THOMAS ÅGOTNES

ACTION AND KNOWLEDGE IN ALTERNATING-TIME TEMPORAL LOGIC

ABSTRACT. Alternating-time temporal logic (ATL) is a branching time temporal logic in which statements about what coalitions of agents can achieve by strategic cooperation can be expressed. Alternating-time temporal *epistemic* logic (ATEL) extends ATL by adding knowledge modalities, with the usual possible worlds interpretation. This paper investigates how properties of agents' *actions* can be expressed in ATL in general, and how properties of the interaction between action and knowledge can be expressed in ATEL in particular. One commonly discussed property is that an agent should know about all available actions, i.e., that the same actions should be available in indiscernible states. Van der Hoek and Wooldridge suggest a syntactic expression of this semantic property. This paper shows that this correspondence in fact does not hold. Furthermore, it is shown that the semantic property is not expressible in ATEL at all. In order to be able to express common and interesting properties of action in general and of the interaction between action and knowledge in particular, a generalization of the coalition modalities of ATL is proposed. The resulting logics, ATL-A and ATEL-A, have increased expressiveness without loosing ATL's and ATEL's tractability of model checking.

1. INTRODUCTION

Alternating-time temporal logic (ATL) (Alur et al. 1997) is a propositional logic in which statements about what coalitions can achieve by strategic cooperation can be expressed. ATL generalizes the path quantifiers A and E, for *all* and *some* computational paths, of the branching time temporal logic computational tree logic (CTL), to coalition modalities $\langle\langle G\rangle\rangle$ for every group of agents G. For example, $\langle\langle G\rangle\rangle \bigcirc p$ and $\langle\langle G\rangle\rangle \Diamond p$ mean that G have a collective strategy to ensure that, no matter what the other agents do, p will be true in the next state, and some future state, respectively.

While ATL is a logic about what agents can *do*, alone or in groups, it was already pointed out in Moore's (1984) seminal work on knowledge and action that agents in general have incomplete information about the world and that a proper logic about action

Synthese (2006) 149: 375–407
Knowledge, Rationality & Action 121–153
DOI 10.1007/s11229-005-3875-8

also needs to provide an account of what the agents *know*. In alternating-time temporal epistemic logic (ATEL) (van der Hoek and Wooldridge 2002) knowledge modalities are introduced, and knowledge is interpreted as truth in all worlds considered possible as in standard epistemic logic (Fagin et al. 1995; Meyer and van der Hoek 1995). For example, $K_i\langle\langle i\rangle\rangle \bigcirc p$ means that agent i knows that he can make p true in the next state.

This paper investigates how properties of the interaction between *action* and knowledge can be expressed in ATEL. As in Moore's work, actions will be considered to be first-class citizens which consequences can depend on the situation in which they are performed. Consider the following example: in a situation, or possible world, where the battery of my car is flat the motor will not start if I turn the key, while in a situation where the battery is not flat the motor will start. If I do not know whether the battery is flat, I cannot discern between these two situations, and from my subjective viewpoint turning the key in the two situations is *the same action* – yet the action has different consequences depending on the actual situation. Similarly, two actions may have the exact same consequences in all situations and still be viewed as different actions. This view of actions with subjective identity across states is facilitated by the latest versions of the semantics for ATL, where actions are represented by action names whose interpretation depend on the current state of the system, very much like propositions are represented by proposition letters.

Moore identifies two main interactions between action and knowledge: first, that knowledge is required prior to taking action and, second, that actions may change knowledge. A particular instance of the first point is a property which recently has been discussed in relation to ATEL: knowledge about all available actions, or equivalently, that the same actions are available in indiscernible states. This semantic property will henceforth be called *complete knowledge about (available) actions*. A syntactic ATEL expression for the semantic property has been proposed in the literature. The relationship between the syntactic expression and the semantic property is investigated in Section 3. It is shown that the claim in fact does not hold; the proposed formula does *not* express complete knowledge about available actions. Furthermore, it is shown that the semantic property is not expressible in ATEL at all. Of course, the former result follows from the latter, but the first result is discussed in some detail to allow for broader interpretations of definability. This shows

that the expressive power of ATEL is not strong enough for certain common and interesting properties of action and knowledge.

In Section 4 an extension of AT(E)L in which the cooperation modalities $\langle\langle G\rangle\rangle$ are generalized is proposed. This is done by allowing (sets of) *actions* instead of agent names inside $\langle\langle \ldots \rangle\rangle$. The new modalities are similar to those in propositional dynamic logic (PDL) (Harel, 1984): for example, $\langle\langle \text{set\_true}_i\rangle\rangle \bigcirc p$ means that p is a consequence of agent i performing the action set_true and $\langle\langle \text{set\_false}_i, \text{acc}_j\rangle\rangle \bigcirc q$ means that q is a consequence of agents i and j respectively performing the actions set_false and acc. The resulting logic, ATEL-A, is quite expressive and can, e.g., express complete knowledge about available actions. Although ATEL-A is presented here in the context of expressibility of the latter property, it is a much more general extension and many other examples of properties involving knowledge and actions expressible in ATEL-A are presented. Thus, ATEL-A is a general proposal for a more expressive logic. While the expressive power is increased, ATELs tractability of model checking is retained.

The following section introduces ATL and ATEL and discusses what it means to express a semantic property syntactically.

2. ALTERNATING-TIME TEMPORAL LOGICS

Alur et al. (2002) define the semantics of ATL via concurrent game structures (CGSs). The following definition, which is used in the remainder of the paper, is slightly different from the original one in that actions are identified by arbitrary labels rather than by natural numbers. Formally the difference is small, but the following model better fits the semantic assumption of actions as first-class entities. Similar variants have also been used by others (Jamroga 2003; van der Hoek et al. 2004), with similar motivations.

DEFINITION 1. (CGS) A CGS is a tuple

$$(k, Q, \Pi, \pi, \text{ACT}, d, \delta)$$

where

- $k > 0$ is a natural number of *players*. The set of players is $\Sigma = \{1, \ldots, k\}$.
- Q is a finite set of *states*.
- Π is a finite set of *propositions*.

- π is the *labeling function*, assigning a set $\pi(q) \subseteq \Pi$ to each $q \in Q$.
- ACT is a finite set of *actions*.
- For each player $i \in \Sigma$ and state $q \in Q$, $d_i(q) \subseteq \text{ACT}$ is the non-empty set of actions available to player i in q. $D(q) = d_1(q) \times \cdots \times d_k(q)$ is the set of *joint actions* in q. If $\vec{a} \in D(q)$, a_i denotes the ith component of $\vec{a}$.
- δ is the *transition function*, mapping each state $q \in Q$ and joint action $\vec{a} \in D(q)$ to a state $\delta(q, \vec{a}) \in Q$.

(Including the set of atomic propositions and the number of agents, which are also parameters of the logical language, in the semantic structures is untraditional in logic in general, but is done in both CGSs and other ATL structures.)

Q^+ is used to denote the set of non-empty finite strings over Q. A *computation* λ is an infinite sequence of states; $\lambda = q_0 q_1 \cdots$, where for each $j \geq 0$ there is a joint action $\vec{a} \in D(q_j)$ such that $\delta(q_j, \vec{a}) = q_{j+1}$. $\lambda[j]$ is used to denote the element in λ with index j (q_j), while $\lambda[0, j] \in Q^+$ is the prefix of λ with length $j+1$. A *strategy* for player i is a function $f_i \colon Q^+ \to \text{ACT}$ where $f_i(q_0 \cdots q_m) \in d_i(q_m)$, mapping any finite prefix of a computation to an action for player i. $\text{Str}(G)$ denotes the set of joint strategies for a group of agents $G \subseteq \Sigma$; $\vec{f}_G \in \text{Str}(G)$ iff $\vec{f}_G = \{f_i \colon i \in G\}$ where each f_i is a strategy for i. Given a state q and a joint strategy $\vec{f}_G$ for G, $\text{out}(q, \vec{f}_G)$ denotes the set of possible computations starting in state q where the agents in G use the strategies $\vec{f}_G$. Formally, $\lambda \in \text{out}(q, \vec{f}_G)$ iff

1. $\lambda[0] = q$,
2. $\forall_{j \geq 0} \exists_{\vec{a} \in D(\lambda[j])}$,
   (a) $\forall_{i \in G} a_i = f_i(\lambda[0, j])$,
   (b) $\delta(\lambda[j], \vec{a}) = \lambda[j+1]$.

ATL formulae $\langle\langle G \rangle\rangle \bigcirc \phi$, $\langle\langle G \rangle\rangle \Box \phi$, $\langle\langle G \rangle\rangle \Diamond \phi$ and $\langle\langle G \rangle\rangle \phi \mathcal{U} \phi'$ mean that the coalition G can cooperate – or that it has a joint strategy – to ensure that ϕ is true in the next state, all future states, some future state and until ϕ' is true, respectively. Formally, the syntax of the ATL language is defined over Π and Σ. Π are formulae, and if ϕ_1, ϕ_2 are formulae and $G \subseteq \Sigma$ then $\neg \phi_1$, $\phi_1 \vee \phi_2$, $\langle\langle G \rangle\rangle \bigcirc \phi_1$, $\langle\langle G \rangle\rangle \Box \phi_1$ and $\langle\langle G \rangle\rangle \phi_1 \mathcal{U} \phi_2$ are formulae. The usual derived propositional connectives are used, including $\top$ for an arbitrary propositional tautology, in addition to $\langle\langle G \rangle\rangle \Diamond \phi$ for $\langle\langle G \rangle\rangle \top \mathcal{U} \phi$. Furthermore,

dual operators are defined as follows. $[\![G]\!]\bigcirc\phi$ means $\neg\langle\langle G\rangle\rangle\bigcirc\neg\phi$, $[\![G]\!]\Box\phi$ means $\neg\langle\langle G\rangle\rangle\Diamond\neg\phi$ and $[\![G]\!]\Diamond\phi$ means $\neg\langle\langle G\rangle\rangle\Box\neg\phi$. Intuitively, $[\![G]\!]\bigcirc\phi$ means that G cannot cooperate to avoid ϕ being true in the next state, and so on for the other duals.

