On the Complexity of Knowledge Condition Games

Sieuwert van Otterloo Michael Wooldridge
University of Liverpool
Department of Computer Science
Liverpool, L69 3BX United Kingdom
00 44 151 794 3706 (fax 3715)
{sieuwert,mjw}@csc.liv.ac.uk

Abstract

Understanding the flow of knowledge in multi agent protocols is essential when proving the correctness or security of such protocols. Current logical approaches, often based on model checking, are well suited for modeling knowledge in systems where agents do not act strategically. Things become more complicated in strategic situations. In this paper we show that such situations can best be understood as a special type of game, a *knowledge condition game*. This paper summarizes some results on these games. Two proofs are sketched for the computational complexity of deciding whether a coalition can win a knowledge condition game with and without opponents (Σ_2P-complete and NP-complete respectively). We also consider a variant of knowledge condition games in which agents do not know which strategies are played, and prove that under this assumption, the presence of opponents does not affect the complexity. The decision problem without opponents is still NP-complete, but for a different reason.

1 Introduction

Knowledge condition games are extensive games with a special twist. Instead of assuming that agents win a play of a game if a certain outcome is reached, they win or lose if, at the end of the play, a certain knowledge state has been reached [22]. Many interesting questions about multi agent protocols can be formulated in this way. One could for instance apply this idea to distributed election protocols where all agents are hoping to learn the outcome before any other agent, or the case of an anonymous auction [3]. One goal of this paper is to encourage the reader to look at a protocol as a structure over which various games can be played. The security of the protocol then depends on who can win these games.

One way to interpret game theory is to consider games to be models for interaction between self-interested agents. As with all models, the level of detail can be varied. In order to make the problem suitable for a logical approach, we have made simplifying assumptions. We define a knowledge condition game as a constant-sum game between two teams, and assume that only two outcomes are possible: either the first team wins or the second team wins. In either case the other team loses, so the teams have mutually exclusive goals. We also disregard the exact probabilities of winning that the coalitions might have, but only ask ourselves whether the first coalition can win with certainty.

The structures over which knowledge games are played are 'game trees', or more formally, extensive game forms with imperfect information [9]. These structures specify which agent can act at a point in the protocol, and which actions this agent can choose from. A 'game tree' does not contain any preferences or winning conditions. Instead of defining these winning conditions in a conventional

way (by saying how good each single outcomes is for each agent), we calculate what each agent knows by the end of the protocols. This depends on the strategies that agents have chosen. We can then specify a certain knowledge property (i.e. 'A knows x and B does not know y'), and say that the first group of agents wins if this property holds.

Knowledge properties can be expressed using epistemic logic [8]. Epistemic logic originated from philosophy [7] but has been successfully used in the domain of computer science to capture the knowledge of agents in interpreted systems [5]. It has been combined with temporal logic [14] and coalition logic [16] in order to capture knowledge development over time, to capture knowledge change after announcements [1] or knowledge evolution in games such as Cluedo [18]. These frameworks have been applied to a range of different situations, such as the Russian Cards problem [19, 20] and the Dining Cryptographers Problem [17]. What we do is in one sense a simplification of these approaches, because using temporal logic one can express more things than just 'At the end of the protocol ...'. In another sense it is an extension, because we explicitly deal with strategies, and take into account that agents may know which strategies are being used. In fact, knowledge condition games have been designed as a response to the problems with ATEL [16]. ATEL is a logic that includes operators for knowledge, time and strategies. Because of the way it is designed, it is implicitly assumed in ATEL that the epistemic relations between states do not depend on the strategies that are played. In ATEL it is for instance not necessarily the case that agents know their own strategy. In knowledge condition games, we have been able to deal more explicitly with this strategic aspect, and secondly we have introduced opponents. The long term goal of this research is to get a good understanding of the interaction between knowledge and strategies, and perhaps this will ultimately lead to a refinement of ATEL to incorporate these aspects, or maybe to a replacement of ATEL by something more similar to knowledge condition games.

In this paper we do not focus on the logical properties of knowledge condition games. Instead we are trying to understand the computational complexity of determining who wins a knowledge condition game under various assumptions. The reason for this is two-fold. First of all, it is important to know for applications such as automated verification whether these problems are tractable. Secondly the complexity results tell us how these games are different from other games or approaches, and what makes these problems difficult.

In the next section (section 2) we formally define what a knowledge condition game is. In section 3 we give examples. In section 4 we present what we know to date about the computational complexity of knowledge condition games, and section 5 is the conclusion.

2 Definitions

In this section we define how one can create a knowledge condition game G from an interpreted game form F. In order to do this one needs a condition ϕ, a set of agents Γ that want ϕ to hold, and a set of agents Ξ that does not want ϕ to hold. We use the notation $G = kcg(F, \Gamma, \Xi, \phi)$ for this game.

