# A Temporal Logic of Strategic Knowledge

**Xiaowei Huang** and **Ron van der Meyden**
The University of New South Wales

**Abstract**

The paper presents an extension of temporal epistemic logic that adds "strategic" agents in a way that allows standard epistemic operators to capture what agents could deduce from knowledge of the strategies of some subset of the set of agents. A number of examples are presented to demonstrate the broad applicability of the framework, including reasoning about implementations of knowledge-based programs, game theoretic solution concepts and notions from computer security. It is shown that notions from several variants of alternating temporal epistemic logic can be expressed. The framework is shown to have a decidable model checking problem.

## Introduction

The design and study of distributed and multi-agent systems typically has to deal with agents who have a choice of action to perform, and have individual and possibly conflicting goals. This leads agents to act strategically, attempting to select their actions over time so as to guarantee these goals even in the face of other agents' behaviour. The choice of actions generally needs to be done on the basis of *imperfect* information concerning the state of the system.

These concerns have motivated the development of a variety of modal logics that aim to capture the essential notions of such settings. One of the earliest, dating from the 1980's was multi-agent epistemic logic (Halpern and Moses 1990; Parikh and Ramanujam 1985), which introduces modal operators that deal with imperfect information by providing a way to state what agents *know*. Combining such constructs with temporal logic (Pnueli 1977) constructs gives temporal-epistemic logics, which support reasoning about how agents' knowledge changes over time. Temporal-epistemic logic is an area about which a significant amount is now understood (Fagin et al. 1995).

Logics dealing with reasoning about strategies, which started to be developed in the same period (Parikh 1983), had a slower initial start, but have in recent years become the focus of intense study (Pauly 2002; Horty. 2001; Alur, Henzinger, and Kupferman 2002). Many subtle issues that arise concerning what agents know in settings where multiple agents act strategically. In the process of understanding these issues, there has been a proliferation of modal logics. In particular, for dealing with epistemic reasoning in strategic settings, there are multiple proposals, with different semantic and syntactic bases, for how to capture reasoning about the availability to groups of agents of strategies for achieving particular goals (Jonker 2003; Schobbens 2004; van Otterloo and Jonker 2005; Jamroga 2003; Jamroga and Ågotnes 2007).

We argue in this paper that this proliferation is unnecessary, and that an appropriate application of temporal epistemic logic, with a minor innovation, already has the power to deal with many issues of concern when dealing with epistemic reasoning in strategic settings.

In particular, whereas logics in this area frequently leave strategies implicit in the semantic modelling, we propose to work in an instance of a standard semantic framework for temporal epistemic logic, but with strategies explicitly represented in the semantic model. In fact, we are not the first to have applied this instance of the standard temporal-epistemic model (Halpern and O'Neill 2008). Our main innovation is small but, we claim, powerful: we introduce new agent names that refer to the strategies being used by the main players, and allow these new agent names to be included in (otherwise standard) operators for group knowledge.

We argue that this gives a logical approach with broad applicability. In particular, it can express many of the subtly different notions that have been the subject of proposals for alternating temporal epistemic logics. We demonstrate this by results that show how such logics can be translated into our setting. We also present a number of other examples including reasoning about possible implementations of knowledge-based programs, game theoretic solution concepts, and issues of concern in computer security. Moreover, as we show, our approach retains from alternating temporal epistemic logic the desirable property that model checking is decidable, in the case of an imperfect recall semantics for knowledge. We show that it is in fact PSPACE-complete, no more than the complexity of model checking the temporal logic LTL, on which we build, although we have a much richer expressiveness.

The structure of the paper is as follows. We first recall some standard definitions from temporal epistemic logic. We then present a semantic model (also standard) for the environments in which agents choose their actions. Building on this model, we show how to construct a model for temporal epistemic logic that builds in information about strategic behaviour, but which introduces the new "strategy" agents. The following sections deal with applications of this model. We show successively that it can express reasoning about implementations of knowledge-based programs, many notions that have been proposed in the area of alternating temporal epistemic logic, game theoretic solution concepts, and problems from computer security. Next, we show that the framework has a decidable model checking problem. We conclude with a discussion of related literature.

## Temporal Epistemic Logic

Suppose that we are working with a system in which there is a finite set $Ags$ of agent names. Let $Prop$ be a set of atomic propositions. The language CTL*K($Ags, Prop$) has the syntax:

$$\phi \equiv p \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid A\phi \mid \bigcirc\phi \mid \phi_1 U \phi_2 \mid D_G\phi \mid C_G\phi$$

where $p \in Prop$ and $G \subseteq Ags$. Intuitively, the formula $D_G\phi$ expresses that agents in $G$ have distributed knowledge of $\phi$, i.e., they could deduce $\phi$ if they pooled their information, and $C_G\phi$ says that $\phi$ is common knowledge to group $G$. The formulas $\bigcirc\phi$ and $\phi_1 U \phi_2$ express the linear time temporal logic notions "$\phi$ at the next moment of time" and "$\phi_1$ until $\phi_2$", respectively. The operator $A$ refers to branching temporal structure: $A\phi$ says that the formula $\phi$ holds for all possible evolutions from the present situation. Other operators can be obtained in the usual way, e.g., $\phi_1 \wedge \phi_2 = \neg(\neg\phi_1 \vee \neg\phi_2)$, $\Diamond\phi = (true U \phi)$, $\Box\phi = \neg\Diamond\neg\phi$, etc. For an agent $i \in Ags$ write $K_i\phi$ for $D_{\{i\}}\phi$, this expresses that agent $i$ knows the fact $\phi$. The notion of everyone in group $G$ knowing $\phi$ can then be expressed as $E_G\phi = \bigwedge_{i\in G} K_i\phi$. Using this, common knowledge $C_G\phi$ can be understood as the conjunction of $E_G^k\phi$ for all $k \geq 0$.

Semantics of the language is given in *interpreted systems*, which are tuples $\mathcal{I} = \langle\mathcal{R}, \pi\rangle$, where $\mathcal{R}$ is a set, called the set of *runs*, and $\pi : \mathcal{R} \times \mathbb{N} \to \mathcal{P}(Prop)$ is a propositional interpretation. It is required that there exist for each $i \in Ags$, a nonempty set $L_i$ of *local states of agent* $i$, such that $\mathcal{R}$ is a subset of the set of functions $r : \mathbb{N} \to \Pi_{i\in Ags} L_i$. Intuitively, this says that a run maps each moment of time to a *global state*, which gives a local state for each agent $i$. The usual presentation (Fagin et al. 1995), also includes a component in the global state for the state of the environment. We will handle this by including the special agent $e \in Ags$, representing the environment, when required. For $n \leq m$ write $r[n \ldots m]$ for the sequence $r(n)r(n+1)\ldots r(m)$. Elements of $\mathcal{R} \times \mathbb{N}$ are called the *points* of $\mathcal{I}$. For each agent $i \in Ags$, we write $r_i(m)$ for the $i$-component of $r(m)$, and define an equivalence relation on points by $(r, m) \sim_i (r', m')$ if $r_i(m) = r'_i(m')$. For $G \subseteq Ags$, we define the equivalence relation $\sim_G$ on points by $(r, m) \sim_G (r', m')$ if $(r, m) \sim_i (r', m')$ for all $i \in G$. Note that $\sim_\emptyset$ is then the universal relation on $\mathcal{R} \times \mathbb{N}$, intuitively representing an agent who knows nothing, and $\sim_{Ags}$ is an agent with complete information about the global state. In particular, it follows for runs $r, r' \in \mathcal{R}$ that if $(r, m) \sim_{Ags} (r', m)$ for all $m \in \mathbb{N}$, then $r = r'$.