Satisfiability of a formula ψ in a state q of a CGS S, written $S, q \models \psi$ or just $q \models \psi$ when S is understood, is defined as follows, where $p \in \Pi$:

$$
\begin{array}{ll}
S, q \models p & \Leftrightarrow p \in \pi(q), \\
S, q \models \neg\phi & \Leftrightarrow S, q \not\models \phi, \\
S, q \models \phi_1 \vee \phi_2 & \Leftrightarrow S, q \models \phi_1 \text{ or } S, q \models \phi_2, \\
S, q \models \langle\langle G\rangle\rangle\bigcirc\phi & \Leftrightarrow \exists_{\vec{f}_G \in \mathrm{Str}(G)} \forall_{\lambda \in \mathrm{out}(q, \vec{f}_G)} S, \lambda[1] \models \phi, \\
S, q \models \langle\langle G\rangle\rangle\Box\phi & \Leftrightarrow \exists_{\vec{f}_G \in \mathrm{Str}(G)} \forall_{\lambda \in \mathrm{out}(q, \vec{f}_G)} \forall_{j \geq 0} S, \lambda[j] \models \phi, \\
S, q \models \langle\langle G\rangle\rangle\phi_1 \mathcal{U} \phi_2 & \Leftrightarrow \\
\multicolumn{2}{l}{\exists_{\vec{f}_G \in \mathrm{Str}(G)} \forall_{\lambda \in \mathrm{out}(q, \vec{f}_G)} \exists_{j \geq 0} (S, \lambda[j] \models \phi_2 \text{ and } \forall_{0 \leq k < j} S, \lambda[k] \models \phi_1).}
\end{array}
$$

If all states in a structure satisfy a formula, we say that the structure is a *model* of the formula, written $M \models \psi$. Two states q, q' in a given CGS are *equivalent*, written $q \equiv q'$, if they satisfy the same formulae. Two structures S, S' are equivalent, written $S \equiv S'$, if they are models of the same formulae.

2.1. *Action in ATL*

Several slightly different versions of structures for ATL exist in the literature. An earlier version which appears in many ATL related papers is alternating transition systems (ATSs) (Alur et al. 1999). ATSs have no explicit actions; choices are identified by their possible outcomes. An ATS is a tuple (k, Q, Π, π, δ) where the δ function maps an agent and a state to a set of possible choices, where a choice is a set of states: $\delta\colon Q \times \Sigma \to 2^{2^Q}$. For all possible $Q_i \in \delta(q, i)$, it is required that $|\bigcap_{i \in \Sigma} Q_i| = 1$. Computations and strategies are defined accordingly: a sequence $\lambda = q_0 q_1 \cdots$ is a computation iff $\{q_{k+1}\} = \bigcap_{i \in \Sigma} Q_i$ for some choices $Q_i \in \delta(q_k, i)$ for each k, and if f_i is a strategy for i then $f_i(\lambda q) \in \delta(q, i)$. The out function, and satisfaction of ATL formulae, is defined similarly to for CGSs.

In this paper the CGS model of action will be used. The reason is that it models the view of actions as first-class citizens with subjective identity mentioned in the introduction in a direct way. This is similar to the model used by Moore (1984) in that actions are

modeled very much like propositions in a possible worlds framework: an action is available to an agent in a possible world iff all the physical pre-conditions for performing the action are satisfied in that world. In the following, it will be assumed that an agent can observe the available actions in a given possible world, in the same way it is assumed that an agent can observe the true propositions in a given world (but, when agents with incomplete information about the world are introduced in the next subsection, an agent does not necessarily know which possible world is the real one and thus not necessarily which propositions are in fact true or which actions are available). The CGS model of *multi-agent* action is also similar to the popular (Fagin et al. 1995) model introduced in Halpern and Fagin (1989).

ATSs will nevertheless sometimes be used in the following when it is not immediately clear that results for CGSs also hold for ATSs.

2.2. *Alternating-time Temporal Epistemic Logic*

Alternating-time Epistemic Logic (van der Hoek and Wooldridge 2002) extends the ATL language with a knowledge modality K_i for each agent $i \in \Sigma$.[1]

Structures for ATEL are structures for ATL extended with an epistemic accessibility relation $\sim_i \subseteq Q \times Q$, required to be an equivalence relation, for each agent i. Originally, the structures were based on ATSs: an alternating epistemic transition system (AETS) is a tuple $(k, Q, \Pi, \pi, \delta, \sim_1, \ldots, \sim_k)$ where (k, Q, Π, π, δ) is an ATS. However, we can of course base ATEL structures on CGSs: a concurrent epistemic game structure (CEGS) is a tuple $(k, Q, \Pi, \pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ where $(k, Q, \Pi, \pi, \mathrm{ACT}, d, \delta)$ is a CGS.

Satisfaction (in either AETSs or CEGSs) for the new operators is defined as follows:

$$S, q \models K_i \phi \Leftrightarrow \forall_{q \sim_i q'} S, q' \models \phi$$

In the following, structures for ATEL will be assumed to be CEGSs unless otherwise noted.

The following are properties of ATEL which will be useful later.[2]

(1) $\langle\langle G \rangle\rangle \Box \phi \leftrightarrow (\phi \wedge \langle\langle G \rangle\rangle \bigcirc \langle\langle G \rangle\rangle \Box \phi)$

(2) $\langle\langle G \rangle\rangle \phi_1 \mathcal{U} \phi_2 \leftrightarrow (\phi_2 \vee (\phi_1 \wedge \langle\langle G \rangle\rangle \bigcirc \langle\langle G \rangle\rangle \phi_1 \mathcal{U} \phi_2))$

2.3. *Expressiveness and Frames*

Since expressiveness of logics is the main topic of this paper, it must be defined properly.

In general, a (possibly singular) set of formulae Γ can be said to define, or express, a semantic property iff the class of models for Γ is precisely the class of structures having the property.

In modal logic, however, it is tradition to say that a (set of) formula(e) define(s), or express(es), a property iff it define(s) the class of *frames* with that property. There seem to be no standard definition of a frame for ATEL, but in the following the most straightforward definition will be used: a frame is a CEGS without a labeling function, i.e., a tuple $(k, Q, \Pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$. Similarly, an AETS frame is a tuple $(k, Q, \Pi, \delta, \sim_1, \ldots, \sim_k)$. The following notation and terminology will be used for frames: a structure is *based* on a frame iff all components except the labeling function are equal; if F is a frame and π is a labeling function for F then (F, π) is the structure based on F with labeling function π; a frame F is a frame of a formula/set of formulae Γ if all structures based on F are models of Γ; $\mathrm{fr}(\Gamma)$ is the class of frames of Γ; two frames F, F' are *equivalent*, written $F \equiv F'$, iff they are frames of the same formulae. A *frame property* is a semantic property which holds for a frame iff it holds for all models based on the frame (e.g., Equation (3)). Similar definitions hold for AETSs. Thus we have another notion of definability: if the frames of Γ are exactly the class of frames having the property. When necessary, we will discern between the two types by using the terms "model-definability" and "frame-definability." Note that for a frame property, model-definability implies frame-definability.

3. EXPRESSING COMPLETE KNOWLEDGE ABOUT ACTIONS IN ATEL

One of the main goals in this paper is to investigate the expressiveness of ATEL when it comes to action properties. A key action property is complete knowledge about actions; that the same actions are available in indiscernible states. This property can be formalized in CEGSs as

$$q \sim_i q' \Rightarrow d_i(q) = d_i(q'). \tag{3}$$

A proposed formalization of the property in AETSs is the following (van der Hoek and Wooldridge 2003):

$$q \sim_i q' \Rightarrow \delta(q, i) = \delta(q', i). \tag{4}$$

A prerequisite for formalizing complete knowledge about actions is a notion of subjective identity of actions across states, as discussed in the introduction. The difference between (3) and (4) seem to be that different notions of action identity is assumed. While the fact that the same actions are available in two different states is naturally expressed in CEGSs as in (3), the formalization of this fact in AETSs used in (4) seems to be too strong. In effect the latter formalization assumes that an action is identified by a set of possible (depending on what the other agents do) outcomes. As discussed in the introduction, this does not correspond to the usual view of action identity in which e.g., the same action can have different sets of possible outcomes in different states. In particular, (3) and (4) are not equivalent. Thus, as already noted, C(E)GSs are used as semantic structures for AT(E)L, and (3) as the formalization of complete knowledge about actions, in most of the following discussion – but corresponding results for A(E)TSs and (4) will also be mentioned.

One of the reasons the property of complete knowledge about actions is interesting is that it has recently (Jamroga 2003; van der Hoek and Wooldridge 2003; Jamroga and van der Hoek 2004) been pointed out that ATEL does not seem to integrate the semantics of knowledge with the ATL semantics properly, because it is not required that a strategy maps indiscernible histories of states to the same action. The fact that agents can base their choices on the state of the whole system, seems to contradict the premise of epistemic logic: that agents may have incomplete information. It has been argued that (3)/(4) is a part of a proper semantic solution to the problem.[3] Nevertheless, the problem under consideration here is orthogonal to the one just mentioned: to express interaction properties between knowledge and action such as (3) syntactically. One suggested expression from the literature is considered next.

3.1. *A Suggested Syntactic ATEL-Expression*

In van der Hoek and Wooldridge (2003 p. 144), it is claimed that the condition "same actions in indiscernible states" can be expressed syntactically

(5) $q \sim_i q' \Rightarrow \delta(q,i) = \delta(q',i).$

This gives us the following syntactic property.

(6) $\langle\langle i \rangle\rangle T\phi \leftrightarrow K_i \langle\langle i \rangle\rangle T\phi,$

where T is a temporal operator. This claim, however, is not true. It is not clear that "this gives us" means "is defined by," but it seems at least to imply that (5) implies (6). In any case, as a model property (5) is neither a sufficient nor a necessary condition for (6), and as a frame property (5) is not defined by (6) either. This was already shown for AETSs in (Ågotnes, 2004); here, it is shown for CEGSs – with (5) replaced by (3).

First, I show that (3) is not a sufficient condition for (6) by constructing a CEGS S_1 where (3) holds but with an agent i, a proposition p and a state q_1 such that

(7) $\quad S_1, q_1 \not\models \langle\langle i\rangle\rangle \bigcirc p \rightarrow K_i \langle\langle i\rangle\rangle \bigcirc p.$

First, let F_1 be the frame $(k^1, Q^1, \Pi^1, \mathrm{ACT}^1, d^1, \delta^1, \sim_1^1, \sim_2^1)$ where

- $k^1 = 2$
- $Q^1 = \{q_1, q_2\}$
- $\Pi^1 = \{p\}$
- $\mathrm{ACT}^1 = \{a, b\}$
- $d_1^1(q_1) = d_1^1(q_2) = d_2^1(q_1) = d_2^1(q_2) = \{a\}$
- $\delta^1(q_1, (a, a)) = q_1$
- $\delta^1(q_2, (a, a)) = q_2$
- $\sim_1^1 = \{(q_1, q_1), (q_2, q_2), (q_1, q_2), (q_2, q_1)\}$
- $\sim_2^1 = \{(q_1, q_1), (q_2, q_2)\}$.

Clearly, (3) holds for F_1. The following observation about F_1 is obvious.

OBSERVATION 1. For any π and any collective strategy $\vec{f}_G$ for any set of agents G, in the system (F_1, π):

(8) $\quad \mathrm{out}(q_1, \vec{f}_G) = \{\lambda_1\} \quad \mathrm{out}(q_2, \vec{f}_G) = \{\lambda_2\},$

where λ_1, λ_2 are the computations

(9) $\quad \lambda_1 = q_1 q_1 \cdots, \quad \lambda_2 = q_2 q_2 \cdots.$

Let $\pi_1 : Q \rightarrow 2^{\Pi}$ be such that $\pi_1(q_1) = \{p\}$ and $\pi_1(q_2) = \emptyset$, and let $S_1 = (F_1, \pi_1)$. S_1 is illustrated in Figure 1.

