop is entered and after every execution of the loop. The invariant of our while-loop will be the statement $I(Q_1, Q_2)$:

$$\exists n \geq -1 (Q_1 = \Gamma^n_\psi \ \& \ Q_2 = \Gamma^{n+1}_\psi) \tag{xxxiii}$$

To prove the while correct, we prove that I is indeed an invariant, and we also show that the loop terminates. We will do that by proving that there is an integer N that decreases every round of the loop. Finally, we show that, if the loop finishes and I is true, that then the desired result holds: $Q_1 = \{q | q \models \langle\langle\Gamma\rangle\rangle\Box\psi\}$. In the algorithm in Figure 3, comments ci give conditions between two statements; we will also do an assignment to the help-variable N.

Now, one easily sees that the while-loop terminates: the variable N decreases every loop. We next result proves correctness:

$$\exists n (\Gamma^n_\psi = Q_1 \ \& \ \Gamma^{n+1}_\psi = Q_2) \ \& \ Q_1 = Q_2 \Rightarrow Q_1 = \{q | q \models \langle\langle\Gamma\rangle\rangle\Box\psi\}$$

To prove the displayed formula, suppose the condition in the antecedent holds. The only non-trivial task is then to show that $Q_1 \subseteq \{q | q \models \langle\langle\Gamma\rangle\rangle\Box\psi\}$. So suppose $q \in Q_1$. Then $q \models\uparrow^{n+1} (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$. Pick a q' that is visited in

```
10.   elsif $\varphi = \langle\langle\Gamma\rangle\rangle\Box\psi$ then
11.      $Q_1 := Q$
12.      $Q_2 := Q_3 := eval(\psi, \langle\Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta\rangle)$
c1       $\{Q_1 = \Gamma_\psi^{-1} \& Q_2 = \Gamma_\psi^1$, hence $I(Q_1, Q_2)$ and $Q_2 \subseteq Q_1\}$
13.      while $Q_1 \not\subseteq Q_2$ do
c2       $\{\exists q(q \in Q_1, q \notin Q_2)$ Put $N := |Q_1|\}$
14.         $Q_1 := Q_1 \cap Q_2$
c3.         $\{\exists n(\Gamma_\psi^n = Q_1 = Q_2), Q_1 < N\}$
15.         $Q_2 := pre(\Gamma, Q_1) \cap Q_3$
c4.         $\{\exists n(\Gamma_\psi^n = Q_1 \ \& \ \Gamma_\psi^{n+1} = Q_2) \ (= I(Q_1, Q_2))\}$
16.      end-while
c5.      $\{\exists n(\Gamma_\psi^n = Q_1 \ \& \ \Gamma_\psi^{n+1} = Q_2) \ \& \ Q_1 = Q_2\}$
17.      return $Q_1$
34. end-function
```

Figure 3. A fragment of the model checking algorithm, with invariants.

the first step, when Γ maintains ψ in q. Then $q' \models\uparrow^n (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$. Hence, q' is also a member of Q_1, and hence $q' \models\uparrow^{n+1} (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$. But this means that $q \models\uparrow^{n+2} (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$. Obviously, for every k, $q \models\uparrow^{n+k} (\langle\langle\Gamma\rangle\rangle \bigcirc \psi)$, hence $q \models \langle\langle\Gamma\rangle\rangle\Box\psi$. ■

THEOREM 2. *The model checking problem for* ATEL *is* PTIME-*complete.*

PROOF. PTIME-hardness follows from the fact that ATEL subsumes ATL, for which the model checking problem is PTIME-complete [3]. Ignoring the obvious cases, consider where the input formula is of the form $\langle\langle\Gamma\rangle\rangle\Box\psi$. Here, the algorithm should return the set of states from which Γ can cooperate to ensure that ψ is always true. Inspecting the proof of Theorem 1, one notices that $\langle\langle\Gamma\rangle\rangle\Box\psi$ has a fixed point character: an axiom of ATEL is $\langle\langle\Gamma\rangle\rangle\Box\psi \leftrightarrow \psi \wedge \langle\langle\Gamma\rangle\rangle \bigcirc \langle\langle\Gamma\rangle\rangle\Box\psi$, so $\langle\langle\Gamma\rangle\rangle\Box\psi$ can be understood as a maximal solution to the equation