The semantics of the language CTL*K is given by a relation $\mathcal{I}, (r, m) \models \phi$, representing that formula $\phi$ holds at point $(r, m)$ of the interpreted system $\mathcal{I}$. This defined inductively on the structure of the formula $\phi$, as follows:

- $\mathcal{I}, (r, m) \models p$ if $p \in \pi(r, m)$;
- $\mathcal{I}, (r, m) \models \neg\phi$ if not $\mathcal{I}, (r, m) \models \phi$;
- $\mathcal{I}, (r, m) \models \phi \wedge \psi$ if $\mathcal{I}, (r, m) \models \phi$ and $\mathcal{I}, (r, m) \models \psi$;
- $\mathcal{I}, (r, m) \models A\phi$ if $\mathcal{I}, (r', m) \models \phi$ for all $r' \in \mathcal{R}$ with $r[0 \ldots m] = r'[0 \ldots m]$;
- $\mathcal{I}, (r, m) \models \bigcirc\phi$ if $\mathcal{I}, (r, m+1) \models \phi$;
- $\mathcal{I}, (r, m) \models \phi U \psi$ if there exists $m' \geq m$ such that $\mathcal{I}, (r, m') \models \psi$ and $\mathcal{I}, (r, k) \models \phi$ for all $k$ with $m \leq k < m'$.
- $\mathcal{I}, (r, m) \models D_G\phi$ if $\mathcal{I}, (r', m') \models \phi$ for all points $(r', m')$ with $(r, m) \sim_G (r'm')$;
- $\mathcal{I}, (r, m) \models C_G\phi$ if $\mathcal{I}, (r', m') \models \phi$ for all points $(r', m')$ with $(r, m) \sim_G^C (r'm')$, where $\sim_G^C = (\cup_{i\in G} \sim_i)^*$.

We write $\mathcal{I} \models \phi$ to mean that $\mathcal{I}, (r, 0) \models \phi$ for all runs of $\mathcal{I}$.

## Strategic Environments

In order to deal with agents that operate in an environment by strategically choosing their actions, we introduce a type of transition system that models the available actions and their effects on the state. An *environment* for agents $Ags$ is a tuple $E = \langle S, I, Acts, \to, \{O_i\}_{i\in Ags}, \pi\rangle$, where

1. $S$ is a set of states,
2. $I$ is a subset of $S$, representing the initial states,
3. $Acts = \Pi_{i\in Ags} Acts_i$ is a set of joint actions, where each $Acts_i$ is a nonempty set of actions that may be performed by agent $i$,
4. $\to \subseteq S \times Acts \times S$ is a transition relation, labelled by joint actions,
5. for each $i \in Ags$, component $O_i : S \to L_i$ is an observation function,
6. $\pi : S \to \mathcal{P}(Prop)$ is a propositional assignment.

Since later constructions add agent $e$, it is assumed here that $e \notin Ags$. An environment is said to be finite if all its components, i.e., $S, Ags, Acts_i, L_i$ and $Prop$ are finite. Intuitively, a joint action $a \in Acts$ represents a choice of action $a_i \in Acts_i$ for each agent $i \in Ags$, performed simultaneously, and the transition relation resolves this into an effect on the state. We assume that $\to$ is serial in the sense that for all $s \in S$ and $a \in Acts$ there exists $t \in S$ such that $(s, a, t) \in \to$.

A *strategy* for agent $i \in Ags$ in such an environment $E$ is a function $\alpha : S \to \mathcal{P}(Acts_i) \setminus \{\emptyset\}$, selecting a set of actions of the agent at each state.[1] We call these the actions *enabled* at the state. A strategy is *deterministic* if $\alpha(s)$ is a singleton for all $s$.

A *strategy* for a group $G \subseteq Ags$ is a tuple $\alpha_G = \langle\alpha_i\rangle_{i\in G}$ such that $\alpha_i$ is a strategy for agent $i$, for each $i \in G$. A group

[1]More generally, one can take a strategy to be a function of the history. We focus in this paper on strategies that depend only on the final state.

strategy is *deterministic* if $\alpha_i(s)$ is a singleton for all states $s$ and all $i \in G$. A strategy $\alpha_i$ for agent $i$ is *uniform* if for all states $s, t$, if $O_i(s) = O_i(t)$, then $\alpha_i(s) = \alpha_i(t)$. A strategy $\alpha_G = \langle \alpha_i \rangle_{i \in G}$ for a group $G$ is *locally uniform (deterministic)* if $\alpha_i$ is uniform (respectively, deterministic) for each agent $i \in G$.[2]

Given an environment $E$, we write $\Sigma(E)$ for the set of all joint strategies for the group $Ags$ of all agents, $\Sigma^{det}(E)$ for the set of deterministic strategies, $\Sigma^{unif}(E)$ for the set of all locally uniform joint strategies, and $\Sigma^{unif,det}(E)$ for the set of all deterministic locally uniform joint strategies.

We now introduce a notation for referring to an agent's strategy, that enables us to treat the strategy as itself an agent. Given an agent $i$, we write $\sigma(i)$ for a new agent that represents agent $i$'s strategy. For a set $G \subseteq Ags$ of agents, we define $\sigma(G) = \{\sigma(i) \mid i \in G\}$. We note that agent $\sigma(i)$ will typically not be associated with any actions, it is intended primarily for use in epistemic operators in the semantics of an interpreted system constructed to allow reference to what can be deduced were agents to reason using information about each other's strategies.

Given an environment $E = \langle S, I, Acts, \rightarrow, \{O_i\}_{i \in Ags}, \pi \rangle$ for agents $Ags$, where $O_i : S \rightarrow L_i$ for each $i \in Ags$, and a set $\Sigma \subseteq \Pi_{i \in Ags} \Sigma_i$ of joint strategies for the group $Ags$, we define the *strategy space* interpreted system $\mathcal{I}(E, \Sigma) = (\mathcal{R}, \pi')$ for agents $Ags \cup \sigma(Ags) \cup \{e\}$. The system $\mathcal{I}(E, \Sigma)$ has global states $\mathcal{G} = S \times \Pi_{i \in Ags} L_i \times \Pi_{i \in Ags} \Sigma_i$. Intuitively, each global state consists of a state of the environment $E$, a local state for each agent $i$ in $E$, and a strategy for each agent $i$. We index the components of this cartesian product by $e$, the elements of $Ags$ and the elements of $\sigma(Ags)$, respectively. We take the set of runs $\mathcal{R}$ of $\mathcal{I}(E, \Sigma)$ to be the set of all runs $r : \mathbb{N} \rightarrow \mathcal{G}$ satisfying the following constraints, for all $m \in \mathbb{N}$ and $i \in Ags$:

1. $r_e(0) \in I$ and $\langle r_{\sigma(i)}(0) \rangle_{i \in Ags} \in \Sigma$,
2. $r_i(m) = O_i(r_e(m))$,
3. $(r_e(m), a, r_e(m+1)) \in \rightarrow$ for some joint action $a \in Acts$ such that for all $j \in Ags$ we have $a_j \in \alpha_j(r_j(m))$, where $\alpha_j = r_{\sigma(j)}(m)$, and
4. $r_{\sigma(i)}(m+1) = r_{\sigma(i)}(m)$.

The first constraint, intuitively, says that runs start at an initial state of $E$, and the initial strategy profile at time 0 is one of the profiles in $\Sigma$. The second constraint states that the agent $i$'s local state at time $m$ is the observation that agent $i$ makes of the state of the environment at time $m$. The third constraint says that evolution of the state of the environment is determined at each moment of time by agents choosing an action by applying their strategy at that time to their local state at that time. The joint action resulting from these individual choices is then resolved into a transition on the state of the environment using the transition relation from $E$. The fourth constraint says that agents' strategies are fixed during the course of a run. Intuitively, each agent picks a strategy, and then sticks to it. The interpretation $\pi'$ of $\mathcal{I}(E, \Sigma)$ is determined from the interpretation $\pi$ of $E$ by taking $\pi'(r, m) = \pi(r_e(m))$ for all points $(r, m)$.