In our notation, Σ denotes the set of all agents, Γ and Ξ denote coalitions of agents, X denotes a single agent, P is a set of atomic propositions and π is used for an interpretation function. In order to express knowledge conditions we use the language of epistemic propositional logic, along with its $S5_n$ semantics [5, 8] and this language is called $\mathcal{L}$ in this paper.

Definition 1. *Let P be a finite set of atomic propositions and Σ a finite set of agents. The language $\mathcal{L} = \mathcal{L}(\Sigma, P)$ of epistemic logic over P and Σ is the smallest set L such that $P \subset L$ and for any $\phi, \psi \in L$ and $X \in \Sigma$, also $\phi \vee \psi \in L$, $\neg\phi \in L$ and $K_X\phi \in L$*

An example formula in this language is $\phi_0 = K_B p$. This formula expresses that B knows that p holds. Conjunction $\phi \wedge \psi$ is defined as $\neg(\neg\phi \vee \neg\psi)$, and we define $M_X\phi$ (X considers it possible that ϕ) as $M_X\phi = \neg K_X \neg\phi$. This logic is interpreted over Kripke models [8].

Definition 2. *A (multi-agent) Kripke model M is a tuple $M = (\Sigma, S, \sim, P, \pi)$, where Σ is a set of agents, S is a set of states, $\sim$ is a collection of equivalence relations $\sim_X$ between histories, one for each agent $X \in \Sigma$, P is a set of atomic propositions and $\pi : S \to 2^P$ is an interpretation function.*

The meaning of the equivalence relations is that it relates states that cannot be distinguished by an agent: $s \sim_X t$ means that the states s and t "look the same" according to X. The function π returns for all states s a set $\pi(s) \subseteq P$ with the atomic propositions that are true in s.

An example Kripke model is the model M_0, which has two states s_1 and s_2 and two agents A and B. Agent A can distinguish these states, but agent B cannot: $s_1 \sim_B s_2$. In this model only one atomic proposition, proposition p, occurs. This proposition only holds in state s_1.

Epistemic formulas $\phi \in \mathcal{L}$ can be interpreted over Kripke models. Let $M = (\Sigma, S, \sim, P, \pi)$ be a Kripke model and $s \in S$. We write $M, s \models \phi$ if ϕ is true in s. This relation is defined by the following.

$$
\begin{array}{lll}
M, s \models p & \text{iff} & p \in \pi(s) \\
M, s \models \phi \vee \psi & \text{iff} & M, s \models \phi \text{ or } M, s \models \psi \\
M, s \models \neg\phi & \text{iff} & \text{not } M, s \models \phi \\
M, s \models K_X\phi & \text{iff} & \forall t \in S : s \sim_X t \implies M, t \models \phi
\end{array}
$$

We can use this definition for instance to show that in the example model we have that $M_0, s_1 \models K_A p$, $M_0, s_1 \models \neg K_B p$ and $M_0, s_2 \models K_B(p \vee \neg p)$.

An interpreted game form is basically a game form to which an interpretation function for atomic propositions has been added. A game form can be described in different but equivalent ways, for instance as a set of runs or as a game tree. We follow Osborne and Rubinstein [9] and use the idea of a set H of runs h. Each run h is one possible sequence of actions by agents that is allowed by the rules of the game. The whole set H of them fully describes what can be done in the game.

Definition 3. *A set of finite sequences H is* prefix-closed *if for any sequence h and action a it is the case that $ha \in H$ implies $h \in H$. For any set of sequences H and $h \in H$ we define the set of next actions $A(H, h) = \{a | ha \in H\}$ and the set of terminal sequences $Z(H) = \{h \in H | A(H, h) = \emptyset\}$.*

We use sequences of actions to denote specific plays of a game. One can also call such sequences histories or runs. $Z(H)$ denotes the set of all sequences that cannot be extended. These are called terminal histories or sequences, and correspond to outcomes.

Definition 4. *An* interpreted extensive game form *F is a tuple $F = (\Sigma, H, turn, \sim, \pi)$, where Σ is a finite set of agents, H is a non-empty, prefix-closed set of finite sequences, $turn$ is a function $turn : H \setminus Z(H) \to \Sigma$, for each $X \in \Sigma$ the relation $\sim_X \subseteq H \times H$ is an equivalence relation between states and $\pi : Z(H) \to 2^P$ returns the true atomic propositions of any terminal history. The following condition must hold: if $turn(h) = X$ and $h' \sim_X h$ then also $turn(h') = X$ and $A(H, h) = A(H, h')$.*

This definition is adapted from Osborne and Rubinstein [9], definition 200.1, where it is called an extensive game form. We have extended the information sets such that agents also have information when they are not in charge, which is a common extension for logical purposes [13, 2].