It is easy to see that

(10) $\quad S_1, q_1 \models \langle\langle 1\rangle\rangle \bigcirc p$

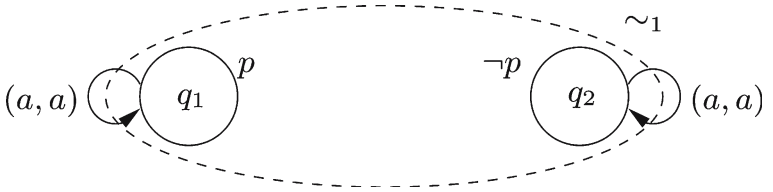

Figure 1. The CEGS S_1.

holds, since $S_1, \lambda_1[1] \models p$. If

$$(11) \qquad S_1, q_1 \models K_1 \langle\langle 1 \rangle\rangle \bigcirc p,$$

then $S_1, q_2 \models \langle\langle 1 \rangle\rangle \bigcirc p$, since $q_1 \sim_1^1 q_2$. But $S_1, q_2 \not\models \langle\langle 1 \rangle\rangle \bigcirc p$ since $S_1, \lambda_2[1] \not\models p$, so (11) does not hold, which shows (7).

Thus, (3) is not a sufficient condition for (6). To show that it is neither a necessary condition, I construct a CEGS S_2 which is a model for (6) for any i, T and ϕ, for which (3) does not hold for any q. Let $S_2 = (k^1, Q^2, \Pi^1, \pi^2, \mathrm{ACT}^1, d^2, \delta^2, \sim_1^2, \sim_2^2)$ where

- $Q^2 = \{q_1, q_2, q_3\}$
- $\sim_1^2 = \sim_1^1 \cup \{(q_3, q_3)\}$, $\sim_2^2 = \sim_2^1 \cup \{(q_3, q_3)\}$
- $d_1^2(q_1) = d_2^2(q_1) = d_2^2(q_2) = d_1^2(q_3) = d_2^2(q_3) = \{a\}$
- $d_1^2(q_2) = \{b\}$
- $\delta^2(q_1, (a,a)) = \delta^2(q_3, (a,a)) = \delta^2(q_2, (b,a)) = q_3$
- $\pi^2(q_1) = \pi^2(q_2) = \pi^2(q_3) = \{p\}$

and $k^1, \Pi^1, \mathrm{ACT}^1, \sim_1^1, \sim_2^1$ are as in S_1. S_2 is illustrated in Figure 2. Let T be a state quantifier and ϕ be a formula. To show that S_2 is a model of (6), it suffices to show that

$$(12) \qquad S_2, q_1 \models \langle\langle 1 \rangle\rangle T \phi \Leftrightarrow S_2, q_2 \models \langle\langle 1 \rangle\rangle T \phi$$

since it follows trivially in state q_3 and when $i = 2$. It is easy to see that the only computation starting in q_1 is $q_1 q_3 q_3 \cdots$ and the only computation starting in q_2 is $q_2 q_3 q_3 \cdots$. Satisfaction of a formula ψ in a state q depends only on (i) $\pi(q)$, (ii) the states accessible for each agent from q and (iii) the set of possible remaining computations starting in q. For q_1 and q_2, (i) $\pi^2(q_1) = \pi^2(q_2)$, (ii) $q_1 \sim_i^1 q'$ iff $q_2 \sim_i^1 q'$ for $i \in \Sigma$ and (iii) the set of possible remaining computations starting in q_1 or q_2 is $\{q_3 q_3 \cdots\}$. Thus, $q_1 \equiv q_2$ in S_2 and (12) holds. However, (3) does not hold for S_2: $q_1 \sim_1^1 q_2$ but $d_1^2(q_1) \neq d_1^2(q_2)$.

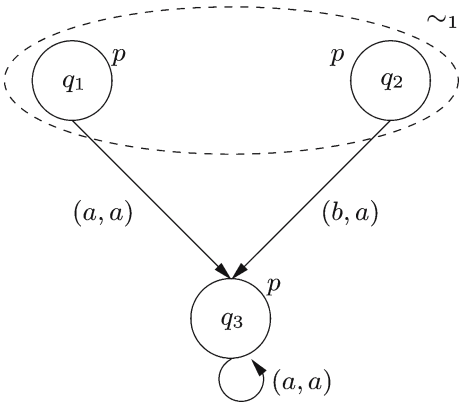

Figure 2. The CEGS S_2.

In other words; there are models of the schema (6) in which (3) does not hold, and there are structures in which (3) holds which are not models of the schema (6).

3.1.1. *A Frame Property*

Since it does not depend on the labeling function, the property (3) can be read as a frame property, rather than as a model property. So far we have only looked at the *models* of (6), but since there are CGSs with the (3) property which are not models of (6), there are also *frames* with the (3) property which are not frames of (6). A natural question is whether the other direction holds for frames: does (3) hold for all frames of (6)? The answer is in fact "yes". It holds, however, because (6) as a frame condition is really not very interesting: if $F = (k, Q, \Pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_n)$ is a frame of (6) then

$$(13) \qquad q \sim_i q' \Rightarrow q = q'$$

for each $i \in \Sigma$. To see this, let $F \models (6)$ and assume that $q \sim_i q'$ for some $q \neq q'$. Let π be such that $\pi(q) = \{p\}$ and $\pi(q') = \emptyset$. Then $(F, \pi), q \models (6)$, particularly[4]

$$(F, \pi), q \models \langle\langle i \rangle\rangle p \mathcal{U} p \leftrightarrow K_i \langle\langle i \rangle\rangle p \mathcal{U} p$$

But $(F, \pi), q \models \langle\langle i \rangle\rangle p \mathcal{U} p$ and $(F, \pi), q' \not\models \langle\langle i \rangle\rangle p \mathcal{U} p$, which is a contradiction showing that the assumption is impossible. Thus, an agent i in a frame of (6) is *factually omniscient*: he always knows every true fact.

Thus, (6) only describes a proper subset of the (3)-frames, consisting of frames in which agents can have no interesting properties of knowledge.

3.1.2. *The Problem*

While (6) does not express (3), the next question is whether the problem can be "fixed" syntactically by looking at the counter models. One source of confusion about action and ability is that a statement such as $\langle\langle i\rangle\rangle \bigcirc \phi$ often is informally interpreted as a statement about agent i's capability to make ϕ come about. This is an imprecise interpretation. It may be that $\langle\langle i\rangle\rangle \bigcirc \phi$ holds because the rest of the system will deterministically make ϕ true in the next state – no matter what agent i does. In other words, it may be that $\langle\langle \emptyset\rangle\rangle \bigcirc \phi$ holds – which trivially implies $\langle\langle i\rangle\rangle \bigcirc \phi$. For example, Equation (10) holds because the system S_1 will if started in state q_1 deterministically stay in q_1 forever.

It may be argued that a case where $\langle\langle i\rangle\rangle \bigcirc \phi$ holds because $\langle\langle \emptyset\rangle\rangle \bigcirc \phi$ holds is a special case which was not the intention of (6), and that the fully deterministic system S_1 is not a very interesting counter model. The case may be ruled out syntactically:

$$(14) \qquad (\langle\langle i\rangle\rangle \bigcirc p \wedge \neg\langle\langle \emptyset\rangle\rangle \bigcirc p) \rightarrow K_i \langle\langle i\rangle\rangle \bigcirc p.$$

Indeed (trivially),

$$S_1 \models (\langle\langle i\rangle\rangle \bigcirc p \wedge \neg\langle\langle \emptyset\rangle\rangle \bigcirc p) \rightarrow K_i \langle\langle i\rangle\rangle \bigcirc p.$$

However, there are (non-deterministic) counter-models of (14) where (3) holds. One such structure, S_3, is illustrated on Figure 3. $S_3 = (k^1, Q^2, \Pi^1, \pi^3, \mathrm{ACT}^1, d^3, \delta^3, \sim_1^2, \sim_2^2)$ where k^1, Q^2, Π^1, ACT^1, $\sim_1^2$, $\sim_2^2$ are as before and

- $\pi^3(q_1) = p$, $\pi^3(q_2) = \pi^3(q_3) = \emptyset$
- $d_1^3(q_1) = d_1^3(q_2) = \{a, b\}$
- $d_2^3(q_1) = d_2^3(q_2) = d_1^3(q_3) = d_2^3(q_3) = \{a\}$
- $\delta^3(q_1, (a, a)) = q_1$
- $\delta^3(q_2, (a, a)) = q_2$
- $\delta^3(q_1, (b, a)) = \delta^3(q_2, (b, a)) = \delta^3(q_3, (a, a)) = q_3$.

$S_3, q_1 \models \langle\langle 1\rangle\rangle \bigcirc p$ and $S_3, q_1 \models \neg\langle\langle \emptyset\rangle\rangle \bigcirc p$, but $S_3, q_2 \not\models \langle\langle 1\rangle\rangle \bigcirc p$ so

$$S_3, q_1 \not\models (\langle\langle 1\rangle\rangle \bigcirc p \wedge \neg\langle\langle \emptyset\rangle\rangle \bigcirc p) \rightarrow K_1 \langle\langle 1\rangle\rangle \bigcirc p.$$

S_3 might be a more interesting counter-model of (6) than S_1 is.

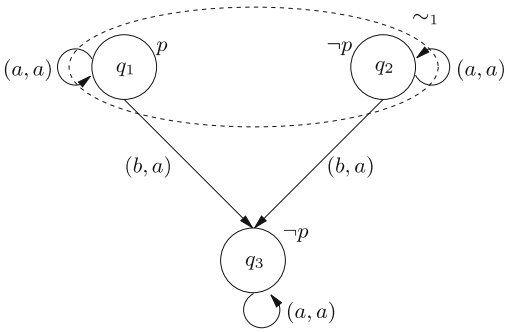

Figure 3. The CEGS S_3.

The problem seems to be that temporal connectives in ATL express something about future states, without regard to how they come about. Therefore, local semantic properties of individual agents' actions, such as interaction between action and knowledge, are difficult to express syntactically in ATEL. We have seen that (3) is not expressed by (6) (or by (14)); in the following section it is shown that it is not expressible in ATEL at all.

3.2. *Undefinability in ATEL*

One of the reasons the semantic property (3) is inexpressible in ATEL is that frames may contain "equivalent" actions – actions which lead to the same outcome when the actions of the other agents are fixed. In this section, that notion is made precise. It is shown that every class of frames (models) expressible by an ATEL formula must be closed under equivalent actions, and by counter examples that the classes described by (3) are not.

Henceforth, if $\vec{a} \in \mathrm{ACT}^k$ is a joint action and $b \in \mathrm{ACT}$ is an action, $\vec{a}[b/i] \in \mathrm{ACT}^k$ is $\vec{a}$ with a_i replaced by b.