Our epistemic strategy logic is now just an instantiation of the extended temporal epistemic logic in the strategy space generated by an environment. That is, we start with an environment $E$ and an associated set of strategies $\Sigma$, and then work with the language CTL*K($Ags \cup \sigma(Ags) \cup \{e\}, Prop$) in the interpreted system $\mathcal{I}(E, \Sigma)$. We call this instance of the language CTL*K($Ags, Prop$), or just CTL*K when the parameters are implicit.

The key point is that the system $\mathcal{I}(E, \Sigma)$ represents the possible temporal evolutions under *all* possible choices of joint strategies from $\Sigma$ by the agents, and provides a way to refer, using distributed knowledge operators $D_G$ where $G$ contains the new strategic agents $\sigma(i)$, to what agents would know, should they take into account not just their own observations, but also information about other agent's strategies. For example, the distributed knowledge operator $D_{\{i,\sigma(i)\}}$ captures the knowledge that agent $i$ has, taking into account the strategy that it is running. $D_{\{i,\sigma(i),\sigma(j)\}}$ captures what agent $i$ would know, taking into account its own strategy and the strategy being used by agent $j$. Various applications of the usefulness of this expressiveness are given in the following sections.

In general, we will be most interested in the case where $\Sigma = \Sigma^{unif}(E)$ is the set of all locally uniform strategies in $E$, so we use the abbreviation $\mathcal{I}^{unif}(E)$ for $\mathcal{I}(E, \Sigma^{unif}(E))$, and similarly for the other strategy sets that we defined above.

We remark that the agent $e$ has perfect information about the state of the environment, including the agent's observation, so $D_{\{e\} \cup G}\phi$, for $G \subseteq Ags$ is equivalent to $D_{\{e\}}\phi$. Thus, the operator $D_{\{e\} \cup \sigma(G)}$ captures what can be deduced from knowledge of the state of the environment and the strategies of agents in $G$.

## Reasoning about Knowledge-Based Programs

Knowledge-based programs (Fagin et al. 1997) are a form of specification of a multi-agent system in the form of a program structure that specifies how an agent's actions are related to its knowledge. They have been shown to be a useful abstraction for several areas of application, including the development of optimal protocols for distributed systems (Fagin et al. 1997), robot motion planning (Brafman et al. 1997), and game theoretic reasoning (Halpern and Moses 2007).

Knowledge-based programs cannot be directly executed, since there is a circularity in their semantics: which actions are performed depends on what the agents know, which in turn depends on which actions the agents perform. The circularity is not vicious, and can be resolved by means of a fixed point semantics, but it means that knowledge-based programs may have multiple distinct implementations (or none), and the problem of reasoning about these implementations is quite subtle. In this section, we show that our framework can capture reasoning about the set of possible knowledge-based program implementations.

[2]We prefer the term "locally uniform" to just "uniform" in the case of groups, since we could say a strategy $\alpha$ for group $G$ is *globally uniform* if for all states $s, t$, if $O_i(s) = O_i(t)$ for all $i \in G$, then $\alpha_i(s) = \alpha_i(t)$ for all $i \in G$. While we do not pursue this in the present paper, this notion would be interesting in settings where agents share information to collude on their choice of move.

We consider joint knowledge-based programs $P$ (as defined by (Fagin et al. 1997)) where for each agent $i$ we have a knowledge-based program

$$P_i = \mathbf{do}\ \phi^i_1 \rightarrow a^i_1\ []\ \ldots []\ \phi^i_{n_i} \rightarrow a^i_{n_i}\ \mathbf{od}$$

where each $\phi^i_j$ is a formula of CTL*K($Ags, Prop$) of the form $K_i\psi$, and each $a_i$ appears just once.[3] Intuitively, this program says to repeat forever the following operation: nondeterministically execute one of the actions $a^i_j$ such that the corresponding guard $\phi^i_j$ is true.

We present a formulation of semantics for knowledge-based programs that refactors the definitions of (Fagin et al. 1997), following the approach of (van der Meyden 1996) which uses the notion of environment defined above rather than the original notion of *context*. A *potential implementation* of a knowledge-based program $P$ in an environment $E$ is a joint strategy $\alpha$ in $E$. Given a potential implementation $\alpha$ in $E$, we can construct the interpreted system $\mathcal{I}(E, \{\alpha\})$, which captures the possible runs of $E$ when the agents choose their actions according to the single possible joint strategy $\alpha$. Given this interpreted system, we can now interpret the epistemic guards in $P$. Say that a state $s$ of $E$ is $\alpha$-reachable if there is a point $(r, m)$ of $\mathcal{I}(E, \{\alpha\})$ with $r_e(m) = s$. We note that for a formula $K_i\phi$, and a point $(r, m)$ of $\mathcal{I}(E, \{\alpha\})$, the statement $\mathcal{I}(E, \{\alpha\}), (r, m) \models K_i\phi$ depends only on the state $r_e(m)$ of the environment at $(r, m)$. For an $\alpha$-reachable state $s$ of $E$, it therefore makes sense to define satisfaction of $K_i\phi$ at $s$ rather than at a point, by $\mathcal{I}(E, \{\alpha\}), s \models K_i\phi$ if $\mathcal{I}(E, \{\alpha\}), (r, m) \models K_i\phi$ for all $(r, m)$ with $r_e(m) = s$. We define $\alpha$ to be an *implementation* of $P$ in $E$ if for all $\alpha$-reachable states $s$ of $E$ and agents $i$, we have

$$\alpha_i(s) = \{a^i_j \mid 1 \le j \le n_i,\ \ \mathcal{I}(E, \{\alpha\}), s \models K_i\phi^i_j\}\ .$$

Intuitively, the right hand side of this equation is the set of actions that are enabled at $s$ by $P_i$ when the tests for knowledge are interpreted using the system obtained by running the strategy $\alpha$ itself. The condition states that the strategy is an implementation if it enables precisely this set of actions at every reachable state.

We now show that our framework for strategic reasoning can express the same content as a knowledge-based program by means of a formula, and that this enables the framework to be used for reasoning about knowledge-based program implementations.

We need one constraint on the environment. Say that an environment $E$ is *action-recording* if for each $a \in Acts_i$ there exists an atomic proposition $did_i(a)$ such that for $s \in I$ we have $did_i(a) \notin \pi(s)$ and for all states $s, t$ and joint actions $a$ such that $(s, a, t) \in \rightarrow$, we have $did_i(b) \in \pi(t)$ iff $b = a_i$, for all agents $i$. It is easily seen that any environment can be made action-recording, just by adding a component to the states that records the latest joint action.