$$f(x) = \psi \wedge \langle\langle\Gamma\rangle\rangle \bigcirc x.$$

where f maps formulae of ATEL to formulae of ATEL. The loop in lines (10)–(17) of figure 4 essentially computes this solution. We showed in Theorem 1 that this while terminates. Note that the computation can in fact be done in polynomial time.

The cases where $\varphi = K_a\psi$ and $\varphi = C_\Gamma\psi$ simply involve the computation of the *img* function at most $|Q|$ times, each computation requiring time at most $O(|Q \times Q|^3)$. Hence the algorithm operates in PTIME. ■

```
function $eval(\varphi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$ returns a subset of $Q$
  if $\varphi \in \Pi$ then
    return $\{q \mid q \in \pi(\varphi)\}$
  elsif $\varphi = \neg\psi$ then
    return $Q \setminus eval(\psi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
  elsif $\varphi = \psi_1 \vee \psi_2$ then
    return $eval(\psi_1, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle) \cup eval(\psi_2, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
  elsif $\varphi = \langle\langle \Gamma \rangle\rangle \bigcirc \psi$ then
    return $pre(\Gamma, eval(\psi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle))$
  elsif $\varphi = \langle\langle \Gamma \rangle\rangle \Box \psi$ then
    $Q_1 := Q$
    $Q_2 := Q_3 := eval(\psi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
    while $Q_1 \not\subseteq Q_2$ do
      $Q_1 := Q_1 \cap Q_2$
      $Q_2 := pre(\Gamma, Q_1) \cap Q_3$
    end-while
    return $Q_1$
  elsif $\varphi = \langle\langle \Gamma \rangle\rangle \psi_1 \mathcal{U} \psi_2$ then
    $Q_1 := \emptyset$
    $Q_2 := eval(\psi_2, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
    $Q_3 := eval(\psi_1, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
    while $Q_2 \not\subseteq Q_1$ do
      $Q_1 := Q_1 \cup Q_2$
      $Q_2 := pre(\Gamma, Q_1) \cap Q_3$
    end-while
    return $Q_1$
  elsif $\varphi = K_a \psi$ then
    $Q_1 := eval(\psi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
    return $\{q \mid img(q, \sim_a) \subseteq Q_1\}$
  elsif $\varphi = C_\Gamma \psi$ then
    $Q_1 := eval(\psi, \langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle)$
    return $\{q \mid img(q, \sim_\Gamma^C) \subseteq Q_1\}$
  end-if
end-function
```

Figure 4. A model checking algorithm for ATEL.

A Model Checking Case Study

We now give an example of how ATEL can be used to reason about multiagent systems, and in particular how ATEL properties of a system can be verified using model checking. The system we consider is a train controller (adapted from [2]). The system contains three agents: two trains, and a controller — see Figure 5(a). The trains, one of which is Eastbound, the other of which is Westbound, each occupy their own circular track. At one point, both tracks pass through a narrow tunnel — there is not room for both trains in the tunnel at the same time. There are traffic lights on both sides of the tunnel, which can be either red or green. Both trains are equipped with a signaller, with which they can send signals to the controller; the idea is that they send a signal when they approach the tunnel. The controller can receive signals from both trains, and controls the color of the traffic lights. The task of the controller is, first and foremost, to ensure that the trains are never both in the tunnel at the same time; the secondary task is to ensure the “smooth running” of the system (e.g., the trains can always move through the tunnel, they cannot be forced into the tunnel, and so on).