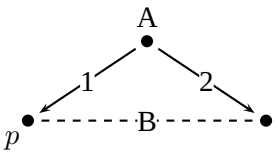

Figure 1: Interpreted game form F_0

The idea is that these propositions can be used to refer to certain histories, for instance to histories where an agent achieves a certain goal. The idea of annotating end states or terminal runs with logical propositions has been used before by Harrenstein e.a. [6] and the authors [22]. Approaches based on temporal logic [14, 15] often annotate all nodes of the model with propositions, so that formulas can be interpreted anywhere in the model.

An example interpreted game form F_0 is depicted in figure 1. In this example, agent A can make a choice from two alternatives (numbered 1 and 2), one of which satisfies p. After this choice, A can distinguish these situations, but B cannot.

For every game form F we can calculate a Kripke model $M = m(F)$ representing the knowledge in the end states of F. We do this by taking all the terminal histories of F as the set of states of M.

Definition 5. *Let $F = (\Sigma, H, turn, \sim, P, \pi)$. The end situation model m is defined as $m(F) = (\Sigma, Z(T), \sim, P, \pi)$.*

If we apply the function m to the example game form F_0, we get the example Kripke model $M_0 = m(F_0)$. The transformation m is used to express when a game form F makes a formula ϕ true. We say that a game form F makes a formula ϕ true, written $F \models \phi$, if ϕ holds after every terminal history of F: $\forall s \in Z(T)\ m(F), s \models \phi$.

Strategies are an important part of every game. Informally a strategy σ_Γ is a function that tells all agents in coalition Γ what to do next in the histories they control. We use nondeterministic strategies for our agents. This means that a strategy does not return a unique option that the agent should take, but it returns a set of options, with the intention that the agent should randomly select an element of this set.

Definition 6. *A strategy σ_Γ is a function such that for any node h with $turn(h) \in \Gamma$ it returns a non-empty set $\sigma_\Gamma(h) \subseteq A(H, h)$. An extra condition is that $h \sim_{turn(h)} h'$ must imply that $\sigma_\Gamma(h) = \sigma_\Gamma(h')$.*

The condition states that a strategy should not prescribe different options for histories that an agent cannot distinguish. An agent would not have the knowledge to adhere to a strategy that does not satisfy this condition.

For the example game form F_0 there are three different strategies $\sigma_{\{A\}}$ for agent A. The strategy can either tell the agent to take the first option, or it can prescribe the second option, or the strategy can express that the agent should randomly choose between both options. Formally these possibilities are defined by respectively $\sigma^1_{\{A\}}(\epsilon) = \{1\}$, $\sigma^2_{\{A\}}(\epsilon) = \{2\}$ and $\sigma^3_{\{A\}}(\epsilon) = \{1, 2\}$.

For any strategy σ_Γ for a game form F we can consider a restricted game form F' in which the agents $X \in \Gamma$ only choose options that are part of the strategy. The agents $Y \notin \Gamma$ can still do whatever they want in F'. Such a restricted game form models the situation in which coalition Γ is committed to the given strategy. The restricted model F' is computed by an update function $F' = u(F, \sigma_\Gamma)$.

Definition 7. *Let $F = (\Sigma, H, turn, \sim, P, \pi)$. We define u by $u(F, \sigma_\Gamma) = (\Sigma, H', turn', \sim', P, \pi')$, where H' is the smallest subset of H such that $\epsilon \in H'$ and for each $h \in H'$ and $a \in A(H, h)$: if $turn(h) \notin \Gamma$ or $a \in \sigma_\Gamma(h)$ then $ha \in H'$). $\sim'$ is such that for all X: $\sim'_X = \sim_X \cap (H' \times H')$, and $turn'$ and π' are the same as $turn$ and π but with their domain restricted to H'*

An update of the example F_0 with strategy $\sigma^3_{\{A\}}$ does not change anything: $u(F_0, \sigma^3_{\{A\}}) = F_0$. An update with $\sigma^1_{\{A\}}$ returns a model F_1 with only two histories: ϵ and 1. This means that the Kripke model of F_1 only has one state in which p holds: $m(u(F_0, \sigma^1_{\{A\}})) \models K_B p$

As promised at the start of the section, the function $G = kcg(F, \Gamma, \Xi, \phi)$ defines a knowledge condition game in which Γ wishes to achieve ϕ, while Ξ hopes to prevent it. The game G is not an extensive game, but a game in normal or strategic form [9]. It is not possible to consider G as an extensive game, because whether the knowledge condition holds is not a local property of each end state.

Definition 8. *A* strategic game *G is a tuple $(\Sigma, \{S\}_\Sigma, \mathfrak{U})$ where Σ is a finite set of agents, for each $X \in \Sigma$, S_X is a set of strategies for agent X and $\mathfrak{U} : S^\Sigma \to \mathbb{R}^\Sigma$ a function that for each choice of strategies returns a payoff vector.*

We are only interested in two-player, constant-sum, 0-1 games, and in these games only two payoff vectors are possible: $(1, 0)$ which is best for the first player, and $(0, 1)$ which is best for the second player.