We can now express knowledge-based program implementations as follows. For a formula $\psi$, write $\psi^\$$ for the result of substituting for each occurrence of an operator $D_G$ in $\psi$ the operator $D_{G\cup\sigma(Ags)}$. Intuitively, this substitution says that the operator $D_G$ in $\psi$ is to be interpreted as if it is known that the current joint strategy is being played. To understand the motivation for this, note that in $\mathcal{I}(E, \{\alpha\})$ it is common knowledge that the joint strategy $\alpha$ is being played. Let

$$\mathbf{imp}(P) = D_{\sigma(Ags)}\Big( \bigwedge_{i\in Ags, j=1\ldots n_i} ((\phi^i_j)^\$ \Leftrightarrow EXdid_i(a^i_j))\Big).$$

Intuitively, this formula says that the current joint strategy gives an implementation of the knowledge-based program $P$. More precisely, we have the following:

**Proposition 1** *Suppose that $P$ is a knowledge-based program that does not make use of common knowledge operators. Let $\alpha$ be a locally uniform joint strategy in $E$ and let $r$ be a run of $\mathcal{I}^{unif}(E)$, in which the agents are running joint strategy $\alpha$, i.e., where $r_{\sigma(i)}(0) = \alpha_i$ for all $i \in Ags$. Let $m \in \mathbb{N}$. Then the strategy $\alpha$ is an implementation of knowledge-based program $P$ in $E$ iff $\mathcal{I}^{unif}(E), (r, m) \models$ **imp**$(P)$.*

Note that here we check **imp**$(P)$ in the system $\mathcal{I}^{unif}(E)$ that contains *all* joint strategies, not just a single strategy $\alpha$, as in the definition of implementation for a knowledge-based program. The point of this is that *a priori*, we do not have an implementation at hand.

In particular, as a consequence of this result, it follows that several properties of knowledge-based programs (that do not make use of common knowledge operators) can be expressed in the system $\mathcal{I}^{unif}(E)$:

1. The statement that there exists an implementation of $P$ in $E$ can be expressed by
$$\mathcal{I}^{unif}(E) \models \neg D_\emptyset \neg \mathbf{imp}(P)$$
2. The statement that all implementations of $P$ in $E$ guarantee that formula $\phi$ holds at all times can be expressed by
$$\mathcal{I}^{unif}(E) \models D_\emptyset(\mathbf{imp}(P) \Rightarrow D_{\sigma(Ags)}\phi^\$))$$

We remark that as a consequence of these encodings and the fact that the model checking problem described below is in PSPACE, we obtain that testing for the existence of an implementation of a knowledge-based program is in PSPACE, as is the problem of determining whether all implementations satisfy some formula. For testing existence, this result was known (Fagin et al. 1997), but the result on verification has not previously been noted (though it could also have been shown using the techniques in (Fagin et al. 1997).)

We remark that the reason for excluding common knowledge $C_G\phi$ from the conditions in the knowledge-based program in Proposition 1 is that this would require us to express a greatest fixedpoint $X$ of the equation $X \equiv \bigwedge_{i\in G} D_{\{i\}\cup\sigma(Ags)}(X \wedge \phi)$, which is not the same as $C_{G\cup\sigma(Ags)}\phi$. One appropriate way to handle this in our framework would be to work with a mu-calculus extension. We do not pursue this here.

[3]The guards in (Fagin et al. 1997) are allowed to be boolean combinations of formulas $K_i\psi$ and propositions $p$ local to the agent: since for such propositions $p \Leftrightarrow K_i p$, and the operator $K_i$ satisfies positive and negative introspection, our form for the guards is equally general. They do not require that $a_i$ appears just once, but the program can always be put into this form by aggregating clauses for $a_i$ into one and taking the disjunction of the guards.

## Connections to variants of ATEL

Alternating temporal logic (ATL) (Alur, Henzinger, and Kupferman 2002) is a generalization of the branching time temporal logic CTL that can express the capability of agent strategies to bring about temporal effects. Essentially, each branching construct $A\phi$ is generalized to an *alternating* construct $\langle\langle G\rangle\rangle\phi$ for a group $G$ of agents, where $\phi$ is a "prefix temporal" formula such as $X\phi'$, $F\phi'$, $G\phi'$ or $\phi_1 U\phi_2$, as would be used to construct a CTL operator. Intuitively, $\langle\langle G\rangle\rangle\phi$ says that the group $G$ has a strategy for ensuring that $\phi$ holds, irrespective of what the other agents do.

Alternating temporal epistemic logic (ATEL), adds epistemic operators to ATL (van der Hoek and Wooldridge 2002). As a number of subtleties arise in the formulation of such logics, several variants of ATEL have since been developed. In this section, we consider a number of such variants and argue that our framework is able to express the main strategic concepts from these variants. We begin by recalling ATEL as defined in (van der Hoek and Wooldridge 2002). Various modellings of the environments in which agents operate have been used in the literature; we base our modelling on the notion of environment introduced above.

The syntax of ATEL is given as follows:

$$\phi \equiv p \mid \neg\phi \mid \phi_1\vee\phi_2 \mid \langle\langle G\rangle\rangle\bigcirc\phi \mid \langle\langle G\rangle\rangle\Box\phi \mid \langle\langle G\rangle\rangle(\phi_1 U\phi_2) \mid K_i\phi \mid C_G\phi$$

where $p \in Prop$, $i \in Ags$ and $G \subseteq Ags$. The intuitive meaning of the constructs is as in CTL$^*$K above, with additionally $\langle\langle G\rangle\rangle\phi$ having the intuitive reading that group $G$ has a strategy for assuring that $\phi$ holds. The semantics is given by a relation $E, s \models^\Sigma \phi$, where $E = \langle S, I, Acts, \rightarrow, \{O_i\}_{i\in Ags}, \pi\rangle$ is an environment, $s \in S$ is a state of $E$, and $\phi$ is a formula. For reasons discussed below, we parameterize the definition on a set $\Sigma$ of strategies for groups of agents in the environment $E$. For the definition, we need the notion of a path in $E$: this is a function $\rho : \mathbb{N} \rightarrow S$ such that for all $k \in \mathbb{N}$ there exists a joint action $a$ with $(\rho(k), a, \rho(k+1)) \in\rightarrow$. A path $\rho$ is *from* a state $s$ if $\rho(0) = s$. A path $\rho$ is *consistent* with a strategy $\alpha$ for a group $G$ if for all $k \in \mathbb{N}$ there exists a joint action $a$ such that $(\rho(k), a, \rho(k+1)) \in\rightarrow$ and $a_i \in \alpha_i(\rho(k))$ for all $i \in G$.

The relation $E, s \models^\Sigma \phi$ is defined inductively on the structure of the formula $\phi$:

- $E, s \models^\Sigma p$ if $p \in \pi(s)$;
- $E, s \models^\Sigma \neg\phi$ if not $E, s \models^\Sigma \phi$;
- $E, s \models^\Sigma \phi \wedge \psi$ if $E, s \models^\Sigma \phi$ and $E, s \models^\Sigma \psi$;
- $E, s \models^\Sigma \langle\langle G\rangle\rangle\bigcirc\phi$ if there exists a strategy $\alpha_G \in \Sigma$ for group $G$ such that for all paths $\rho$ from $s$ that are consistent with $\alpha_G$, we have $E, \rho(1) \models^\Sigma \phi$;
- $E, s \models^\Sigma \langle\langle G\rangle\rangle\Box\phi$ if there exists a strategy $\alpha_G \in \Sigma$ for group $G$ such that for all paths $\rho$ from $s$ that are consistent with $\alpha_G$, we have $E, \rho(k) \models^\Sigma \phi$ for all $k \in \mathbb{N}$;
- $E, s \models^\Sigma \langle\langle G\rangle\rangle(\phi U\psi)$ if there exists a strategy $\alpha_G \in \Sigma$ for group $G$ such that for all paths $\rho$ from $s$ that are consistent with $\alpha_G$, there exists $m \geq 0$ such that $E, \rho(m) \models^\Sigma \psi$, and for all $k < m$, we have $E, \rho(k) \models^\Sigma \phi$.
- $E, s \models^\Sigma K_i\phi$ if $E, t \models^\Sigma \phi$ for for all $t \in S$ with $t \sim_i s$;
- $E, s \models^\Sigma C_G\phi$ if $E, t \models^\Sigma \phi$ for for all $t \in S$ with $(s, t) \in (\cup_{i\in G} \sim_i)^*$;

The specific version of ATEL defined in (van der Hoek and Wooldridge 2002) is obtained from the above definitions by taking $\Sigma = \Sigma^{det} = \{\sigma_G \mid G \subseteq Ags,\ \sigma_G$ a deterministic $G$-strategy in $E\}$. That is, following the definitions for ATL, this version works with arbitrary deterministic group strategies, in which an agent selects its action as if it has full information of the state. This aspect of the definition has been critiqued by Jonker (Jonker 2003) and (in the case of ATL without epistemic operators) by Schobbens (Schobbens 2004), who argue that this choice is not in the spirit of the epistemic extension, in which observations are intended precisely to represent that agents do not have full information of the state. They propose that the definition instead be based on the set $\Sigma^{\mathrm{det},unif} = \{\sigma_G \mid G \subseteq Ags,\ \sigma_G$ a locally uniform deterministic $G$-strategy in $E\}$. This ensures that in choosing an action, agents are able to use only the information available in their observations.