First, the notion of two actions being equivalent for a given agent in a given state is formalized.

DEFINITION 2. Actions a and b are *equivalent for agent i in state q of frame* $F = (k, Q, \Pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ iff

(1) $a \in d_i(q) \Leftrightarrow b \in d_i(q)$,
(2) $a, b \in d_i(q) \Rightarrow \forall_{\vec{a} \in D(q)} \delta(q, \vec{a}[a/i]) = \delta(q, \vec{a}[b/i])$.

The result of *removing* one of two equivalent actions from a frame is simply a frame where the agent no longer can select the particular action. Formally:

DEFINITION 3. If a and b are equivalent for i in q' of $F = (k, Q, \Pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$, the result of *removing* action a for i in q' is the frame $F' = (k, Q, \Pi, \mathrm{ACT}, d', \delta', \sim_1, \ldots, \sim_k)$ where

$$d'_j(q) = \begin{cases} d_j(q) \setminus \{a\}, & \text{if } j = i, q = q', \\ d_j(q), & \text{otherwise,} \end{cases}$$

and δ' is the restriction of δ to d'.

The following Lemma shows that satisfaction is invariant under removal (or, equivalently, addition) of equivalent actions.

LEMMA 1. If a and b are equivalent for i in q' of F and F' is the result of removing action a for i in q', then for any labeling function π, formula ϕ and state q of F:

$$(15) \quad (F, \pi), q \models \phi \Leftrightarrow (F', \pi), q \models \phi.$$

Proof. See the appendix. □

That the semantic property (3) is inexpressible in ATEL can now be seen by constructing two frames, one which is the result of removing one of two equivalent actions from the other; one which has the property and one which does not.

THEOREM 1. The class of frames with the property (3) is not ATEL-definable.

Proof. We can use one of the frames we already have: $F_1 = (k^1, Q^1, \Pi^1, \mathrm{ACT}^1, d^1, \delta^1, \sim_1^1, \sim_2^1)$ from Section 3.1. Let $F_4 = (k^1, Q^1, \Pi^1, \mathrm{ACT}^1, d^4, \delta^4, \sim_1^1, \sim_2^1)$ where

$$(16) \quad d_i^4(q) = \begin{cases} \{a, b\}, & i = 1 \text{ and } q = q_1, \\ d_i^1(q), & \text{otherwise,} \end{cases}$$

$$(17) \quad \delta^4(q, \vec{a}) = \begin{cases} q_1, & q = q_1 \text{ and } \vec{a} = (b, a), \\ \delta^1(q, \vec{a}), & \text{otherwise.} \end{cases}$$

In other words, actions a and b are equivalent for agent 1 in state q_1 of frame F_4, and F_1 is F_4 with b removed for 1 in q_1. Thus by Lemma 1

(18) $\quad F_1 \equiv F_4.$

Let $\mathcal{F}_1$ be the class of frames having the property (3) and let Γ be a set of ATEL formulae. If $F_1 \not\models \Gamma$, then $\mathcal{F}_1 \not\subseteq \mathrm{fr}(\Gamma)$ since $F_1 \in \mathcal{F}_1$. If $F_1 \models \Gamma$, then $F_4 \models \Gamma$ by (18), and then $\mathrm{fr}(\Gamma) \not\subseteq \mathcal{F}_1$ since $F_4 \notin \mathcal{F}_1$. Thus, $\mathcal{F}_1 \neq \mathrm{fr}(\Gamma)$. □

COROLLARY 1. The class of models with the property (3) is not ATEL-definable.

Proof. Follows immediately, since (3) is a frame property (see Section 2.3). □

Since AETSs differ from CEGSs exactly in the definition of what an action/a choice is, and since ATEL was originally based on this earlier version of ATL structures and the claim about expressiveness (quoted on p. 382) was made in terms of AETSs, it should be mentioned that an analogous result to Theorem 1 can be obtained for AETSs – (5) is not ATEL definable under AETS semantics.

Counter examples[5] are the AETS frames $\hat{F}_1 = (k^1, Q^1, \Pi^1, \hat{\delta}, \sim_1^1, \sim_2^1)$ and $\hat{F}_4 = (k^1, Q^1, \Pi^1, \hat{\delta^2}, \sim_1^1, \sim_2^1)$ where

- $\hat{\delta}(q_1, 1) = \hat{\delta}(q_2, 1) = \{\ \{q_1, q_2\}\ \}$
- $\hat{\delta}(q_1, 2) = \{\ \{q_1\}\ \}$
- $\hat{\delta}(q_2, 2) = \{\ \{q_2\}\ \}$
- $\hat{\delta^2}(q, i) = \begin{cases} \{\ \{q_2\}\ \}, & q = q_2 \text{ and } i = 1, \\ \hat{\delta}(q, i), & \text{otherwise.} \end{cases}$

and the other components are as in the CEGS frame F_1. It is easy to see that $\hat{F}_1$ and $\hat{F}_2$ are logically equivalent, while the former has the property (5) and the latter does not.

4. ALTERNATING-TIME TEMPORAL LOGICS WITH ACTIONS

We have looked at the problems with expressing semantic properties, such as complete knowledge about available actions, in ATEL. In this section, a simple but more expressive extension of ATEL is presented.

The extension consists in generalizing the $\langle\langle \ldots \rangle\rangle$ operators to taking (sets of) *actions* from ACT as parameters in addition to agent names, in order to express what the agents can achieve with the given actions rather than just what they can achieve with *some* actions. For example,

$$\langle\langle \text{send}_i \rangle\rangle \bigcirc p$$

means that p will be true in the next state if agent i uses the send action, no matter what the other agents do. An example with a set of actions is $\langle\langle \{\text{send}, \text{receive}\}_i \rangle\rangle \bigcirc p$; i can use (at least) *one* of the actions send or receive to make p be true in the next state. $\langle\langle i \rangle\rangle \bigcirc \phi$ can thus be expressed as $\langle\langle \text{ACT}_i \rangle\rangle \bigcirc \phi$. ATL and ATEL with the new action operators are called ATL-A and ATEL-A, respectively. The idea is similar to the one behind dynamic linear time temporal logic (DLTL) (Henriksen and Thiagarajan 1999), where the temporal operators of LTL are indexed with programs of the type used in PDL and where atomic programs can be seen as actions.

Actions in ATL have consequences for the *next state* of the system; thus the semantics of the new operators are intimately connected to the $\bigcirc$ state quantifier.[6]

Moore (1984) describes two principal interactions between action and knowledge: first, knowledge is often required prior to taking action, and, second, action can change what is known. After a formal definition, it is shown how among other things such interactions can be expressed in ATEL-A.

4.1. *Formal Definitions*

The new generalized operators are constructed by restricting the actions one or more of the agents can use in the next step. Formally, an *action descriptor* consists of a set of legal actions for each agent. The set of all action descriptors is

$$\text{descr} = (\wp(\text{ACT}) \setminus \{\emptyset\})^k.$$

Given an $A \in \text{descr}$, A_j will be used to denote the jth component of A, and

$$A^{\times} = \bigtimes_{j \in \Sigma} A_j$$

is the set of legal joint actions. As before, we write a_i for the ith element of a given tuple $\vec{a} \in \text{ACT}^k$, and correspondingly, if $b_1, \ldots, b_k \in$

ACT we write $\vec{b}$ for $(b_1, \ldots, b_n)$. If $\vec{a}, \vec{b} \in \mathrm{ACT}^k$ are two joint actions, then

$$\vec{b}[\vec{a}/G] = (c_1, \ldots, c_k) \quad \text{where} \quad \begin{cases} c_i = a_i, & i \in G, \\ c_i = b_i, & i \notin G \end{cases}$$

is the joint action with actions from $\vec{a}$ for G and actions from $\vec{b}$ for $\Sigma \setminus G$.

The language of ATL-A (ATEL-A) is the language of ATL (ATEL) with the formation rule for $\langle\langle G\rangle\rangle\bigcirc$ replaced by: if ϕ is a formula, $G \subseteq \Sigma$ and $A \in \mathrm{descr}$ then $\langle\langle A, G\rangle\rangle \bigcirc \phi$ is a formula. This notation allows not only restricting the legal actions for the agents in the coalition G, but also for the agents outside the coalition, allowing, e.g., the expression of what a coalition can achieve when other agents' actions are fixed or otherwise restricted.

CGSs/CEGSs are still used to define the semantics of ATL-A/ATEL-A formulae. The definition of satisfaction is extended to the new operators as follows. In evaluating the expression $S, q \models \langle\langle A, G\rangle\rangle \bigcirc \phi$ we must consider the actions available to each agent, both in G and in $\Sigma \setminus G$, i.e., the set of joint actions that the set Σ of all agents have available. This set is restricted in two ways. First, the set of joint actions available to the agents in q in the first place is $D(q)$. Second, the action descriptor A says that we should only consider what happens when joint actions in the set $A^{\times}$ are used. Thus, finding the truth of $\langle\langle A, G\rangle\rangle \bigcirc \phi$ is very much like finding the truth of $\langle\langle G\rangle\rangle \bigcirc \phi$, except that we must only consider next states which are the result of performing a joint action from

$$A^{\times} \cap D(q)$$

rather than from $D(q)$. There is, however, an important difference. In the case of $\langle\langle G\rangle\rangle \bigcirc \phi$ we are guaranteed that the system can go on to a next state from q, i.e., there *exists* a joint action in $D(q)$. In the case of $\langle\langle A, G\rangle\rangle \bigcirc \phi$, we are *not* guaranteed that $A^{\times} \cap D(q)$ is non-empty – thus it may be that the system cannot go to a next state when agents are restricted to actions from A. In that case the proper semantics of the formula $\langle\langle A, G\rangle\rangle \bigcirc \phi$ is the value *false*, since there does not exist a next state where ϕ is true and thus G have no strategy to reach such a state. The following is the formal definition of satisfaction, and is discussed below:

$$S, q \models \langle\langle A, G\rangle\rangle \bigcirc \phi \Leftrightarrow$$
$$\exists_{\vec{a} \in A^{\times} \cap D(q)} \forall_{\vec{b} \in A^{\times} \cap D(q)} S, \delta(q, \vec{b}[\vec{a}/G]) \models \phi.$$

Informally, $\langle\langle A, G\rangle\rangle \bigcirc \phi$ holds iff (i) all agents (in and not in G) can act under the restriction A and (ii) given that all agents (in and not in G) are restricted by A, G can perform actions such that ϕ will be true in the next state no matter what the other agents do. In the definition of satisfaction, the existence of $\vec{a}$ ensures (i) and fixes the actions of G while $\vec{b}$ denotes arbitrary actions for the agents not in G.[7]

Similarly to $\langle\langle A, G\rangle\rangle\bigcirc$, operators $\langle\langle A, G\rangle\rangle\Box$ and $\langle\langle A, G\rangle\rangle\mathcal{U}$ can also be used in ATL-A/ATEL-A. However, as mentioned before, action descriptors only restrict the actions that can be used in the *next* step, and not forever. Thus, the two latter operators can be defined as derived operators by the former, by restricting the actions used in the first step. The derived operators are defined as follows, and their semantic interpretation can be defined similarly to (but slightly more complicated than) $\langle\langle A, G\rangle\rangle\bigcirc$.

$$\langle\langle A, G\rangle\rangle\Box\phi \equiv \phi \wedge \langle\langle A, G\rangle\rangle \bigcirc \langle\langle G\rangle\rangle\Box\phi, \tag{19}$$

$$\langle\langle A, G\rangle\rangle\phi_1\mathcal{U}\phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \langle\langle A, G\rangle\rangle \bigcirc \langle\langle G\rangle\rangle\phi_1\mathcal{U}\phi_2)\,. \tag{20}$$

For each tense quantifier T, the ATL operator $\langle\langle G\rangle\rangle T$ can be written as $\langle\langle A, G\rangle\rangle T$, as shown in Section 4.3. Restricting the actions only in the next step gives an intuitive interpretation also of the derived connectives; e.g., $\langle\langle \text{send}_i\rangle\rangle\Diamond p$ can be read as "if i uses send now, he can eventually achieve p."