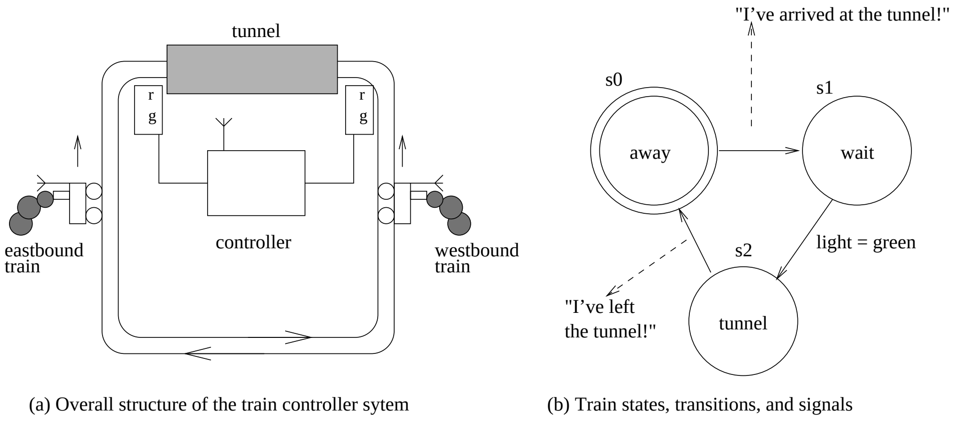
tunnel
r
g
r
g
controller
eastbound train
westbound train
"I've arrived at the tunnel!"
s0
away
s1
wait
s2
tunnel
light = green
"I've left the tunnel!"
(a) Overall structure of the train controller sytem
(b) Train states, transitions, and signals

Figure 5. The train controller system.

The train controller system was modelled by Alur and colleagues using a prototype model checking system for ATL called MOCHA [4, 2]. MOCHA takes as input an alternating transition system described using a (relatively) high level language called REACTIVEMODULES. The system is then capable of either randomly simulating the execution of this system, or else of taking formulae of ATL and automatically checking their truth or falsity in the transition system. As well as ATL formulae, MOCHA is capable of *invariant* checking — of checking whether a given property is true across all reachable states of the system. Although in its current implementation, MOCHA is capable of model checking arbitrary ATL formulae, and hence of determining whether or not there exists a collective strategy to achieve some multiagent goal, it does not exhibit such a strategy; it merely announces whether one exists.

In the model developed by Alur and colleagues, each train was represented by an automaton that could be in one of three states (see Figure 5(b)): “away” (state s_0 — the initial state of the train); “wait” (state s_1 — waiting for a green light to enter the tunnel); and “tunnel” (state s_2 — the train is in the tunnel). Transitions between states may be guarded: for example, in order for a train to go from s_1 to s_2, the condition “signal is green” must be true. If a state transition is not labelled with a condition, then the condition is assumed to be always true. In addition, when an agent makes a transit from one state to another, it may send a signal, as indicated by dashed lines in Figure 5(b). So, for example, when a train is entering the tunnel, it sends a signal to the controller to this effect. Note that just because a train *can* make a state transition does not necessarily mean it does so: it may be "lazy" (in the terminology of MOCHA), staying in the same state.

The train controller itself starts by setting both traffic lights to red. When a train approaches the tunnel (indicated by an "entering the tunnel" signal), the controller checks whether the opposing light is red; if it is, then the light for the approaching train is set green, allowing access. When a train moves out of the tunnel (also indicated by a signal to the controller), the controller sets the light associated with this train to red.

In [2], various possible ATL properties of this system are discussed, and may be automatically checked using MOCHA. However, currently MOCHA does not support the knowledge modalities of ATEL. We now discuss a preliminary approach we have developed to check knowledge properties using MOCHA, which involves translating knowledge formulae into ATL. The idea is inspired by translation-based theorem proving methods for modal logics, which exploit the fact that formulae of modal logic can be automatically translated into first-order logic; the theory of the approach is discussed in more detail in [24].