We concur that the use of locally uniform strategies is the more appropriate choice, but in either event, we now argue that our approach using strategy space is able to express everything that can be expressed in ATEL. Consider the following translation from ATEL to CTL$^*$K$(Prop, Ags^+ \cup \sigma(Ags) \cup \{e\})$. For a formula $\phi$, we write $\phi^*$ for the translation of $\phi$, defined inductively on the construction of $\phi$ by the following rules

$$p^* = p \qquad (\neg\phi)^* = \neg\phi^* \qquad (\phi_1 \wedge \phi_2)^* = \phi_1^* \wedge \phi_2^*$$
$$(K_i\phi)^* = K_i\phi^* \qquad (C_G\phi)^* = C_G\phi^*$$
$$(\langle\langle G\rangle\rangle\bigcirc\phi)^* = \neg K_e\neg D_{\{e\}\cup\sigma(G)}\bigcirc\phi^*$$
$$(\langle\langle G\rangle\rangle\Box\phi)^* = \neg K_e\neg D_{\{e\}\cup\sigma(G)}\Box\phi^*$$
$$(\langle\langle G\rangle\rangle\phi_1 U\phi_2)^* = \neg K_e\neg D_{\{e\}\cup\sigma(G)}(\phi_1^* U\phi_2^*)$$

Given a strategy $\alpha = \langle\alpha_i\rangle_{i\in G}$ for a group of agents $G$ in an environment $E$, define the *completion* of the strategy to be the joint strategy $comp(\alpha) = \langle\alpha'_i\rangle_{i\in G}$ with $\alpha'_i = \alpha_i$ for $i \in G$ and with $\alpha_i(s) = Acts_i$ for all $i \in Ags \setminus G$ and $s \in S$. Intuitively, this operation completes the group strategy to a joint strategy for all agents, by adding the random strategy for all agents not in $G$. Given a set of strategies $\Sigma$ for groups of agents, we define $comp(\Sigma) = \{comp(\alpha) \mid \alpha \in \Sigma\}$. Say that a set $\Sigma$ of group strategies is *restrictable* if for every $\alpha \in \Sigma$ for group of agents $G$ and every group $G' \subseteq G$, the restriction $\alpha_{G'}$ of $\alpha$ to agents in $G'$ is also in $\Sigma$. Say that $\Sigma$ is *extendable* if for every strategy for a group $G' \subseteq G$, there exists a strategy $\alpha' \in \Sigma$ for group $G$ whose restriction $\alpha'_{G'}$ to $G'$ is equal to $\alpha$. For example, the set of all group strategies, and the set of all locally uniform group strategies, are both restrictable and extendable.

For an environment $E$, write $E[S/I]$ for the environment obtained by making all states in $E$ be initial, i.e., replacing the set of initial states $I$ of $E$ by the set of all states $S$ of $E$. (This is a technical transformation that we require because $E$ may have temporally unreachable states that would not occur in an interpreted system constructed from $E$, but that can be accessed via an equivalence relation $\sim_i$ in the semantics of ATEL.) We can then show the following.

**Theorem 1** *For every environment $E$, nonempty set of group strategies $\Sigma$ that is restrictable and extendable, for every state $s$ of $E$ and ATEL formula $\phi$, we have $E, s \models^{\Sigma} \phi$ iff for all (equivalently, some) points $(r, m)$ of $\mathcal{I}(E[S/I], comp(\Sigma))$ with $r_e(m) = s$ we have $\mathcal{I}(E[S/I], comp(\Sigma)), (r, m) \models \phi^*$.*

Similar translation results can be given for other alternating temporal epistemic logics from the literature. We sketch a few of these translations here.

Jamroga and van der Hoek (Jamroga and van der Hoek 2004) note that ATEL admits situations consisting of an environment $E$ and a state $s$ where $E, s \models K_i\langle\langle i\rangle\rangle\phi$, i.e., in every state consistent with agent $i$'s knowledge, some strategy for agent $i$ is guaranteed to satisfy $\phi$, but still there is no strategy for agent $i$ that agent $i$ knows will work to achieve $\phi$. They formulate a construct $\langle\langle G\rangle\rangle^{\bullet}_{\mathcal{K}(H)}\phi$ that says, effectively, that there is a strategy for a group $G$ that another group $H$ knows (for notion of group knowledge $\mathcal{K}$, which could be $E$ for everyone knows, $D$ for distributed knowledge, or $C$ for common knowledge) to achieve goal $\phi$. More precisely,

> $E, s \models \langle\langle G\rangle\rangle^{\bullet}_{\mathcal{K}(H)}\phi$ if there exists a uniform strategy $\alpha$ for group $G$ such that for all states $t$ with $s \sim^{\mathcal{K}}_{H} t$, we have that all paths $\rho$ from $t$ that are consistent with $\alpha$ satisfy $\phi$.

Here $\sim^{\mathcal{K}}_{H}$ is the appropriate epistemic indistinguishability relation on states of $E$. The particular case $\langle\langle G\rangle\rangle^{\bullet}_{E(G)}\phi$ is also proposed as the semantics for the ATL construct $\langle\langle G\rangle\rangle\phi$ in (Schobbens 2004; Jonker 2003; Jamroga and Ågotnes 2007).

The construct $\langle\langle G\rangle\rangle^{\bullet}_{D(H)}\phi$ can be represented in our language as

$$\neg D_e \neg D_{H\cup\sigma(G)}\phi\ .$$

Intuitively, here the first modal operator $\neg D_e\neg$ switches the strategy of all the agents while maintaining the state $s$, thereby selecting a strategy $\alpha$ for group $G$ in particular, and the next operator $D_{\{H,\sigma(G)\}}$ verifies that the group $H$ knows that the strategy $G$ being used by group $G$ guarantees $\phi$. Similarly, $\langle\langle G\rangle\rangle^{\bullet}_{E(H)}\phi$ can be represented as

$$\neg D_e \neg \bigwedge_{i\in H} D_{\{i\}\cup\sigma(G)}\phi\ .$$

This gives a reduction of these complex operators of (Jamroga and van der Hoek 2004) to a set of standard epistemic operators.[4] (An alternate approach to decomposing the operators $\langle\langle G\rangle\rangle^{\bullet}_{\mathcal{K}(H)}$ is proposed in (Jamroga and Ågotnes 2007). By comparison with our standard semantics, this proposal uses "constructive knowledge" operators which require a nonstandard semantics in which formulas are evaluated at sets of states rather than at individual states.)