Definition 9. *Let $F = (\Sigma, H, turn, \sim, \pi)$ be an interpreted extensive game form, $\Gamma \subseteq \Sigma$ a coalition of proponent agents, $\Xi \subseteq \Sigma \setminus \Gamma$ a coalition of opponent agents and $\phi \in \mathcal{L}$ a knowledge formula. The knowledge game $kcg(M, \Gamma, \Xi, \phi)$ is defined as $kcg(M, \Gamma, \Xi, \phi) = (\{\Gamma, \Xi\}, \{S_\Gamma, S_\Xi\}, \mathfrak{U})$ where S_Γ consists of all strategies of Γ in F, S_Ξ contains all strategies for Ξ and*

$$\mathfrak{U}(\sigma_\Gamma, \sigma_\Xi) = \begin{cases} (1,0) & \text{if } u(u(G, \sigma_\Gamma), \sigma_\Xi) \models \phi \\ (0,1) & \text{otherwise} \end{cases}$$

Let F_0 be the example game form and take $\phi_0 = K_B p$. For the game $G_0 = kcg(F_0, \{A\}, \emptyset, \phi_0)$ we can compute a payoff matrix. As calculated before, $\{A\}$ has three strategies. The empty coalition has only the function $f_\emptyset$ as a strategy. This function $f_\emptyset$ is the function with an empty domain.

	$\sigma^1_{\{A\}}$	$\sigma^2_{\{A\}}$	$\sigma^3_{\{A\}}$
$f_\emptyset$	(1,0)	(0,1)	(0,1)

We see that for this game, $\{A\}$ has a winning strategy (namely $\sigma^1_{\{A\}}$). The existence of a winning strategy for the first player of a game is denoted by $w(G) = 1$. We define this to be the case if $\exists \sigma_\Gamma \forall \sigma_\Xi \, \mathfrak{U}(\sigma_\Gamma, \sigma_\Xi) = (1, 0)$.

3 Examples

3.1 Anonymous Voting

A voting protocol can be used when a group of agents has to make a joint decision on a certain issue. A standard protocol is majority voting: each agent can vote for an option and the option that gets the most votes is the outcome of the protocol. In the example interpreted game form F_V, three agents

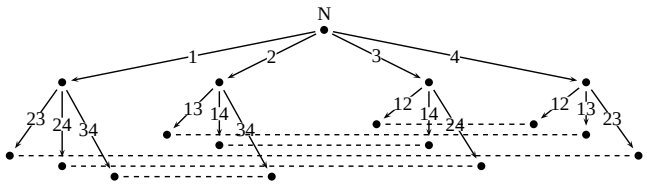

Figure 2: The fifty fifty problem F_Q

A, B and C use this protocol to decide whether plan $\mathcal{P}$ should be accepted or not. Each agent has a choice from two action: support (s) the plan, or reject it (r). They vote in alphabetical order, so first A chooses from action s or r, then B (without knowing A's choice) chooses either s or r and finally C does the same, unaware of what A, B did. This protocol thus has eight terminal runs. The proposition p indicates whether $\mathcal{P}$ is accepted and p holds if at least two agents choose s. Furthermore a holds if A chooses s, b if B chooses s and the same for C with c. The interpretation function is thus $\pi(sss) = \{a, b, c, p\}, \pi(ssr) = \{a, b, p\} \ldots \pi(rrr) = \emptyset$. We assume that $s \not\sim_X s'$ if s and s' differ in the evaluation of the outcome p, or if the vote of X differs in s from that in s'.

The following game results hold.

$$w(kcg(F_V, \{A, B\}, \{C\}, p)) = 1$$
$$w(kcg(F_V, \{A, B\}, \{C\}, K_B c \vee K_B \neg c)) = 1$$
$$w(kcg(F_V, \{B\}, \{C\}, K_B c \vee K_B \neg c)) = 0$$

A and B together can ensure that p is true, by voting s and s. What they also can do is vote differently,so that a and $\neg b$ result. In this case the outcome will solely depend on C's choice. They thus learn what C voted. Agent B cannot achieve this on its own.

One example, described by Schneier [11], p. 133, is a voting protocol where B would have the option of copying A's (encrypted) vote. In that case one might get

$$w(kcg(F'_V, \{B\}, \{A, C\}, K_B a \vee K_B \neg a)) = 1$$

This is an unwanted property and thus a 'bug' in the protocol. It is necessary to reason about knowledge to express this bug, so a standard game-theoretic analysis might not have revealed this shortcoming.

3.2 Fifty fifty Problem

> In a weekly broadcasted TV show the quiz master asks a candidate the next question: Which day of the week comes directly after Tuesday? Is that a) Monday, b) Wednesday, c) Friday or d) Saturday. The candidate replies: 'I am not sure. Can I do fifty fifty?'. The quiz master has to remove two options that are not the answer, so he says: 'The answer is not Monday and neither Friday'. Does the candidate know the answer?