The main component of ATEL missing from MOCHA is the accessibility relations used to give a semantics to knowledge. Where do these relations come from? We use the *interpreted systems* approach of [15]. Given a state $q \in Q$ and agent $a \in \Sigma$, we write $state_a(q)$ to denote the *local* state of agent a when the system is in state q. The agent's local state includes its program counter and all its local variables. We then define the accessibility relation $\sim_a$ as follows:

$$q \sim_a q' \quad \text{iff } state_a(q) = state_a(q'). \tag{xxxiv}$$

We emphasise that this approach is well known and widely used in the distributed systems and epistemic logic communities. So, suppose we want to check whether, when the system is in some state q, agent a knows φ, i.e., whether $S, q \models K_a\varphi$. Then by (xxxiv), this amounts to showing that

$$\forall q' \in Q \text{ s.t. } state_a(q) = state_a(q') \text{ we have } S, q' \models \varphi. \tag{xxxv}$$

We can represent such properties directly as formulae of ATL, which can be automatically checked using MOCHA. In order to do this, we need some additional notation:

- we express the value of $state_a(q)$ as a constant s; and

- we have a logical variable $state_a$ that denotes, in any given state q, the value of $state_a(q)$.

Then we can express (xxxv) as the following ATL invariant formula:

$$\langle\langle\rangle\rangle\Box((state_a = s) \rightarrow \varphi) \qquad \text{(xxxvi)}$$

Note that in saying that (xxxvi) is an invariant, we are stating that it must hold across all reachable states of the system. Another way of reading (xxxvi) is as "agent a's state *s carries the information that* φ".

Such formulae can be directly written as properties that can be checked using MOCHA:

```
<< >>G((stateA = s) -> phi)
```

The `G` here is the MOCHA text form of the "always" operator ("$\Box$").

Turning back to the train example, we now show how a number of knowledge properties of the system were proven. First, consider the property that "when one train is in the tunnel, it knows the other train is not in the tunnel":

$$(state_a = tunnel) \rightarrow K_a(state_b \neq tunnel) \quad (a \neq b \in \{E, W\})$$

Translating into the MOCHA text form of ATL this schema gives the following two formulae

```
<<>> G ((stateE=tunnel) => ~(stateW=tunnel))
<<>> G ((stateW=tunnel) => ~(stateE=tunnel))
```

which were successfully model checked.

We can also show that when a train is away from the tunnel, it does not know whether or not the other train is in the tunnel.

$$\begin{aligned}&\langle\langle\rangle\rangle\Box(state_a \neq tunnel) \rightarrow\\ &\quad[(\neg K_a(state_b = tunnel)) \wedge (\neg K_a(state_b \neq tunnel))]\\ &\qquad\qquad\qquad\qquad\qquad (a \neq b \in \{E, W\})\end{aligned}$$

For the westbound train, we do this by checking the following formulae, both of which fail.

```
<<>> G ~(stateE=tunnel) => (stateW=tunnel)
<<>> G ~(stateE=tunnel) => ~(stateW=tunnel)
```

We can conclude that the *only way a train knows whether the other train is in the tunnel is if it is in the tunnel itself* (in which case it knows the other is not).

$$\langle\langle\rangle\rangle\Box[(K_a(state_b \neq tunnel)) \leftrightarrow (state_a = tunnel)] \quad a \neq b \in \{E, W\}$$

We can also check properties relating knowledge and ability. For example, we can prove that an agent always knows that the system can cooperate with it to allow it eventual access.

$$\langle\langle\rangle\rangle\Box K_a\langle\langle\Sigma\rangle\rangle\Diamond(state_a = tunnel) \qquad (a \in \{E, W\})$$

Since we wish to show that an agent always knows something, the quantification over agent a's knowledge-accessible states is across *all* states of the system. We can thus write, for the westbound train (`C` is the controller, and `F` is the MOCHA form of "$\Diamond$"):

```
<<>> G <<C,TrainW,TrainE>> F (stateW=tunnel)
```

In fact, since we are quantifying over *all* states of the system, this gives us that it is *always common knowledge that the grand coalition of all agents can cooperate to eventually get train a in the tunnel*:

$$\langle\langle\rangle\rangle\Box C_\Sigma\langle\langle\Sigma\rangle\rangle\Diamond(state_a = tunnel) \qquad (a \in \{E, W\}) \qquad \text{(xxxvii)}$$