(W. van der Hoek, Jamroga, and Wooldridge 2005) introduce constants that refer to strategies, and adds to ATL a new (counterfactual) modality $C_i(c, \phi)$, with the intended reading "if it were the case that agent $i$ committed to the strategy denoted by $c$, then $\phi$". The formula $\phi$ here is not permitted to contain further references to agent $i$ strategies.

[4]As with knowledge-based programs above, to handle common knowledge, it seems we require an extension to mu-calculus.

To interpret the formula $C_i(c, \phi)$ in an environment $E$, the environment is first updated to a new environment $E'$ by removing all transitions that are inconsistent with agent $i$ running the strategy referred to by $c$, and then the formula $\phi$ is evaluated in $E'$. After introducing propositional constants $p_{i,c}$ that say that agent $i$ is running the strategy referred to by $c$, the formula $C_i(c, \phi)$ could be expressed in our framework as $D_{\{e\}\cup\sigma(Ags\setminus\{i\})}(p_{i,c} \Rightarrow \phi^{+\sigma(i)})$ where in the translation $\phi^{+\sigma(i)}$ of $\phi$ we ensure that there is no further deviation from the strategy of agent $i$ by adding $\sigma(i)$ to the group of every knowledge operator occurring later in the translation.

(W. van der Hoek, Jamroga, and Wooldridge 2005) do not include epistemic operators. Were they to be added in the obvious way, the effect would be to make the meaning of the knowledge operator (with respect to the agent's knowledge concerning the strategies in play) dependent on the context in the formula. In particular, in $(K_j\phi) \wedge C_i(c, K_j\psi)$, the first occurrence of $K_j$ would refer to an agent $j$ who does not know that $i$ is playing $c$, whereas the second $K_j$ would effectively refer to our $K_{\{j,\sigma(i)\}}$, i.e., $j$'s knowledge taking into account that $i$ is playing $c$. Our framework is more expressive in that we can continue to use the former meaning even in the second context.

## Solution Concepts

It has been shown for a number of logics for strategic reasoning that they are expressive enough to state a variety of game theoretic solution concepts, e.g., (W. van der Hoek, Jamroga, and Wooldridge 2005; Chatterjee, Henzinger, and Piterman 2010) show that Nash Equilibrium is expressible. We now sketch the main ideas required to show that our framework also has this expressive power. We assume two players $Ags = \{0, 1\}$ in a normal form game, and assume that these agents play a deterministic strategy. The results in this section can be easily generalized to multiple players and extensive form games.

Given a game $\mathcal{G}$ we construct an environment $E_{\mathcal{G}}$ that represents the game. Each player has a set of actions that correspond to the moves that the player can make. We assume that $E_{\mathcal{G}}$ is constructed to model the game so that play happens in the first step from a unique initial state, and that subsequent transitions do not change the state.

Let $u_i$ for $i \in \{0, 1\}$ be a variable denoting the utility gained by player $i$ when play is finished. Let $V_i$ be the set of possible values for $u_i$. We write $-i$ to denote the adversary of player $i$. We use formula

$$U_i(v) = \bigcirc(u_i = v)$$

to express that value $v$ is player $i$'s utility once play finishes.

**Nash equilibrium (NE)** is a solution concept that states no player can gain by unilaterally changing their strategy. We may write

$$BR_i(v) = U_i(v) \wedge K_{\sigma(-i)} \bigwedge_{v'\in V_i} (U_i(v') \Rightarrow v' \leq v)$$

to express that, given the current adversary strategy, the value $v$ attained by player $i$'s current strategy is the best possible utility attainable by player $i$, i.e., the present strategy of player $i$ is a best response to the adversary. Thus

$$BR_i = \bigvee_{v \in V_i} BR_i(v)$$

says that player $i$ is playing a best-response to the adversary's strategy. The following statement then expresses the existence of a Nash equilibrium for the game $\mathcal{G}$:

$$\mathcal{I}^{unif,det}(E_{\mathcal{G}}) \models \neg D_{\emptyset} \neg (BR_0 \wedge BR_1) \ .$$

That is, in a Nash equilibrium, each player is playing a best response to the other's strategy.

**Perfect cooperative equilibrium (PCE)** is a solution concept intended to overcome deficiencies of Nash equilibrium for explaining cooperative behaviour (Halpern and Rong 2010). It says that each player does at least as well as she would if the other player were best-responding. The following formula

$$BU_i(v) = D_{\emptyset}( \bigwedge_{v' \in V_i} ((BR_{-i} \wedge U_i(v')) \Rightarrow v' \leq v))$$

states that $v$ is as good as any utility that $i$ can obtain if the adversary always best-responds to whatever $i$ plays. Thus,

$$BU_i = \bigvee_{v \in V_i} (U_i(v) \wedge BU_i(v))$$

says that $i$ is currently getting a utility as good the best utility that $i$ can obtain if the adversary is a best-responder. Now, the following formula expresses the existence of perfect cooperative equilibrium for the game $\mathcal{G}$:

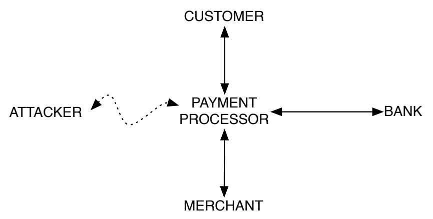
$$\mathcal{I}^{unif,det}(E_{\mathcal{G}}) \models \neg D_{\emptyset} \neg (BU_0 \wedge BU_1)$$

That is, in a PCE, no player has an incentive to change their strategy, on the assumption that the adversary will best-respond to any change.

## Computer Security Example: Erasure policies

Formal definitions of computer security frequently involve reference to the strategies available to the players, and to agent's reasoning based on these strategies. In this section we sketch an example that illustrates how our framework might be applied in this context.

Consider the scenario depicted in the following diagram:



A customer $C$ can purchase items at a web merchant $M$. Payment is handled by a trusted payment processor $P$ (this could be a service or device), which interacts with the customer, merchant, and a bank $B$ to securely process the payment. (To keep the example simple, we suppose that the customer and merchant use the same bank). One of the guarantees provided by the payment processor is to protect the customer from attacks on the customer's credit card by the merchant: the specification for the protocol that runs the transaction requires that the merchant should not obtain the customer's credit card number. In fact, the specification for the payment processor is that after the transaction has been successfully completed, the payment processor should *erase* the credit card data, to ensure that even the payment processor's state does not contain information about the customer's credit card number. The purpose of this constraint is to protect the customer against subsequent attacks by an attacker $A$, who may be able to use vulnerabilities in the payment processor's software to obtain access to the payment processor's state.

We sketch how one might use our framework to express the specification. To capture reasoning about all possible behaviours of the agents, and what they can deduce from knowledge of those behaviours, we work in $\mathcal{I}^{unif}(E)$ for a suitably defined environment $E$. To simplify matters, we take $Ags = \{C, M, P, A\}$. We exclude the strategy of the bank from consideration: this amounts to assuming that the bank has no actions and is trusted to run a fixed protocol. We similarly assume that the payment processor $P$ has no actions, but in order to talk about what information is encoded in the payment processor's local state, we do allow that this agent has observations. The customer $C$ may have actions such as entering the credit card number in a web form, pressing a button to submit the form to the payment processor, and pressing a button to approve or cancel the transaction. The customer observes variable `cc`, which records the credit card number drawn from a set `CCN`, and boolean variable `done` which records whether the transaction is complete (which could mean either committed or aborted).

We assume that the attacker $A$ has some set of exploit actions, as well as some innocuous actions (e.g., setting a local variable or performing *skip*). The effect of the exploit actions is to exploit a vulnerability in the payment processor's software and copy parts of the local state of the payment processor to variables that are observable by the attacker. We include in the environment state a boolean variable `exploited`, which records whether the attacker has executed an exploit action at some time in the past. The merchant $M$ may have actions such as sending cost information to the payment processor and acknowledging a receipt certifying that payment has been approved by the bank (we suppose this receipt is transmitted from the bank to the merchant via the payment processor).