This situation frequently occurs on television in several European countries in the 'Millionaire show'. One can also consider this situation to be a metaphor for a multi agent information exchange situation. Let us model this in an interpreted game form F_Q involving an agent N (nature) that determines what the right answer is, a quiz master Q that eliminates two answers, and a candidate C. This interpreted game form is depicted in figure 2. First nature selects one of the answers to be the right answer. it can choose from the actions $1, 2, 3$ and 4. The quiz master, who knows the right answer,

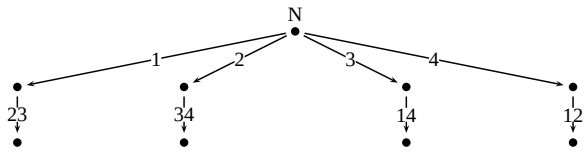

Figure 3: The updated interpreted game form $u(F_Q, \sigma_{\{Q\}})$

can then select an action ij that indicates that the two options i and j are eliminated. i, j must be different from the right answer. The terminal histories are thus all histories (k, ij). For such histories, $(k, ij) \sim_C (k', i'j')$ if the same options are eliminated: $ij = i'j'$. The set of atomic propositions is $P = \{a_i | 1 \leq i \leq 4\} \cup \{e_i | 1 \leq i \leq 4\}$, and each terminal history is interpreted in the following way: $\pi((k, ij)) = \{a_k, e_i, e_j\}$. The question is whether the candidate knows the answer at the end of the protocol. This is expressed by $\psi = K_C a_1 \vee K_C a_2 \vee K_C a_3 \vee K_C a_4$. The following table lists several properties of this situation.

Nature may favour the candidate:	$w(kcg(F_Q, \{N\}, \emptyset, \psi)) = 1$
Nature may not favour the candidate:	$w(kcg(F_Q, \{N\}, \emptyset, \neg\psi)) = 1$
The quiz master can help the candidate:	$w(kcg(F_Q, \{Q\}, \emptyset, \psi)) = 1$

We see that the question whether the candidate knows the answer depends on nature and on the quiz master Q. If nature uses a deterministic strategy, in which for instance a_1 always holds, then the candidate knows that this is the right answer. However, if Nature uses the non-deterministic strategy in which each answer could be the right answer, the candidate will not know the answer.

It becomes more interesting if the quiz master gets involved. In this game the quiz master has the ability to signal the right answer to the candidate. Take for instance this strategy $\sigma_{\{Q\}}$.

$$\sigma_{\{Q\}}(1) = \{23\}$$
$$\sigma_{\{Q\}}(2) = \{34\}$$
$$\sigma_{\{Q\}}(3) = \{14\}$$
$$\sigma_{\{Q\}}(4) = \{12\}$$

This strategy tells the candidate exactly what the right answer is: The answer directly before the two eliminated options (assuming 4 comes before 1). The updated model $u(F_Q, \sigma_{\{Q\}})$ is given in figure 3. This strategy acts as a code between the candidate and the quiz master. It is the strategy that proves that $w(kcg(F_Q, \{Q\}, \emptyset, q)) = 1$. A practical conclusion one can draw is that one should not bet on this quiz if one does not know what the interests of the quiz master are.

4 Computational Complexity

Looking at computational complexity is interesting for two reasons. First of all it can tell you whether a certain problem is 'tractable', i.e. whether the problem can be solved in practice. Secondly it can tell you more about the problem, for instance whether something is very general problem (i.e whether the problem format can be used to formulate questions about many different situations, such as logic), or what aspect makes a problem difficult. In this section we are looking at the complexity of the *kcg decision problem*, which is the problem to decide for a game $kcg(F, \Gamma, \Xi, \phi)$ whether the first

coalition Γ has a winning strategy. we look at this problem under various assumptions. We report four theorems. The first theorem is concerned with the problem of deciding whether a coalition G can win a knowledge condition game with an empty set of opponents. This is called the *no-opponents* knowledge condition game decision problem. It turns out that this problem is already NP-complete, and thus not tractable. The second theorem states that the general kcg decision problem is even harder: with opponents the problem is Σ_2P-complete. For the third theorem we formulate a variant of the knowledge condition game, kcg'. In this variant agents do not know what strategies are played. This variant makes knowledge condition games more similar to ATEL. In the third theorem we claim that the no-opponents problem is as hard as the general problem. Both problems are $NP-$complete, which is the fourth theorem.

A problem is in the class NP if it can be solved in polynomial time (there is a time bound that is polynomial in the input size) by a nondeterministic Turing machine. In practice this means that we can check a solution in reasonable time, but maybe not compute the answer in reasonable time [10, 4]. Some problems in the class NP are $NP-$complete: each NP-complete problem can be expressed as an instance of these complete problems. These problems are thus at least as hard as any problem in the class NP. It is widely believed, but not yet proven, that these complete problems cannot be solved in polynomial time.