Other derived connectives are $\langle\langle A, G\rangle\rangle\Diamond\phi$, for $\langle\langle A, G\rangle\rangle\top\mathcal{U}\phi$, and the duals $[\![A, G]\!]\bigcirc$, $[\![A, G]\!]\Box$, $[\![A, G]\!]\Diamond$, $[\![A, G]\!]\mathcal{U}$ which are defined similarly to their ATL counterparts.

The following notational shorthands are used. If we want to restrict the actions of only some of the agents, we want an action descriptor where $A_j = \text{ACT}$ for all the other agents. The sets $A_j = \text{ACT}$ will often be omitted when writing the descriptor, and the agents identified by subscripts. For example, a shorthand for the descriptor $A = (\text{ACT}, \{\text{send}\}, \text{ACT}, \{\text{send}, \text{receive}\})$ is $(\{\text{send}\}_2, \{\text{send}, \text{receive}\}_4)$. If the sets different from ACT in a descriptor A are exactly the sets for a group G, we will write $\langle\langle A\rangle\rangle$ as a shorthand for $\langle\langle A, G\rangle\rangle$. Braces around tuples and sets inside $\langle\langle\rangle\rangle$ will sometimes be dropped. For example,

$$\langle\langle \text{send}_4\rangle\rangle \bigcirc \phi$$

is shorthand for

$$\langle\langle(\{\text{send}\}, \{4\})\rangle\rangle \bigcirc \phi$$

Finally, $\langle\langle G\rangle\rangle$ is used as shorthand for $\langle\langle \text{ACT}^k, G\rangle\rangle$.

4.2. *Expressiveness and Examples*

The following are informal readings of formulae with some of the instances of the new operators. Let $a, b \in \mathrm{ACT}$, $X \subseteq \mathrm{ACT}$.

(1) $\langle\langle a_i \rangle\rangle \bigcirc \phi$: ϕ is a necessary consequence of i using action a.
(2) $\langle\langle a_i \rangle\rangle \Diamond \phi$: by starting with action a, i can eventually achieve ϕ.
(3) $[\![a_i]\!] \bigcirc \phi$: ϕ is a possible consequence of i using action a.
(4) $\langle\langle X_i \rangle\rangle \bigcirc \phi$: for some action in X, ϕ is a necessary consequence of i using the action.
(5) $[\![X_i]\!] \bigcirc \phi$: for each action in X, ϕ is a possible consequence of i using the action.
(6) $\langle\langle a_i, b_j \rangle\rangle \bigcirc \phi$: if i uses action a and j uses action b, ϕ is a necessary consequence.
(7) $\langle\langle a_i, X_j \rangle\rangle \bigcirc \phi$: if i uses action a, ϕ is the consequence of j using a particular action in X.
(8) $\langle\langle (a_i, X_j), \{i\} \rangle\rangle \bigcirc \phi$: if i uses action a, ϕ is the consequence of j using any arbitrary action in X.
(9) $\langle\langle X_j, \{i\} \rangle\rangle \bigcirc \phi$: i can enforce ϕ as long as j is restricted to actions X.

A concrete model checking example follows, before the expression of certain common and/or interesting properties of knowledge and action using the new operators is discussed.

4.2.1. *A Model Checking Example*

The following is a modified version of an example from (Jamroga and van der Hoek 2004). Consider a system with a single boolean variable, represented by the atomic proposition x (Figure 4). The system has two processes, the *controller o* and the *client l*. The client cannot observe the value of the variable, and always has the following available actions: set_true and set_false which are requests to the controller to set the value of the variable to *true* and *false*, respectively, and *switch* which switches the value of the variable. The controller can observe the value of the variable, and always has available actions *acc* and *rej*, which accepts or rejects, respectively, a possible request from the client. If the client performs *switch*, the variable is changed no matter what the controller does. The following hold in this system.

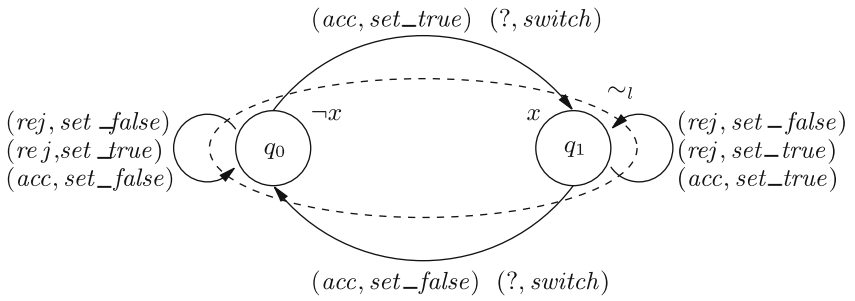

Figure 4. A variable controller/client example. ? means "any action".

(1) $\neg x \rightarrow \langle\langle \text{switch}_l \rangle\rangle \bigcirc x$. If the value of the variable is false, the client alone can change it with the switch action.
(2) $\langle\langle \text{set\_true}_l, \text{acc}_o \rangle\rangle \bigcirc x$. By performing the actions set_true and acc respectively, the client and the controller can together ensure that the variable will be true in the next state.
(3) $\langle\langle \{\text{set\_true}, \text{switch}\}_l \rangle\rangle \bigcirc x$. The client can always set the variable to true, if he guesses correctly between the set_true and switch actions (or finds out which one to use in any other way).
(4) $[\![\text{set\_true}_l]\!] \bigcirc x$. The variable being true is a possible consequence of the client performing the set_true action.
(5) $\langle\langle l \rangle\rangle \bigcirc x$, but there is no action a such that $\langle\langle a_l \rangle\rangle \bigcirc x$. The client *can* always ensure (in the ATL sense) that the variable will be true in the next state, but no single action is sufficient to ensure it.
(6) $[\![\text{set\_true}_l]\!] \bigcirc x$ and $\neg [\![c]\!] \bigcirc x$. The client cannot always avoid the variable being set to true in the next state by using the set_true action, but he can avoid it if his actions are not restricted.
(7) $q_0 \sim_l q_1$ and $\text{switch} \in d_l(q_0) = d_l(q_1)$, so switch in q_0 and switch in q_1 are subjectively the same action. But $q_0 \models \langle\langle \text{switch}_l \rangle\rangle \bigcirc x$ and $q_1 \models \langle\langle \text{switch}_l \rangle\rangle \bigcirc \neg x$; switch in q_0 and in q_1 are objectively different actions – they have different outcomes.
(8) $x \rightarrow \langle\langle \{\text{set\_true}, \text{set\_false}\}_l, o \rangle\rangle \bigcirc x$. If the variable is true, the controller can ensure that it stays true in the next state if the client is restricted to actions set_true and set_false.
(9) $\langle\langle \text{acc}_o, l \rangle\rangle \bigcirc x$ and $\langle\langle \text{acc}_o, l \rangle\rangle \bigcirc \neg x$. If the controller is restricted to using the acc action, the client can alone set the value of the variable in the next state.

4.2.2. *Available Actions*
The fact that an action $a \in \mathrm{ACT}$ is currently available to agent i can be expressed by

(21) $\quad \langle\langle a_i \rangle\rangle \bigcirc \top$

and the fact that a is *not* currently available to i by

(22) $\quad [\![a_i]\!] \bigcirc \bot.$

In ATL, $\langle\langle G \rangle\rangle \bigcirc \top$ and $\neg\langle\langle G \rangle\rangle \bigcirc \bot$ hold (in fact, they are taken as axioms in the axiomatization of ATL by Goranko and Van Drimmelen 2003), because of the requirement that there must be at least one action available for each agent in each state ($d_a(q) \neq \emptyset$). These axioms still hold in ATL-A (where they are written $\langle\langle \mathrm{ACT}^{|G|}, G \rangle\rangle \bigcirc \top$ and $\neg\langle\langle \mathrm{ACT}^{|G|}, G \rangle\rangle \bigcirc \bot$ respectively) but, e.g., $\langle\langle (A_1, \ldots, A_{|G|}), G \rangle\rangle \bigcirc \top$ does not hold for *arbitrary* non-empty A_js containing only non-available actions.

Formally, given $a \in \mathrm{ACT}$, it is easy to see that (21) and (22) hold in q iff $a \in d_i(q)$ and $a \notin d_i(q)$, respectively.

4.2.3. *Epistemic Pre-Conditions for Action*
A theory about action must explain when an agent can achieve a particular goal by performing a particular action. In (Moore, 1984), an agent i can use an action a to achieve a goal ϕ, $\mathrm{can}(i, a, \phi)$, iff (1) the goal will result from the agent performing the action and (2) the agent knows that the goal will be the result of performing the action.

In order to discuss how different ATEL/ATEL-A expressions capture this notion, consider a safe with combination c and let dial^x be the action of dialing the combination x and open be the proposition that the safe is open. For simplicity, we assume that all other pre-conditions for opening the safe are satisfied, so that the safe will be opened if an agent dials c. The formula

$$\langle\langle i \rangle\rangle \bigcirc \mathrm{open}$$

expresses the fact that i has a strategy to ensure that the safe will be open in the next state. However, this formula does not express the fact that a purposeful agent can open the safe if he wants to – he may consider worlds possible where he cannot open the safe at all, and even if he knew that he *could* open the safe he would not necessarily know *how*. In other words, i can open the safe if he guesses

the right combination, but this is not necessarily known to him. A second attempt is

$$\langle\langle \text{dial}_i^c \rangle\rangle \bigcirc \text{open}$$

which expresses the fact that the safe is opened if i dials c, but this is not necessarily known to him: he may consider it possible that other combinations than c will open the safe.

$$K_i \langle\langle i \rangle\rangle \bigcirc \phi$$

expresses that i knows that he can open the safe, but not that he necessarily knows how to do it. Finally,

$$K_i \langle\langle \text{dial}_i^c \rangle\rangle \bigcirc \text{open}$$

expresses that i can open the safe by dialing c, and that he knows this.