We now consider formulae that express the fact that "one agent causes another agent to know something". The following formula means that Γ can cooperate to make a know φ.

$$\langle\langle\Gamma\rangle\rangle\Diamond K_a\varphi$$

Without quantification, which we do not have in MOCHA, this property is not expressed so easily — but it is possible. To see how we might do this, assume that agent a can be in n distinct states $s_1, \ldots, s_n$. Then saying that Γ can bring about knowledge of φ in agent a is the same as saying:

- agent a's state s_1 carries information φ and Γ can ensure that a enters s_1; or
- agent a's state s_2 carries information φ and Γ can ensure that a enters s_2; or...
- agent a's state s_n carries information φ and Γ can ensure that a enters s_n.

This observation allows us to rewrite $\langle\langle\Gamma\rangle\rangle\Diamond K_a\varphi$ in ATL as:

$$\bigvee_{1\leq i\leq n}(\langle\langle\rangle\rangle\Box((state_a = s_i)\rightarrow\varphi)\wedge\langle\langle\Gamma\rangle\rangle\Diamond state_a = s_i)$$

Such ATL formulae can be directly coded and checked in MOCHA.

The first such property we prove relates to a train's knowledge about whether or not the other train is in the tunnel. Consider: is it possible to cause a train to know that the other is not in the tunnel? We saw above that when one train is in the tunnel, it knows that the other is not; but when a train is away from the tunnel, it has no definite knowledge about the position of the other train. So, for a train to know that the other train is not in the tunnel, *it must be in the tunnel.* The first property we can check is that the grand coalition of agents can cooperate to make a train know that the other is not in the tunnel:

$$\langle\langle\rangle\rangle\Box\langle\langle\Sigma\rangle\rangle\Diamond K_a(state_b\neq tunnel)\qquad(a\neq b\in\{E,W\})$$

For the westbound train, this property when translated and slightly simplified becomes the following property, which can readily be checked in MOCHA.

```
<<>>G (<<TrainW,C,TrainE>> F (stateW=tunnel))
```

Interestingly, *no other subset of agents can bring this knowledge about* — because no other subset of agents can be guaranteed to get the westbound train into the tunnel. Thus, for example, the following property does not hold.

$$\langle\langle\rangle\rangle\Box\langle\langle a\rangle\rangle\Diamond K_a(state_b\neq tunnel)\qquad(a\neq b\in\{E,W\})$$

From (xxxvii), we know that it is always common knowledge that the entire system can cooperate to get a train in the tunnel. We can thus conclude that it is always common knowledge that the entire system can cooperate to eventually cause train a to know that train b is not in the tunnel:

$$\langle\langle\rangle\rangle\Box C_\Sigma\langle\langle\Sigma\rangle\rangle\Diamond K_a(state_b\neq tunnel)\qquad(a\neq b\in\{E,W\})$$

7. Conclusion

In this paper, we have introduced a natural extension to the Alternating Temporal Logic of Alur and colleagues, which includes modalities for representing knowledge, common knowledge, and the like. Using a simple example, we illustrated how an existing model checker can be put to work to verify formulas in ATEL.

In recent years, there has been much interest in the use of logic for representing and reasoning about game-like interactions. Examples include the development of logics intended for reasoning about coalitional power in games [31]; Goranko exploited the similarity between coalition logic and ATL to give a completeness proof for a subsystem of ATL [16]. This might suggest that Pauly's [31] may give us hints to explore the complexity of ATL and ATEL further. Alternatively, our work on ATEL may provide ways to enrich coalition logic with epistemic notions. In spite of the close relationship between the two frameworks, there are also differences, of which the details have yet to be sorted out.

Also of relevance to our work is the use of dynamic epistemic logics to capture properties of games [22, 6, 11, 5], Bonanno's work on the relationship of branching time logic to extensive form games [8], (building on earlier work by Ladner and colleagues [25, pp.208–209]), and of course the use of epistemic logic for capturing such game theoretic concepts as perfect recall [15].