It is perhaps worthwhile to note that we encode game forms in an explicit way, by listing all histories. In reality protocols are often specified in an implicit way (for instance in some form of source code) and such representations can be more exponentially more efficient.

Theorem 1. *The problem to decide for given F, Γ and ϕ whether $w(kcg(F, \Gamma, \emptyset, \phi)) = 1$ is NP-complete.*

Proof. A formal proof of this theorem has been presented elsewhere [21]. Proving that this problem is in NP is more or less straightforward. In order to prove that the problem is indeed NP-hard, one can reduce the NP-complete problem 3SAT to a knowledge condition game decision problem. We sketch the procedure here and illustrate it with an example.

Consider the satisfiability of the 3SAT formula $\phi = (p \vee \neg q \vee r) \wedge (\neg q \vee \neg p \vee r)$. This formula contains three propositions. We construct a game form with one agent A in which for each atomic proposition p two end nodes appear: p^+ indicates that truth of p, and p^- indicates that p is false. A nondeterministic strategy, that selects for each atomic proposition the positive or negative version (but not both) can now encode a satisfying assignment for the original 3SAT formula. The game form is depicted in figure 4. The second part of the procedure is to find a formula ψ_K such that A can ensure ψ_K exactly if the 3SAT formula ϕ has a satisfying assignment. This formula ψ_K consists of two parts. The first, consisting of a conjunction of for each proposition p the expression $(M_A p^+ \vee M_A p^-) \wedge \neg(M_A p^+ \wedge M_A p^-)$, expresses that any winning strategy must correspond to an assignment. The second part is a translation of ϕ. The whole formula is given below.

$$\begin{aligned}\psi_K = &(M_A p^+ \vee M_A p^-) \wedge \neg(M_A p^+ \wedge M_A p^-) \wedge \\ &(M_A q^+ \vee M_A q^-) \wedge \neg(M_A q^+ \wedge M_A q^-) \wedge \\ &(M_A r^+ \vee M_A r^-) \wedge \neg(M_A r^+ \wedge M_A r^-) \wedge \\ &M_A(p^+ \vee q^- \vee r^+) \wedge M_A(q^- \vee p^- \vee r^+)\end{aligned}$$

□

A typical NP-complete problem is to determine whether a propositional logic formula is satisfiable. Suppose ϕ is a formula with atomic propositions $x_1, x_2, \ldots x_n$. We can thus write $\phi = \phi(\vec{x})$ where the vector $\vec{x}$ consists of all the x_i. The satisfaction problem can now be phrased as deciding

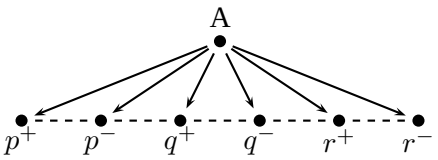

Figure 4: The model of 3SAT formula ψ

whether $\exists\vec{x} : \phi(\vec{x})$. In the same way we can formulate more difficult problems, by allowing more quantifiers: $\exists y \forall x : \phi(\vec{x}, \vec{y})$ is the problem where one has to decide whether there is an $\vec{x}$ such that $\phi(\vec{x}, \vec{y})$ is true for all $\vec{y}$. This problem, called SAT_2, is a typical Σ_2P complete problem [10](chapter 17). It is widely believed that these problems are strictly more difficult than NP-complete problems.

Theorem 2. *The problem to decide for given F, Γ, Ξ and ϕ whether $w(kcg(F, \Gamma, \Xi, \phi)) = 1$ is Σ_2P-complete.*