Jamroga and van der Hoek (2004) note the similarity between "knowing that there exist some strategy that will solve the problem" and "for some strategy, knowing that it will solve the problem" and the notion of *de dicto* and *de re* formulae as used in quantified modal logic (see, e.g., (Hughes and Cresswell, 1996)), and call the former "having a strategy *de dicto*" and the latter "having a strategy *de re*". In these terms the two latter formulae above express having a strategy *de dicto* and *de re*, respectively. Knowing *de re* that the action will satisfy the goal (having a *rigid designator* for the action) is exactly what Moore requires.

Thus, the fact that agent i can achieve a goal ϕ by performing an action a in the sense discussed by Moore, $\text{can}(i, a\,\phi)$, can in ATEL-A be defined as

$$\text{can}(i, a, \phi) \equiv K_i \langle\langle a_i \rangle\rangle \bigcirc \phi$$

$\text{can}(i, a, \phi) \rightarrow \langle\langle a_i \rangle\rangle \bigcirc \phi$ holds by virtue of the S5 properties of knowledge. Note that the other expressions given above are interesting in their own right, as long as it is clear precisely what they mean. These examples show that ATEL-A can express a range of subtly different facts about knowledge and action.

Van der Hoek and Wooldridge (2003) discuss general knowledge pre-conditions, such as "ψ is a necessary knowledge pre-condition for i having the ability to bring about ϕ":

$$\langle\langle i \rangle\rangle \bigcirc \phi \rightarrow K_i \psi.$$

This formula does not express ability in the sense of Moore precisely. For example, if the proposition comb^c is true in the worlds in which the combination of the safe is c, then $\langle\langle i\rangle\rangle \bigcirc \text{open} \rightarrow K_i \text{comb}^c$ expresses the fact that i cannot even *guess* the right action for opening the safe without knowing the combination first. A formula such as $\text{can}(i, a, \phi) \rightarrow K_i \text{comb}^c$ captures the intention better.

4.2.4. *Epistemic Post-Conditions for Action*
Epistemic consequences of action is the second main interaction between knowledge and action, according to Moore. The epistemic consequences of action can be expressed by ATEL-A formulae such as

$$\langle\langle a_i\rangle\rangle \bigcirc K_i \phi.$$

(after performing action a agent i will know ϕ), or

$$K_i \langle\langle a_i\rangle\rangle \bigcirc K_i \phi,$$

(agent i knows that he will know ϕ after performing a).

4.2.5. *Complete Knowledge about Actions*
We can now express the semantic condition (3) as a knowledge pre-condition about available actions: all available actions must be known.

THEOREM 2. The schema

$$(23) \quad \langle\langle a_i\rangle\rangle \bigcirc \top \rightarrow K_i \langle\langle a_i\rangle\rangle \bigcirc \top$$

defines the property (3).

Proof. Let $S = (k, Q, \Pi, \pi, \text{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ be a model of (23), and let q, q' be such that $q \sim_i q'$. If $a \in d_i(q)$ then $S, q \models \langle\langle a_i\rangle\rangle \bigcirc \top$, so $S, q' \models \langle\langle a_i\rangle\rangle \bigcirc \top$ by (23) i.e., $a \in d_i(q')$. If $a \in d_i(q')$, it follows similarly that $a \in d_i(q)$ since $q' \sim_i q$ by symmetry.

Conversely, let $S = (k, Q, \Pi, \pi, \text{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ be a CEGS where the frame property (3) holds, $q \in Q$ and a be such that $S, q \models \langle\langle a_i\rangle\rangle \bigcirc \top$. Let $q \sim_i q'$. $a \in d_i(q')$ by (3) since $a \in d_i(q)$, so $S, q' \models \langle\langle a_i\rangle\rangle \bigcirc \top$. Thus, $S, q \models K_i \langle\langle a_i\rangle\rangle \bigcirc \top$.

Thus, (23) defines the class of (3)-models, and, since (3) is a frame property, also the class of (3)-frames. □

The other direction of (23), $K_i\langle\langle a_i\rangle\rangle \bigcirc \top \rightarrow \langle\langle a_i\rangle\rangle \bigcirc \top$, if an agent knows that an action is available then it is available, holds by virtue of the S5 properties of knowledge.

4.3. *Properties*

Properties of ATL-A involving the new operators are now discussed. Of course, properties of the ATL operator $\langle\langle G\rangle\rangle$ do not carry over to the ATL-A operator $\langle\langle A, G\rangle\rangle$ for arbitrary A, see e.g., Section 4.2.2. Some properties of $\langle\langle G\rangle\rangle$ have similar $\langle\langle A, G\rangle\rangle$ counterparts; such as, e.g., the properties (1) and (2) (p. 380) and the definition of $\langle\langle A, G\rangle\rangle\Box$ and $\langle\langle A, G\rangle\rangle\mathcal{U}$.

LEMMA 2.

(1) $S, q \models \langle\langle \mathrm{ACT}^k, G\rangle\rangle \bigcirc \phi$ in ATL-A (ATEL-A) if and only if $S, q \models \langle\langle G\rangle\rangle \bigcirc \phi$ in ATL (ATEL)
(2) $S, q \models \langle\langle \mathrm{ACT}^k, G\rangle\rangle\Box\phi$ in ATL-A (ATEL-A) if and only if $S, q \models \langle\langle G\rangle\rangle\Box\phi$ in ATL (ATEL)
(3) $S, q \models \langle\langle \mathrm{ACT}^k, G\rangle\rangle\phi_1\mathcal{U}\phi_2$ in ATL-A (ATEL-A) if and only if $S, q \models \langle\langle G\rangle\rangle\phi_1\mathcal{U}\phi_2$ in ATL (ATEL)

Proof. 1 follows immediately from the semantic definitions, 2 and 3 follow from 1 by the properties (1) and (2) in section 2.2. □

The above lemma expresses the fact that the new ATL-A operators are generalizations of the traditional ATL operators.

LEMMA 3. The following are valid (T is any temporal operator).

(1) $\langle\langle (A_1, \ldots, A_k), G\rangle\rangle T\phi \rightarrow \langle\langle (A'_1, \ldots, A'_k), G\rangle\rangle T\phi$ when

$$A_j \subseteq A'_j \;\; j \in G,$$
$$A'_j \subseteq A_j \;\; j \notin G,$$

(2) $\langle\langle (A_1, \ldots, A_k), G\rangle\rangle T\phi \leftrightarrow \bigvee_{A'_j \in X_j} \langle\langle (A'_1, \ldots, A'_k), G\rangle\rangle T\phi$ where

$$X_j = \begin{cases} \{\{a_j\} : a_j \in A_j\}, & j \in G \\ \{A_j\}, & j \notin G \end{cases}$$

(3) $\langle\langle (A_1, \ldots, \{a\}_j, \ldots, A_k), G\rangle\rangle T\phi$
$\leftrightarrow \langle\langle (A_1, \ldots, \{a\}_j, \ldots, A_k), G \setminus \{j\}\rangle\rangle T\phi$

Proof. Immediate from the semantic definitions. □

The first property in the above Lemma 3 is a generalization of action descriptors: a coalition cannot achieve less if they are given more, and their opponents less, action possibilities.

The second property is a definition of what it means to achieve something with a set of actions, in terms of what it means to achieve something with a single action. An example instance is

$$\langle\langle A_i \rangle\rangle \bigcirc \phi \leftrightarrow \bigvee_{a \in A} \langle\langle a_i \rangle\rangle \bigcirc \phi$$

agent i can alone achieve ϕ with the actions A iff there is an action in A which he can achieve ϕ with.

The third property says that restricting an agent j to a single action implies that any coalition can achieve the same with or without j.

4.4. *Model Checking*

One of the reasons for the popularity of the branching time logics CTL and ATL is the tractability of the model checking problem. For both logics, the problem can be answered in time polynomial in the size of the formula and the size of the structure (Alur et al. 2002), and model checkers such as SMV (McMillan 1993) and MOCHA (Alur et al. 1998) have been implemented.

Van der Hoek and Wooldridge (2003) have shown that tractability of model checking carries over to ATEL, by constructing an algorithm. In this section that algorithm is generalized to ATEL-A. It is shown that the resulting algorithm still runs in polynomial time, and thus that the increase in expressiveness from ATEL to ATEL-A, does not come at the expense of increased computational complexity of model checking.

The algorithm for ATEL, henceforth called "the original algorithm," is defined recursively over the structure of the formula, and the corresponding correctness and complexity proofs by induction over the structure, and since ATEL-A is an extension of ATEL we can just extend the definition and the proofs with cases for the new clauses in the language. The extensions are straightforward.

The original algorithm is implemented as a function

- $eval(\phi, S)$ which, given an ATEL formula ϕ and a CEGS S, returns the set of states in S satisfying ϕ.

which in turn relies on the following function:

- $pre(S, G, Q')$, which, given a CEGS S, a group of agents G and a set of states Q', returns the set of states in which G can cooperate to make the next state of the system be in Q'.

The extended algorithm makes use of the following modified system.

DEFINITION 4. ($S(A)$) Given a CEGS $S = (k, Q, \Pi, \pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ and an action descriptor $A = (A_1, \ldots, A_k)$, the CEGS

$$S(A) = (k, Q \cup \{\hat{q}\}, \Pi, \hat{\pi}, \mathrm{ACT} \cup \{\hat{a}\}, \hat{d}, \hat{\delta}, \sim_1, \ldots, \sim_k)$$

where $\hat{q} \notin Q$ and $\hat{a} \notin \mathrm{ACT}$ and

- $\hat{\pi}(q) = \emptyset$ if $q = \hat{q}$, $\hat{\pi}(q) = \pi(q)$ otherwise,
- $\hat{d}_i(q) = \{\hat{a}\}$ if $q = \hat{q}$ or $d_i(q) \cap A_i = \emptyset$, $\hat{d}_i(q) = d_i(q) \cap A_i$ otherwise,
- $\hat{\delta}(q, \vec{a}) = \hat{q}$ if $q = \hat{q}$ or $a_i = \hat{a}$ for some i, $\hat{\delta}(q, \vec{a}) = \delta(q, \vec{a})$ otherwise.

Clearly, $S(A)$ is a proper CEGS. Intuitively, $S(A)$ is S when the agents are restricted to A with the addition of the state $\hat{q}$ which the agents go to (using action $\hat{a}$) when one or more of them cannot act.

In Figure 5, the extended algorithm is presented by the case for the new language clause.

THEOREM 3. $eval(\phi, S)$ (Figure 5) terminates and returns the set $\{q : S, q \models \phi\}$.

Proof. Since the original algorithm and $pre(\ldots)$ terminates, the algorithm terminates.