Recently, there has been a lot of emphasis on modelling knowledge and its dynamics in one and the same framework [11, 5], which, in turn, also takes the Multi Agent System approaches to belief revision seriously, since it can model agents updating information about each other's information. It is clear that ATEL offers a framework to facilitate this. Moreover, many impressive platforms have emerged that integrate (the dynamics of) epistemics, rationality, and decision making. Enhancement of the work begun in this paper might further the computational relevance of such integrated theories. This is especially so, since the focus of the mentioned platforms has thus far been on formalising epistemic notions in game-theoretic settings. The question of how to use these formalisations in finding winning strategies in games of imperfect information for example, has only recently been asked (cf. [12]).

References

[1] Allen, J. F., H. Kautz, R. Pelavin, and J. Tenenberg, *Reasoning About Plans*, Morgan Kaufmann Publishers, San Mateo, CA, 1991.

[2] Alur, R., L. de Alfaro, T. A. Henzinger, S. C. Krishnan, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Taşiran, MOCHA user manual, University of Berkeley Report, 2000.

[3] Alur, R., T. A. Henzinger, and O. Kupferman, 'Alternating-time temporal logic', in *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*, pages 100–109, Florida, October 1997.

[4] Alur. R., T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Taşiran, *Mocha: Modularity in model checking*, in *CAV 1998: Tenth International Conference on Computer-aided Verification, (LNCS Volume 1427)*, pp. 521–525, Springer-Verlag, Berlin, Germany, 1998.

[5] Baltag, A.. 'A logic for supicious players', *Bulletin of Economic Research*, 54(1):1–45, 2002.

[6] van Benthem, J., 'Extensive games as process models', *Journal of Logic, Language, and Information*, 11(3):289–313, 2002.

[7] Binmore, K., *Fun and Games: A Text on Game Theory*, D. C. Heath and Company: Lexington, MA, 1992.

[8] Bonanno, G., 'Branching time logic, perfect information games and backward induction', *Games and Economic Behavior*, 36(1):57–73, July 2001.

[9] Burrows, M., M. Abadi, and R. M. Needham, 'A logic of authentication', *ACM Transaction on Computer Systems*, 8:18–36, 1990.

[10] Clarke, E. M., O. Grumberg, and D. A. Peled, *Model Checking*, The MIT Press: Cambridge, MA, 2000.

[11] van Ditmarsch, H. P., *Knowledge Games*, PhD thesis, University of Groningen, Groningen, 2000.

[12] Druiven, S., *Knowledge development in games of imperfect information*, Master's thesis, Groningen University, 2002. `http://www.ai.rug.nl/~sjoerd/cv/ps/thesis.ps`, retrieved october 2002.

[13] Emerson, E. A., 'Temporal and modal logic', in J. van Leeuwen, (ed.), *Handbook of Theoretical Computer Science Volume B: Formal Models and Semantics*, pp. 996–1072, Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1990.

[14] Emerson, E. A., and J. Y. Halpern, "Sometimes' and 'not never' revisited: on branching time versus linear time temporal logic', *Journal of the ACM*, 33(1):151–178, 1986.

[15] Fagin, R., J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning About Knowledge*, The MIT Press: Cambridge, MA, 1995.

[16] Goranko, V., 'Coalition games and alternating temporal logics', in J. van Benthem, (ed.), *Theoretical Aspects of Rationality and Knowledge (TARK VIII)*, pp. 259–272. Morgan Kaufmann, 2001.

[17] Grant, J., S. Kraus, and D. Perlis, 'A logic for characterizing multiple bounded agents', *Autonomous Agents and Multi-Agent Systems*, 4(3):351–387, 2000.

[18] Halpern, J. Y., and Y. Moses, 'Knowledge and common knowledge in a distributed environment', *Journal of the ACM*, 37(3):549–587, 1990.

[19] Halpern, J. Y., and Y. Moses, 'A guide to completeness and complexity for modal logics of knowledge and belief', *Artificial Intelligence*, 54:319–379, 1992.