Proof. The proof for this theorem is based on a reduction from SAT_2. We sketch the procedure here, and will publish the detailed proof later. We need to show, to prove that this problem is in Σ_2P, that a kcg problem $w(kcg(F, \Gamma, \emptyset, \phi)) = 1$ can be expressed in the following form $\exists\vec{y}\forall\vec{x} : \psi(\vec{x}, \vec{y})$, where the predicate $\psi(\vec{x}, \vec{y})$ can be computed in polynomial time. The idea is that the propositional formula $\psi(\vec{x}, \vec{y})$ can express whether the end situation after playing strategies σ_Γ and σ_Ξ satisfies the formula ϕ. The variables $\vec{y}$ can be used for encoding the strategy of Γ, and $\vec{x}$ for the strategy of Ξ.
The second part of the proof is to show that the kcg problem is indeed complete for this class, and this can be done by reducing SAT_2 to a knowledge condition game. The proof is similar to the previous NP-completeness proof, but now involves two agents. Assume a SAT_2 problem $\exists\vec{y}\forall\vec{x} : \psi(\vec{x}, \vec{y})$ is given, such that ψ is in conjunctive normal form with three literals per conjunct. We define a game form in the following way: Agent A takes the first move of every play and can choose one action from the $\{x_0^+, x_0^-, x_1^+, x_1^-, \ldots\}$ where the propositions x_i are all propositions that occur in $\vec{x}$. After the action of agent A, agent B has a choice from a similar set of actions $\{y_0^+, y_0^-, y_1^+, y_1^-, \ldots\}$. After these two moves the game ends. Each terminal history is thus of the form $(x_i^\pm, y_j^\pm)$, and there are $2|\vec{x}| \cdot 2|\vec{y}|$ terminal histories. The size of the tree is thus at most quadratic in the size of the input. We assume that agent B does not know what A has done, so $(x_i^\pm) \sim_B (x_j^\pm)$ for all i, j. The formula ϕ of the knowledge condition game is now the following: $\phi = (\neg\phi^B) \vee (\phi^A \wedge f(\psi(\vec{x}, \vec{y}))$, where ϕ^B expresses that B's strategy is a valid assignment, ϕ^A expresses that A's strategy represents a valid assignment, and $f(\psi(\vec{x}, \vec{y}))$ is a translation of $\psi(\vec{x}, \vec{y})$ as the one used in the previous proof. It is not hard to verify that the winning condition for this knowledge condition game $\exists\sigma_{\{A\}}\forall\sigma_{\{B\}}\mathfrak{U}(\sigma_{\{A\}}, \sigma_{\{B\}}) = (1, 0)$ corresponds to $\exists\vec{y}\forall\vec{x} : \psi(\vec{x}, \vec{y})$. □

In the two previous proofs, it is essential that the agents are aware of the strategies they choose. To prove that this is indeed essential, we modify the definition of a knowledge condition game such the epistemic relations between terminal histories do not depend on the strategies chosen. The choice of strategies affects which states will be reached, but does not change which states are considered possible. This defines a different type of knowledge condition game kcg'.

Instead of saying that

$$\mathfrak{U}(\sigma_\Gamma, \sigma_\Xi) = (1, 0) \Leftrightarrow \forall z \in Z(u(u(G, \sigma_\Gamma), \sigma_\Xi)) : m(u(u(G, \sigma_\Gamma), \sigma_\Xi)), z \models \phi$$

as we have done, we say that the payoff is 1 if for any reached state of the unmodified model, the next formula holds:

$$\mathfrak{U}'(G, \sigma_\Gamma, \sigma_\Xi) = (1, 0) \Leftrightarrow \forall z \in Z(u(u(G, \sigma_\Gamma), \sigma_\Xi)) : m(H), z \models \phi$$

This alternative assumption leads to a modified definition of a knowledge condition game $kcg'()$. This new definition is more similar to what can be expressed in ATEL. One would perhaps expect that the computational complexity of this problem would be lower. Indeed one can prove that in this case it does not matter whether there are opponents.

Theorem 3. *Assume that F, Γ, Ξ and ϕ are given. $w(kcg'(F, \Gamma, \Xi, \phi)) = 1$ iff $w(kcg'(F, \Gamma, \emptyset, \phi)) = 1$.*

Proof. Notice that the goal of coalition Ξ is to choose a strategy σ_Ξ such that $u'(G, \sigma_\Gamma, \sigma_\Xi) = (0, 1)$. Since u is defined using universal quantification over the set $Z(u(u(G, \sigma_\Gamma), \sigma_\Xi))$, the best thing to do for this coalition is to make sure that this set is as large as possible. In order to achieve this, σ_Ξ should choose the neutral strategy that allows any action: the strategy σ with $\sigma(h) = A(H, h)$. Since we have assumed that neutral agents can do any action, we might as well assume that the agents $X \in \Xi$ are neutral, and determine the value of the game $w(kcg'(F, \Gamma, \emptyset, \phi)) = 1$. □

We see thus that the presence of opponents is not relevant, and indeed in ATEL no distinction between opponents and neutral agents is made. The question is now whether solving the kcg' decision problem is still as hard as the original no-opponents kcg problem. The answer is yes. The no-opponents kcg' problem is also NP-complete. However the proof is different in an interesting way.