We may then capture the statement that the system is *potentially* vulnerable to an attack that exploits an erasure flaw in the implementation of the payment processor, by the following formula:

$$\neg D_{\emptyset} \neg (\texttt{done} \wedge \bigvee_{x \in \texttt{CCN}} K_P(\texttt{cc} \neq x)))$$

This says that there exist behaviours of the agents, which can (at least on some points in some runs) leave the payment processor in a state where the customer has received confirmation that the transaction is done, but in which the payment processor's local state somehow still encodes *some* information about the customer's credit card number. This encoding could be direct (e.g., by having a variable *customer_cc* that still stores the credit card number) or indirect (e.g. by the local state including both a symmetric encryption key $K$ and an encrypted version of the credit card number, *enc_customer_cc*, with value $\texttt{Encrypt}_K(\texttt{cc})$ that was used for secure transmission to the bank). Note that for a breach of security, it is only required that the information suffices to *rule out* some credit card number (so that, e.g., knowing the first digit of the number would constitute a vulnerability)

The vulnerability captured by this formula is only potential, because it does not necessarily follow that the attacker is able to obtain the credit card information. Whether this is possible can be checked using the formula

$$\neg D_\emptyset \neg(\texttt{done} \land \neg\texttt{exploited} \land EF \bigvee_{x \in \texttt{CCN}} D_{\{A,\sigma(A)\}}(\texttt{cc} \neq x))$$

which says that it is possible for the attacker to obtain information about the credit card number even after the transaction is done. (To focus on erasure flaws, we deliberately wish to exclude here the possibility that the attack occurs during the processing of the transaction.) Note that here we assume that the attacker knows their own strategy when making deductions from the information obtained in the attack. This is necessary, because the attacker can typically write its own local variables, so it needs to be able to distinguish between a value it wrote itself and a value it copied from the payment processor.

However, even this formula may not be sufficiently strong. Suppose that the payment processor implements erasure by writing a random value to its variable *customer_cc*. Then, even if the attacker obtains a copy of this value, and it happens to be equal to the customer's actual credit card number, the attacker would not have any knowledge about the credit card number, since, as far as the attacker knows, it could be looking at a randomly assigned number. However, there may still be vulnerabilities in the system. Suppose that the implementation of the payment processor operates so that the customer's credit card data is not erased by randomization until the merchant has acknowledged the receipt of payment from the bank, but to avoid annoying the customer with a hanging transaction, the customer is advised that the transaction is approved (setting done true) if the merchant does not respond within a certain time limit. It is still the case that on observing the copied value of *customer_cc*, the attacker would not be able to deduce that this is the customer's credit card number, since it might be the result of erasure in the case that the merchant responded promptly. However, if the attacker knows that the merchant has not acknowledged the receipt, the attacker can then deduce that the value is not due to erasure. One way in which the attacker might know that the merchant has not acknowledged receipt is that the attacker is in collusion with the merchant, who has agreed to omit sending the required acknowledgement messages.

This type of attack can be captured by replacing the term $D_{\{A,\sigma(A)\}}(\texttt{cc} \neq x)$ by $D_{\{A,\sigma(A),\sigma(M)\}}(\texttt{cc} \neq x)$, capturing that the attacker reasons using knowledge of both its own strategy as well as the strategy of the merchant, or even $D_{\{A,\sigma(A),\sigma(M),M\}}(\texttt{cc} \neq x)$ for a collusion in which the merchant shares information observed. Similarly, to focus on erasure flaws in the implementation of the payment gateway, independently of the attackers capability, we would replace the term $K_P(\texttt{cc} \neq x)$ above by $D_{\{P,\sigma(M)\}}(\texttt{cc} \neq x)$.

We remark that in the case of the attacker's knowledge, it would be appropriate to work with a perfect recall semantics of knowledge, but when using knowledge operators to express information in the payment gateway's state for purposes of reasoning about erasure policy, the more appropriate semantics of knowledge is imperfect recall.

This example illustrates some of the subtleties that arise in the setting of reasoning about security and the way that our framework helps to represent them. Erasure policies have previously been studied in the computer security literature, beginning with (Chong and Myers 2005), though generally without consideration of strategic behaviour by the adversary. However, many other notions in the security literature do involve reasoning about strategies and agent's knowledge based on strategies, including nondeducibility on strategies (Wittbold and Johnson 1990) and robust declassification (Zdancewic and Myers 2001). We leave the further exploration of such notions using our approach for future work.

## Model Checking

We now consider the problem of model checking in strategy space. Recall that, given an interpreted system $\mathcal{I}$, we write $\mathcal{I} \models \phi$ when $\mathcal{I}, (r, 0) \models \phi$ for all runs $r$ of $\mathcal{I}$. Since interpreted systems are infinite, we need a finite representation for a decision problem, for this we use a finite state environment $E$ together with an associated set of strategies $\Sigma$, and consider the problem of determining whether $\mathcal{I}(\mathcal{E}(E, \Sigma)) \models \phi$. Typically, $\Sigma$ would be a set such as the set $\Sigma^{unif}(E)$ of all uniform strategies for $E$.

For generality, we abstract $\Sigma$ further, to a paramaterized class such that for each environment $E$, the set $\Sigma(E)$ is a set strategies for $E$. We say that the parameterized class $\Sigma(E)$ is *PTIME-presented*, if it is presented by means of an algorithm that runs in time polynomial in the size of $E$ and verifies if a given strategy $\alpha$ is in $\Sigma(E)$. (We assume that strategies $\alpha$ are given as a list of tuples $(s, a) \in S \times Acts$ such that $a \in \alpha(s)$.) For example, the class $\Sigma(E)$ of all strategies for $E$ can be PTIME-presented, as can $\Sigma^{unif}(E)$, $\Sigma^{det}(E)$ and $\Sigma^{unif,det}(E)$.

A naive model checking algorithm would construct an extended transition system over the state space $S \times \Sigma(E)$ and then apply standard model checking techniques for temporal epistemic logic. Note that a joint strategy for an environment $E$ can be represented in space $|S| \times |Acts|$, where $S$ is the set of states of $E$ and $Acts$ the set of joint actions. Thus, such an extended transition system generally has exponentially many states as a function of the size of $E$. This means that the naive procedure would require not less than exponential time. In fact it, is possible to do better than this (assuming that PSPACE is better than EXPTIME). We show the following:

**Theorem 2** *Let $\Sigma(E)$ be a PTIME presented class of strate-*

*gies for environments E. The complexity of deciding, given an environment E, and a CTL*K formula $\phi$ for agents $\{e\} \cup Ags(E) \cup \sigma(Ags(E))$, whether $\mathcal{I}(E, \Sigma(E)) \models \phi$, is PSPACE-complete.*

It is interesting to note that, although we have significantly more expressive power than the temporal logic LTL on which we build, we achieve this without an increase in complexity: model checking LTL is already PSPACE-complete (Sistla and Clarke 1985). We note that this result also strongly suggests that we do have a strictly stronger expressive power than ATEL, since the complexity of model checking ATEL, assuming uniform strategies, is $P^{NP}$-complete. The class $P^{NP}$ consists of problems solvable by PTIME computations with access to an NP oracle). For ATEL, model checking can be done with a polynomial time (with respect to the size of formula) computation with access to an oracle that is in NP with respect to both the number of states and the number of joint actions. In particular, (Schobbens 2004) proves this upper bound and (Jamroga and Dix 2006) proves a matching lower bound.