The proof of correctness of $eval(\phi, S)$ for ATEL-formulae ϕ is by induction over the structure of ϕ, thus it suffices to show that

Algorithm: $eval(\phi, S)$, where $S = (k, Q, \Pi, \pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_n)$, returns a subset of Q

if $\phi = \langle\langle (A_1, \ldots, A_k), G \rangle\rangle \bigcirc \psi$ **then**
  return $pre(S(A_1, \ldots, A_k), G, eval(\psi, S))$
else
  {the other cases are as in the original algorithm}
end if

Figure 5. A model checking algorithm for ATEL (van der Hoek and Wooldridge 2003) adapted to ATEL-A. Only the operation for the new language clause is shown.

$eval(\phi, S) = \{q : S, q \models \phi\}$ for the new case in Figure 5 assuming correctness for simpler formulae.

If $S, q \models \langle\langle A, G\rangle\rangle \bigcirc \psi$, let $\vec{a} \in \times_{i\in\Sigma}(d_i(q) \cap A_i)$ be as in the definition; thus $d_i(q) \cap A_i \neq \emptyset$ for all i, and $a_j \in \hat{d}_j(q)$ since $q \neq \hat{q}$. Let $b_i \in \hat{d}_i(q)$, $i \notin G$. $b_i \in d_i(q) \cap A_i$ since $q \neq \hat{q}$. Let $b_i \in d_i(q) \cap A_i$ be arbitrary for $i \in G$, and let $\vec{a}' = \vec{b}[\vec{a}/G]$. $S, \delta(q, \vec{a}') \models \psi$ and since $q \neq \hat{q}$ and $a'_i \neq \hat{a}$ for all i, $\hat{\delta}(q, \vec{a}') = \delta(q, \vec{a}')$ and, by the induction hypothesis for ψ, G can thus cooperate by using actions a_i $(i \in G)$ so that no matter which actions the other agents use, $S(A)$ will go to a state in $eval(\psi, S)$.

For the other direction, assume that $q \in pre(S(A), G, eval(\psi, S))$. Thus, there are $a_j \in \hat{d}(q)$, $j \in G$, such that the outcome will be in $eval(\psi, S)$ no matter which actions $\Sigma \setminus G$ use. Let $a_j \in \hat{d}_j(q)$ be arbitrary for $j \notin G$. Since $\hat{\delta}(q, \vec{a}) \in eval(\psi, S)$ and $\hat{q} \notin eval(\psi, S) =$ (by the ind. hyp.) $\{q : S, q \models \psi\}$, $a_j \neq \hat{a}$ for all j. Clearly, $S(A)$ cannot go from $\hat{q}$ to any state in $eval(\psi, S)$, so $q \neq \hat{q}$. Thus, $\vec{a} \in \times_{i\in\Sigma}(d_i(q) \cap A_i)$. Let $\vec{b} \in \times(d_i(q) \cap A_i)$; $b_i \in \hat{d}(q)$ since $d_i(q) \cap A_i$ is non-empty, and let $\vec{a}' = \vec{b}[\vec{a}/G]$. The outcome in $S(A)$ of $\vec{a}'$ will be $\hat{\delta}(q, \vec{a}') \in eval(\psi, S)$, and since $q \neq \hat{q}$ and both $a_i \neq \hat{a}$ and $b_i \neq \hat{a}$ for any i, $\delta(q, \vec{a}') = \hat{\delta}(q, \vec{a}')$, and thus $S, \delta(q, \vec{a}') \models \psi$. □

THEOREM 4. The model checking problems for ATL-A and ATEL-A are PTIME-complete.

Proof. The result follows from the PTIME-completeness of model checking for ATL (Alur et al. 2002) and ATEL (van der Hoek and Wooldridge 2003). It is easy to see that $S(A_1, \ldots, A_k)$ can be constructed in time polynomial in the size of $S, A_1, \ldots, A_k$. For example, the time to compute $\hat{d}$ can be expressed as a polynomial in k, $|Q|$, $|d_i(q)|$ for each q and i and $|A_i|$ for each i. Since $pre(S(A), G, Q')$ can be computed in time polynomial in the size of $S(A), G, Q'$ (van der Hoek and Wooldridge 2003) and the size of $S(A)$ is a polynomial in the size of S, $pre(S(A), G, Q')$ can be computed in time polynomial in the size of S, G, Q', A by constructing $S(A)$. Each occurrence of a descriptor A in the formula is a part of the input, and is used in only one *pre*-computation by the algorithm. Thus, since $eval(\phi, S)$ can be computed in polynomial time for ATEL formulae ϕ, so it can for ATEL-A formulae. The problem for ATEL-A is PTIME-hard since ATEL-A extends ATEL.

PTIME-completeness for the ATL-A problem follows since ATL-A is included in ATEL-A, and ATL is included in ATL-A. □

5. CONCLUSIONS

ATEL lacks the expressive power to express certain interesting properties about the interaction between action and knowledge. The proposed generalization of the ATL operators is both syntactically and semantically simple and computationally tractable, but gives a rich syntax for statements about actions. Formally, the fact that ATEL-A is more expressive than ATEL is shown by Theorems 1 and 2.

Several properties of ATEL-A have been discussed, but a complete axiomatization is left for future work. Implementation of the semantics of the new operators in a model checker such as MOCHA should be straightforward.

In future work the expressiveness of ATEL-A and the logics ATOL/ATEL-R* mentioned in note 3 should be compared. An interesting direction would be to introduce the generalized operators into ATOL/ATEL-R*.

ATL-A can be viewed as allowing, in addition to atomic actions ($\langle\langle\langle a_i\rangle\rangle\rangle$), PDL-like composition of actions into more complex ones, viz. concurrency ($\langle\langle\langle a_i, b_j\rangle\rangle\rangle$) and choice ($\langle\langle\langle\{a, b\}_i\rangle\rangle\rangle$). Another interesting direction is to also allow a sequencing operation on actions in order to express *plans*, as mentioned in note 4. A key point in a logic about knowledge and plans would be to give an account of what it means to know that a plan can be used to achieve a goal. ATOL/ATEL-R* may be the best candidates for such an extension with plan-expressions.

APPENDIX

Proof of Lemma 1. Let $F=(k, Q, \Pi, \mathrm{ACT}, d, \delta, \sim_1, \ldots, \sim_k)$ and $F'=(k, Q, \Pi, \mathrm{ACT}, d', \delta', \sim_1, \ldots, \sim_k)$ be as in the lemma, and let π be a labeling function. In Section 2, the set of strategies $\mathrm{Str}(G)$ and the outcome function out for a given C(E)GS were defined. Both these notions are properties of frames rather than models, and in order to talk about them for both frame F and F', they will be subscripted by the frame. For example, $\mathrm{Str}_F(G)$ is the set of strategies for group G in frame F, and out_F is the output function on frame F.

First, two intermediate results are shown. For these, let $G \subseteq \Sigma$ be arbitrary. The first one is:

$$(24) \quad \vec{f}_G \in \mathrm{Str}_F(G) \Rightarrow \exists_{\vec{f}'_G \in \mathrm{Str}_{F'}(G)} \forall_{q \in Q} \mathrm{out}_{F'}(q, \vec{f}'_G) \subseteq \mathrm{out}_F(q, \vec{f}_G).$$

Let $\vec{f}_G \in \mathrm{Str}_F(G)$, and define $\vec{f}'_G$ as follows, for $j \in G$:

$$f'_j(q_0 \cdots q_m) = \begin{cases} b, & \text{if } j = i, q_m = q', f_j(q_0 \cdots q_m) = a, \\ f_j(q_0 \cdots q_m), & \text{otherwise.} \end{cases}$$

It is easy to see that $\vec{f}'_G \in \mathrm{Str}_{F'}(G)$. Let $q \in Q$ and assume that $\lambda \in \mathrm{out}_{F'}(q, \vec{f}'_G)$. We must show that $\lambda \in \mathrm{out}_F(q, \vec{f}_G)$ (recall the definition of out from Section 2). Clearly, $\lambda[0] = q$. Let $j \geq 0$. There is an $\vec{a}' \in D'(\lambda[j])$ (where $D'(q)$ is the set of joint actions in q in F') such that a) $\forall_{l \in G} a'_l = f'_l(\lambda[0, j])$ and b) $\delta'(\lambda[j], \vec{a}') = \lambda[j+1]$. Let $\vec{a}$ be $\vec{a}'$, except that each agent $l \in G$ uses f_l instead of f'_l:

$$a_l = \begin{cases} f_l(\lambda[0, j]), & \text{if } l \in G, \\ a'_l, & \text{otherwise.} \end{cases}$$

$\vec{a} \in D(\lambda[j])$: if $l \in G$ then $a_l = f_l(\lambda[0, j]) \in d_l(\lambda[j])$ and if $l \notin G$ then $a_l = a'_l \in d'_l(\lambda[j]) \subseteq d_l(\lambda[j])$. We must show that a) $a_l = f_l(\lambda[0, j])$ when $l \in G$, and b) $\delta(\lambda[j], \vec{a}) = \lambda[j+1]$. a) holds by definition of a_l. For b), it suffices to show that $\delta(\lambda[j], \vec{a}) = \delta(\lambda[j], \vec{a}')$, since $\delta(\lambda[j], \vec{a}') = \delta'(\lambda[j], \vec{a}') = \lambda[j+1]$. If $\vec{a} = \vec{a}'$ we are done, so let m be such that $a_m \neq a'_m$. The only possibility for this is that $m \in G$ in which case $a_m = f_m(\lambda[0, j])$ and $a'_m = f'_m(\lambda[0, j])$ and since $f_m(\lambda[0, j]) \neq f'_m(\lambda[0, j])$, the only possibility is that $m = i$, $\lambda[j] = q'$, $a'_m = f'_m(\lambda[0, j]) = b$ and $a_m = f_m(\lambda[0, j]) = a$. Thus, $\vec{a} = \vec{a}[a/m]$ and $\vec{a}' = \vec{a}[b/m]$, and $\delta(\lambda[j], \vec{a}) = \delta(\lambda[j], \vec{a}')$ follows by equivalence of a and b in q'. Thus, $\lambda \in \mathrm{out}_F(q, \vec{f}_G)$, and (24) holds for any G.