Theorem 4. *The problem to decide for given F, Γ and ϕ whether $w(kcg'(F, \Gamma, \emptyset, \phi)) = 1$ is NP-complete.*

Proof. We can prove that this problem is in NP by a similar argument as given for theorem 1. For the hardness result we again show a reduction from 3SAT. Assume that ϕ^3 is a propositional formula. We define a game form between two agents: an agent Q that asks questions, and an agent A that answers them. The proponent coalition is $\Gamma = \{A\}$ and Q is assumed to be neutral. Every terminal history is of the form (p, b, f, n). The first action p is chosen by agent Q and must be one atomic proposition of ϕ^3. The agent A must then reply by giving a boolean b. This indicates what truth value A has in mind for p. Then agent Q choses one triplet f that appears in ϕ^3. This triplet is of the form $(a \vee b \vee c)$. Agent A then has to choose which of these three parts it thinks should be true: either a or b or c. The trick however is that $\sim_A$ is defined in such a way that A does not know, when choosing a, b or c which answer it has given on its first turn. This agent thus risks giving inconsistent information. For instance in the history $(p, true, (\neg p \vee q \vee r), \neg p)$ agent A first says that p is true, and then says that it thinks that $\neg p$ holds. The goal of agent A in the game is to avoid these inconsistent histories. It is not hard to show that A only has a strategy for avoiding these inconsistent histories if there is a satisfiable assignment for ϕ^3. □

The proof sketched above is very similar to a proof given by Schobbens [12] for the NP-completeness of ATL with imperfect information. This corroborates our claim that this variant of knowledge condition games is closely related to ATL and ATEL. This proof exploits the fact that in games where coalitions do not have perfect recall, it is very difficult for agents and coalitions to coordinate their own actions.

5 Conclusion

By combining protocols and knowledge conditions into games, one can express properties of multi agent protocols relating to security and secrecy. In a knowledge condition game one can make fine distinctions between for instance neutral and opponent agents, and one can give examples where this distinction is significant. Therefore these games are a promising direction for future research into the interaction between knowledge and strategies.

The complexity results reported in this paper draw an interesting picture. There seems to be a computational cost for assuming that agents know strategies. The single agent decision problem is already intractable. The presence of opponents makes it even harder to computer whether a coalition can guarantee a property. If we drop this assumption and reformulate the notion of winning a knowledge condition game, then the extra complexity of adding opponents disappears. However the problem without opponents is still NP-complete and thus intractable, but for different reasons. The complexity proof is no longer based upon formulating a difficult knowledge formula, but on the hardness of coordinating in a game form without perfect recall.

Future research could focus on comparing decision problems for knowledge condition games to other game-theoretic decision problems, in order to establish what exactly the complexity cost is of considering knowledge goals. It would also be interesting to find out under which assumptions knowledge condition games can be solved in polynomial time. Other directions include looking at knowledge condition games from a logical viewpoint by searching for axioms, and to consider the mechanism design problem to find a game form with given properties.

References

[1] A. Baltag, L.S. Moss, and S. Solecki. The logic of public announcements, common knowledge and private suspicions. Originally presented at TARK 98, accepted for publication in Annals of Pure and Applied Logic, 2002.

[2] G. Bonanno. Memory implies von neumann-morgenstern games. *Knowledge Rationality and Action*, to appear, 2004.

[3] F. Brandt and T. Sandholm. (im)possibility of unconditionally privacy-preserving auctions. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 810–817, 2004.

[4] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press: Cambridge, MA, 1990.

[5] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about knowledge*. The MIT Press: Cambridge, MA, 1995.

[6] B. Harrenstein, W. van der Hoek, J.-J. Ch. Meyer, and C. Witteveen. A modal characterization of nash equilibrium. *Fundamenta Informaticae*, 57(2–4):281–321, 2003.

[7] J. Hintikka. *Knowledge and Belief: an introduction to the logic of the two notions*. Cornell University Press: Ithaca, NY, 1962.

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

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

[10] C. Papadimitriou. *Computational Complexity*. Addison Wesley Longman, 1994.

[11] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[12] Pierre-Yves Schobbens. Alternating-time logic with imperfect recall. In Wiebe van der Hoek, Alessio Lomuscio, Erik de Vink, and Mike Wooldrige, editors, *Electronic Notes in Theoretical Computer Science*, volume 85. Elsevier, 2004.

[13] J. van Benthem. Games in dynamic-epistemic logic. *Bulletin of Economic Research*, 53(4):219–248, 2001.

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

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

[16] W. van der Hoek and M. Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(4):125–157, 2003.

[17] R. van der Meyden and K. Su. Symbolic model checking the knowledge of the dining cryptographers. under review, 2004.

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

[19] H. P. van Ditmarsch. The russian cards problem. *Studia Logica*, 75(4):31–62, 2003.

[20] S. van Otterloo, W. van der Hoek, and M. Wooldridge. Model checking a knowledge exchange scenario. In *Proceedings of Model Checking and Artificial Intelligence(MoChArt)*, pages 37–44, Acapulco, August 2003.

[21] S. van Otterloo, W. van der Hoek, and M. Wooldridge. Knowledge condition games. In S. van Otterloo, P. McBurney, W. van der Hoek, and M. Wooldridge, editors, *Proceedings of the first Knowledge and Games Workshop*. University of Liverpool, 2004.

[22] S. van Otterloo, W. van der Hoek, and M. Wooldridge. Preferences in game logics. In *Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, New York, July 2004.