[20] Halpern, J. Y., and M. Y. Vardi', 'The complexity of reasoning about knowledge and time. I. Lower bounds', *Journal of Computer and System Sciences*, 38:195–237, 1989.

[21] Harel, D., D. Kozen, and J. Tiuryn, *Dynamic Logic*, The MIT Press: Cambridge, MA, 2000.

[22] Harrenstein, P., W. van der Hoek, J.-J Meyer, and C. Witteveen, 'On modal logic interpretations for games', in *Proceedings of the Fifteenth European Conference on Artificial Intelligence (ECAI-2002)*, pp. 28–32, Lyon, France, 2002.

[23] van der Hoek, W., B. van Linder, and J.-J. Ch. Meyer, 'Group knowledge is not always distributed (neither is it always implicit)', *Mathematics for the Social Sciences*, 38(2):215–240, 1999.

[24] van der Hoek, W., and M. Wooldridge, 'Model checking knowledge and time', in D. Bošnački and S. Leue, (eds.), *Model Checking Software, Proceedings of SPIN 2002 (LNCS Volume 2318)*, pp. 95–111. Springer-Verlag: Berlin, Germany, 2002.

[25] Ladner, R. E., and J. H. Reif, 'The logic of distributed protocols: preliminary report', in *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge*, pp. 207–222, Morgan Kaufmann Publishers: San Mateo, CA, 1986.

[26] Lomuscio, A., *Knowledge Sharing among Ideal Agents.* PhD thesis, School of Computer Science, University of Birmingham, Birmingham, UK, June 1999.

[27] Meyer, J.-J. Ch., and W. van der Hoek, *Epistemic Logic for AI and Computer Science*, Cambridge University Press: Cambridge, England, 1995.

[28] Moore, R. C., 'A formal theory of knowledge and action', in J. F. Allen, J. Hendler, and A. Tate, (eds.), *Readings in Planning*, pp. 480–519 Morgan Kaufmann Publishers: San Mateo, CA, 1990.

[29] Osborne, M. J. and A. Rubinstein, *A Course in Game Theory*, The MIT Press: Cambridge, MA, 1994.

[30] Pauly, M., *Logic for Social Software*, PhD thesis, University of Amsterdam, 2001, ILLC Dissertation Series 2001-10.

[31] Pauly, M., 'A modal logic for coalitional power in games', *Journal of Logic and Computation*, 12(1):149–166, 2002.

[32] van der Meyden, R., and M. Vardi, 'Synthesis from knowledge-based specifications', in *Proceedings CONCUR'98, 9th International Conference on Concurrency Theory*, number 1466 in LNCS, pp. 34–49. Springer, 1998.

[33] Vardi, M. Y., 'Branching vs. linear time: Final showdown', in T. Margaria and W. Yi, (eds.), *Proceedings of the 2001 Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2001 (LNCS Volume 2031)*, pp. 1–22, Springer-Verlag: Berlin, Germany, April 2001.

[34] Wooldridge, M., 'A knowledge-theoretic approach to distributed problem solving', in *Proceedings of the Thirteenth European Conference on Artificial Intelligence (ECAI-98)*, pp. 308–312, Brighton, United Kingdom, August 1998.

[35] Wooldridge, M., *An Introduction to Multiagent Systems*, John Wiley & Sons, 2002.

[36] Wooldridge, M., C. Dixon, and M. Fisher, 'A tableau-based proof method for temporal logics of knowledge and belief', *Journal of Applied Non-Classical Logics*, 8(3):225–258, 1998.

[37] Wooldridge, M., and N. R. Jennings, Intelligent agents: Theory and practice, *The Knowledge Engineering Review*, 10(2):115–152, 1995.

Wiebe van der Hoek
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF
Liverpool, United Kingdom
`wiebe@csc.liv.ac.uk`

Michael Wooldridge
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF
Liverpool, United Kingdom
`m.j.wooldridge@csc.liv.ac.uk`