The second intermediate result is dual to (24):

$$(25) \quad \vec{f}'_G \in \mathrm{Str}_{F'}(G) \Rightarrow \exists_{\vec{f}_G \in \mathrm{Str}_F(G)} \forall_{q \in Q} \mathrm{out}_F(q, \vec{f}_G) \subseteq \mathrm{out}_{F'}(q, \vec{f}'_G)$$

Let $\vec{f}'_G \in \mathrm{Str}_{F'}(G)$ and define

$$\vec{f}_G = \vec{f}'_G$$

It is easy to see that $\vec{f}_G \in \mathrm{Str}_F(G)$: $f_l(q_0 \cdots q_m) \in d'_l(q_m) \subseteq d_l(q_m)$. Let $q \in Q$ and assume that $\lambda \in \mathrm{out}_F(q, \vec{f}_G)$. We must show that $\lambda \in \mathrm{out}_{F'}(q, \vec{f}'_G)$. Clearly, $\lambda[0] = q$. Let $j \geq 0$. There is an $\vec{a} \in D(\lambda[j])$

such that a) $\forall_{l \in G} a_l = f_l(\lambda[0, j])$ and b) $\delta(\lambda[j], \vec{a}) = \lambda[j+1]$. Let $\vec{a}'$ be defined as follows:

$$a'_l = \begin{cases} b, & \text{if } l = i, \lambda[j] = q', a_l = a \quad \text{(case A)}, \\ a_l, & \text{otherwise} \qquad\qquad\qquad\quad \text{(case B)} \end{cases}$$

$\vec{a}' \in D'(\lambda[j])$: in case A, $a_i = a \in d_i(q')$, and $a'_l = b \in d_i(q')$ by equivalence of a and b for i in q', so $a'_l = b \in d'_i(q')$. In case B if not both $l = i$ and $\lambda[j] = q'$ then $a'_l = a_l \in d_l(\lambda[j]) = d'_l(\lambda[j])$. In case B if $l = i$ and $\lambda[j] = q'$ then $a_l \neq a$ and $a'_l = a_l \in d_l(\lambda[j]) \setminus \{a\} = d'_l(\lambda[j])$. We must show that a) $a'_l = f'_l(\lambda[0, j])$ when $l \in G$ and b) $\delta'(\lambda[j], \vec{a}) = \lambda[j+1]$. For (a), let $l \in G$ and first consider case A in the definition of a'_l. This case is impossible since $l \in G$: $a_i = a = f_i(\lambda[0, j]) = f'_i(\lambda[0, j])$, so $a \in d'_i(q')$ which is impossible. So a'_l must be defined by case B, in which case $a'_l = a_l = f_l(\lambda[0, j]) = f'_l(\lambda[0, j])$. For (b), it suffices to show that $\delta(\lambda[j], \vec{a}') = \delta(\lambda[j], \vec{a})$, since $\delta'(\lambda[j], \vec{a}') = \delta(\lambda[j], \vec{a}')$ and $\delta(\lambda[j], \vec{a}) = \lambda[j+1]$. If $\vec{a} = \vec{a}'$ we are done, so let m be such that $a_m \neq a'_m$. The only possibility is that $\lambda[j] = q'$, $m = i$, $a_i = a$ and $a'_i = b$. Then $\vec{a} = \vec{a}[a/i]$ and $\vec{a}' = \vec{a}[b/i]$, and $\delta(\lambda[j], \vec{a}') = \delta(\lambda[j], \vec{a})$ follows by equivalence of a and b for i in q. Thus, $\lambda \in \text{out}_{F'}(q, \vec{f}'_G)$, and (25) holds for any G.

Finally,

$$(26) \qquad \forall_{q \in Q} \left((F, \pi), q \models \phi \Leftrightarrow (F', \pi), q \models \phi \right)$$

can be shown for all ϕ by structural induction. Only two cases are shown here; the propositional cases are trivial and the other temporal cases are similar to the one shown.

$\phi = \langle\langle G \rangle\rangle \phi_1 \mathcal{U} \phi_2$: $(F, \pi), q \models \phi$ iff $\exists_{\vec{f}_G \in \text{Str}_F(G)} \forall_{\lambda \in \text{out}_F(q, \vec{f}_G)} \exists_{j \geq 0}$ $(F, \pi), \lambda[j] \models \phi_2$ and $\forall_{0 \leq o < j} (F, \pi), \lambda[o] \models \phi_1$ iff, by the induction hypotheses, $\exists_{\vec{f}_G \in \text{Str}_F(G)} \forall_{\lambda \in \text{out}_F(q, \vec{f}_G)} \exists_{j \geq 0} (F', \pi), \lambda[j] \models \phi_2$ and $\forall_{0 \leq o < j} (F', \pi), \lambda[o] \models \phi_1$ iff, by (24) for the direction to the right and (25) for the direction to the left, $\exists_{\vec{f}'_G \in \text{Str}_{F'}(G)} \forall_{\lambda \in \text{out}_{F'}(q, \vec{f}'_G)} \exists_{j \geq 0}$ $(F', \pi), \lambda[j] \models \phi_2$ and $\forall_{0 \leq o < j} (F', \pi), \lambda[o] \models \phi_1$ iff $(F', \pi), q \models \phi$.

$\phi = K_i \psi$: $(F, \pi), q \models \phi$ iff for all $q' \in Q$ such that $q \sim_i q'$ $(F, \pi), q' \models \psi$ iff, by the induction hypothesis, for all $q \sim_i q'$ $(F', \pi), q' \models \psi$ iff $(F', \pi), q \models \phi$.

□

ACKNOWLEDGEMENTS

I have discussed ideas in this paper with Wojciech Jamroga, Valentin Goranko and Michal Walicki. I would also like to thank the anonymous referees for their very valuable comments.

NOTES

[1] In addition, ATEL has operators for common knowledge and "everyone knows." These are not used in the following and are therefore not presented here.
[2] They are used as axioms for ATL by Goranko and van Drimmelen (2003), and it is easy to see that they also hold for ATEL.
[3] Particularly, Jamroga and van der Hoek (2004) introduce the logics alternating-time temporal observational logic (ATOL) and ATEL-R* in order to deal with the mentioned and related problems. In these logics (3) is taken as a semantic assumption, so the question of expressing the property in these logics is irrelevant. The expressiveness of action properties in these logics is nevertheless an interesting direction for further work; see Section 5.
[4] It is assumed here that this formula fits the intention of the schema (6) since in (van der Hoek and Wooldridge 2003) T is said to be "a temporal operator" and $\mathcal{U}$ is a temporal operator.
[5] See (Ågotnes 2004) for AETS counter models corresponding to the CEGS ones given in Section 3.1.
[6] A possibility for further work is to allow *sequences* of actions, like in (Moore 1984; Morgenstern 1986, 1987) in order to be able to express *plans* in ATL.
[7] The definition has been made as succinct as possible, but may be confusing to read (particularly because the arbitrary (existence ensured by $\vec{a}$) actions b_i for $i \in G$ are not actually used). Here is an equivalent, as the reader can verify, formulation, where $G = \{g_1, \ldots, g_m\}$ and $\Sigma \backslash G = \{s_1, \ldots, s_p\}$:

$$S, q \models \langle\langle A, G\rangle\rangle \bigcirc \phi \Leftrightarrow A^{\times} \cap D(q) \neq \emptyset \text{ and}$$
$$\exists_{c_{g_1} \in d_{g_1}(q) \cap A_{g_1}} \cdots \exists_{c_{gm} \in d_{gm}(q) \cap A_{gm}} \forall_{c_{s_1} \in d_{s_1}(q) \cap A_{s_1}} \cdots \forall_{c_{sp} \in d_{sp}(q) \cap A_{sp}} S, \delta(q, \vec{c}) \models \phi.$$

REFERENCES

Ågotnes, T.: 2004, 'A Note on Syntactic Characterization of Incomplete Information in ATEL', in *Proceedings of the First Workshop on Knowledge and Games (KAG 2004)*, Liverpool, UK.

Alur, R., T.A. Henzinger, and O. Kupferman: 1997, 'Alternating-time Temporal Logic', in *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, Miami Beach, FL, pp. 100–109.

Alur, R., T.A. Henzinger, and O. Kupferman: 1999, 'Alternating-time Temporal Logic', in *Compositionality: The Significant Difference*, Lecture Notes in Computer Science, 1536, Springer-Verlag, Berlin, pp. 23–60.

Alur, R., T.A. Henzinger, and O. Kupferman: 2002, 'Alternating-time Temporal Logic', *Journal of the ACM* **49**, 672–713.

Alur, R., T.A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Tasiran: 1998, 'MOCHA: Modularity in Model Checking', in *Computer Aided Verification*, pp. 521–525.

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

Goranko, V. and G. van Drimmelen: 2003, 'Decidability and Complete Axiomatization of the Alternating-time Temporal Logic', Submitted.

Halpern, J.Y. and R. Fagin: 1989, 'Modelling Knowledge and Action in Distributed Systems', *Distributed Computing* **3**(4), 159–177.

Harel, D.: 1984, 'Dynamic Logic', in D. Gabbay and F. Guenthner (eds.) *Handbook of Philosophical Logic, Volume II: Extensions of Classical Logic*, Vol. 165 of *Synthese Library*, D. Reidel Publishing Co., Dordrecht, pp. 497–604.

Henriksen, J.G. and P.S. Thiagarajan: 1999, 'Dynamic Linear Time Temporal Logic', *Annals of Pure and Applied Logic* **96**, 187–207.

Hughes, G.E. and M.J. Cresswell: 1996, *A New Introduction to Modal Logic*, Routledge, London and New York.

Jamroga, W.: 2003, 'Some Remarks on Alternating Temporal Epistemic Logic', in *FAMAS03 – Formal Approaches to Multi-Agent Systems, Proceedings*, Warsaw, Poland, pp. 133–140.

Jamroga, W. and W. van der Hoek: 2004, 'Agents that Know How to Play', *Fundamenta Informaticae* **63**, 185–219.

McMillan, K.: 1993, *Symbolic Model Checking*, Kluwer Academic Publishers, Norwell, MA.

Meyer, J.-J.C. and W. van der Hoek: 1995, *Epistemic Logic for AI and Computer Science*, Cambridge University Press, Cambridge, UK.

Moore, R.C.: 1984, 'A Formal Theory of Knowledge and Action', in *Formal Theories of the Commonsense World*, Ablex Publishing Corp, Norwood, NJ.

Morgenstern, L.: 1986, 'A First Order Theory of Planning, Knowledge, and Action', in J. Y. Halpern (ed.) *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the First Conference*, Los Altos, CA, pp. 99–114, Morgan Kaufmann Publishers, Inc.

Morgenstern, L.: 1987, 'Knowledge Preconditions for Actions and Plans', in *Proceedings of the National Conference on Artificial Intelligence*. Seattle, WA, pp. 867–874. (Also published in *Readings in Distributed Artificial Intelligence*, Alan H. Bond and Les Gasser (eds.), pp. 151–158, Morgan Kaufmann, 1988).

van der Hoek, W., M. Roberts, and M.Wooldridge: 2004, 'Social Laws in Alternating Time: Effectiveness, Feasibility, and Synthesis', Technical Report ULCS-04-017, Computer Science Department, University of Liverpool.

van der Hoek, W. and M. Wooldridge: 2002, 'Tractable Multiagent Planning for Epistemic Goals', in *Proceedings of the First International Conference on Autonomous Agents and Multiagent Systems (AAAMAS-02)*, Bologna, Italy.

van der Hoek, W. and M. Wooldridge: 2003, 'Cooperation, Knowledge and Time: Alternating-time Temporal Epistemic Logic and its Applications', *Studia Logica* **75**, 125–157.

Department of Informatics
University of Bergen
PB. 7800, N-5020 Bergen
Norway
E-mail: agotnes@ii.uib.no