## Conclusions

We refer the reader to the sections above for references and comparison to related work on each of the topics that we cover. Beside these references, the following are also worth mentioning.

Strategy Logic (Chatterjee, Henzinger, and Piterman 2010) is a (non-epistemic) generalization of ATL for perfect information strategies in which strategies may be explicitly named and quantified. Our logic has implicit quantification over strategies using the strategic agents in the epistemic operators. Our logic could be further generalized to make the reference to strategies explicit, but we note that strategy logic has a non-elementary model checking problem, so this could come at the cost of a large increase in complexity. Work on identification of more efficient variants of quantified strategy logic includes (Mogavero, Murano, and Vardi 2010), who formulate a variant with a 2-EXPTIME-complete model checking problem.

(van Eijck 2013) introduces a variant of propositional dynamic logic (PDL) for describing strategy profiles in normal form games subject to preference relations, but does not cover temporal aspects as we have in this paper. Another approach based on PDL is given in (Ramanujam and Simon 2008), which describes strategies by means of formulas.

(Schnoor 2010) defines a very rich generalization of ATEL for probabilistic environments, that includes variables that refer to *strategy choices*, and in which the strategic operators refer to these variables, so that statements of the form "when coalition A runs the strategy determined by variable S1, and coalition B runs the strategy determined by S2, then probability that $\phi$ holds is at least $\delta$" can be expressed. Here a strategy choice uniformly maps each state, coalition and formula to a uniform strategy for the coalition. The strategic variables may be quantified, but only in the prefix of the formula. Because of the complexity of this framework, we leave a more detailed comparison for later work, but we note that the epistemic operators in this approach are interpreted with respect to a fixed strategy choice, compared to our approach in which the epistemic operators work over the entire strategy space.

Our focus on this paper has been on an observational, or imperfect recall, semantics for knowledge. Other semantics for knowledge are also worth consideration, but left for future work. We note one issue in relation to the connection to ATEL that we have established, should we consider a perfect recall version of our logic. ATEL operators effectively allow reference to situations in which agents switch their strategy after some actions have already been taken, whereas in our model an agent's strategy is fixed for the entire run. When switching to a new strategy, there is the possibility that the given state is not reachable under this new strategy. We have handled this issue in our translation by assuming that all states are initial, so that the run can be reinitialized if necessary to make the desired state reachable. This is consistent with an imperfect recall interpretation of ATEL, but it is not clear that this approach is available on a perfect recall interpretation. We leave a resolution of this issue to future work.

In several places in the present paper (Proposition 1 and Theorem 1), we have made a technical restriction to formulas not involving the common knowledge operator in order to establish a result about the expressive power of our logic, and remarked that an extension of the logic appears to be required for these results to encompass common knowledge operators. In a subsequent work (Huang and van der Meyden 2014), we have developed an extension that has the required expressiveness, by introducing a capacity for quantifying over strategies.

## References

Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-Time Temporal Logic. *Journal of the ACM* 49(5):672–713.

Brafman, R. I.; Latombe, J.-C.; Moses, Y.; and Shoham, Y. 1997. Applications of a Logic of Knowledge to Motion Planning under Uncertainty. *JACM* 44(5).

Chatterjee, K.; Henzinger, T. A.; and Piterman, N. 2010. Strategy logic. *Inf. Comput.* 208(6):677–693.

Chong, S., and Myers, A. C. 2005. Language-based information erasure. In *IEEE Computer Security Foundations Workshop*, 241–254.

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

Fagin, R.; Halpern, J. Y.; Moses, Y.; and Vardi, M. Y. 1997. Knowledge-based programs. *Distributed Computing* 10(4):199–225.

Halpern, J. Y., and Moses, Y. 1990. Knowledge and Common Knowledge in a Distributed Environment. *JACM* 37(3):549–587.

Halpern, J. Y., and Moses, Y. 2007. Characterizing Solution Concepts in Games Using Knowledge-Based Programs. In *the 20nd International Joint Conference on Artificial Intelligence (IJCAI2007)*, 1300–1307.

Halpern, J. Y., and O'Neill, K. R. 2008. Secrecy in Multiagent Systems. *ACM Transactions on Information and System Security* 12(1).

Halpern, J. Y., and Rong, N. 2010. Cooperative Equilibrium (Extended Abstract). In *9th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'10)*.

Horty., J. F. 2001. *Agency and deontic logic*. Oxford University Press.

Huang, X., and van der Meyden, R. 2014. An epistemic strategy logic. In *2nd Int. Workshop on Strategic Reasoning*, EPTCS. to appear.

Jamroga, W., and Ågotnes, T. 2007. Constructive knowledge: what agents can achieve under imperfect information. *Journal of Applied Non-Classical Logics* 17(4):423–475.

Jamroga, W., and Dix, J. 2006. Model checking abilities under incomplete information is indeed delta2-complete. In *the 4th European Workshop on Multi-Agent Systems (EUMAS'06)*.

Jamroga, W., and van der Hoek, W. 2004. Agents that Know How to Play . *Fundamenta Informaticae* 62:1–35.

Jamroga, W. 2003. Some Remarks on Alternating Temporal Epistemic Logic. In *Proceedings of Formal Approaches to Multi-Agent Systems (FAMAS 2003)*.

Jonker, G. 2003. Feasible strategies in alternating-time temporal. Master's thesis, University of Utrech, The Netherlands.

Mogavero, F.; Murano, A.; and Vardi, M. Y. 2010. Reasoning about strategies. In *FSTTCS*, 133–144.

Parikh, R., and Ramanujam, R. 1985. Distributed Processes and the Logic of Knowledge. In *Logics of Programs 1985*, 256–268.

Parikh, R. 1983. Propositional game logic. In *IEEE Symp. on Foundations of Computer Science*, 195–200.

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

Pnueli, A. 1977. The Temporal Logic of Programs. In *Symp. on Foundations of Computer Science*, 46–57.

Ramanujam, R., and Simon, S. E. 2008. Dynamic logic on games with structured strategies. In *KR*, 49–58.

Schnoor, H. 2010. Explicit strategies and quantification for atl with incomplete information and probabilistic games. Technical Report 1008, Institut für Informatik, Christian-Albrechts Universität zu Kiel.

Schobbens, P.-Y. 2004. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science* 85(2):82–93.

Sistla, A. P., and Clarke, E. M. 1985. The complexity of propositional linear temporal logics. *J. ACM* 32(3):733–749.

van der Hoek, W., and Wooldridge, M. 2002. Tractable multiagent planning for epistemic goals. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, 1167–1174.

van der Meyden, R. 1996. Knowledge Based Programs: On the Complexity of Perfect Recall in Finite Environments. In *6th Conference on Theoretical Aspects of Rationality and Knowledge (TARK 1996)*, 31–49.

van Eijck, J. 2013. PDL as a multi-agent strategy logic. In *Proc. Conf. on Theoretical Aspects of Reasoning about Knowledge*. published in CoRR, http://arxiv.org/abs/1310.6437.

van Otterloo, S., and Jonker, G. 2005. On Epistemic Temporal Strategic Logic. *Electronic Notes in Theoretical Computer Science (ENTCS)* 126:77–92.

W. van der Hoek; Jamroga, W.; and Wooldridge, M. 2005. A logic for strategic reasoning. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems (AAMAS'05)*, 157–164.

Wittbold, J. T., and Johnson, D. M. 1990. Information flow in nondeterministic systems. In *Proc. IEEE Symp. on Security and Privacy*, 144–161.

Zdancewic, S., and Myers, A. C. 2001. Robust declassification. In *IEEE Computer Security Foundations Workshop*, 15.