TU Clausthal
Clausthal University of Technology

Model Checking Abilities under Incomplete Information Is Indeed Δ_2^{P}-complete

Wojciech Jamroga and Jürgen Dix

IfI Technical Report Series **IfI-06-10**

IfI

Department of Informatics
Clausthal University of Technology

Impressum

Publisher: Institut für Informatik, Technische Universität Clausthal
Julius-Albert Str. 4, 38678 Clausthal-Zellerfeld, Germany
Editor of the series: Jürgen Dix
Technical editor: Wojciech Jamroga
Contact: wjamroga@in.tu-clausthal.de
URL: http://www.in.tu-clausthal.de/forschung/technical-reports/
ISSN: 1860-8477

The IfI Review Board

Model Checking Abilities under Incomplete Information Is Indeed $\mathbf{\Delta_2^P}$-complete

Wojciech Jamroga and Jürgen Dix

Department of Informatics, Clausthal University of Technology
Julius Albert Str. 4, D-38678 Clausthal Germany
{wjamroga,dix}@in.tu-clausthal.de

Abstract

We study the model checking complexity of *Alternating-time temporal logic with imperfect information and imperfect recall* (ATL_{ir}). Contrary to what we have stated in [10], the problem turns out to be $\mathbf{\Delta_2^P}$-complete, thus confirming the initial intuition of Schobbens. We prove the $\mathbf{\Delta_2^P}$-hardness through a reduction of the $\mathbf{SNSAT}$ problem, while the membership in $\mathbf{\Delta_2^P}$ stems from the algorithm presented in [16].

Keywords: multi-agent systems, model checking, temporal logic, strategic ability, computational complexity.

1 Introduction

Alternating-time temporal logic [1, 2] is one of the most interesting frameworks that emerged recently for reasoning about computational systems. ATL_{ir} is a variant of ATL, proposed by Schobbens in [16] for agents with *imperfect information* and *imperfect recall*. We have already investigated the complexity of ATL_{ir} model checking in [10], concluding that the problem is $\mathbf{NP}$-complete. Unfortunately, our claim was incorrect; we want to set it right with this paper.

We begin with a presentation of the frameworks of ATL and ATL_{ir}. Then we present some existing complexity results with respect to ATL_{ir} model checking, and we give an alternative proof of $\mathbf{NP}$-hardness of the problem. In Section 3.2, we extend the construction to present a reduction of $\mathbf{SNSAT}$, thus proving that model checking ATL_{ir} is $\mathbf{\Delta_2^P}$-hard. As the membership in $\mathbf{\Delta_2^P}$ stems from the algorithms presented in both [16] and [10], we get that model checking ATL_{ir} is $\mathbf{\Delta_2^P}$-complete.

ATL_{ir} can be seen as the "core", minimal ATL-based language for ability under incomplete information. In consequence, we obtain a lower bound for model checking of most (if not all) logics of this kind, and for most of them the bound is tight.

2 What Agents Can Achieve

ATL [1, 2] has been invented by Alur, Henzinger and Kupferman in order to capture properties of *open computational systems* (such as computer networks), where different components can act autonomously, and computations in such systems result from their combined actions. Alternatively, ATL can serve as a logic for systems involving multiple agents, that allows one to reason about what agents can achieve in game-like scenarios. As ATL does not include incomplete information in its scope, it can be seen as a logic for reasoning about agents who always have complete knowledge about the current state of affairs.

2.1 ATL: Ability in Perfect Information Games

ATL is a generalization of the branching time temporal logic CTL [3, 4], in which path quantifiers are replaced with so called *cooperation modalities*. Formula $\langle\langle A\rangle\rangle\varphi$, where $A \subseteq \mathbb{A}\mathrm{gt}$ is a coalition of agents, expresses that coalition A has a collective strategy to enforce φ. ATL formulae include temporal operators: "$\bigcirc$" ("in the next state"), $\Box$ ("always from now on") and $\mathcal{U}$ ("until"). Operator $\Diamond$ ("now or sometime in the future") can be defined as $\Diamond\varphi \equiv \top\,\mathcal{U}\,\varphi$. Like in CTL, every occurrence of a temporal operator is immediately preceded by exactly one cooperation modality.[1] The broader language of ATL*, in which no such restriction is imposed, is not used in this paper.

A number of semantics have been defined for ATL, most of them equivalent [5, 6]. In this paper, we refer to a variant of *concurrent game structures*, which includes a nonempty finite set of all agents $\mathbb{A}\mathrm{gt} = \{1, ..., k\}$, a nonempty set of states St, a set of atomic propositions Π, a valuation of propositions $\pi : \Pi \to \mathcal{P}(St)$, and the set of (atomic) actions Act. Function $d : \mathbb{A}\mathrm{gt} \times St \to \mathcal{P}(Act)$ defines nonempty sets of actions available to agents at each state, and o is a (deterministic) transition function that assigns the outcome state $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to state q and a tuple of actions $\langle\alpha_1, \ldots, \alpha_k\rangle$ that can be executed by the agent in q. A *strategy* s_a of agent a is a conditional plan that specifies what a is going to do for every possible state (i.e., $s_a : St \to Act$ such that $s_a(q) \in d_a(q)$).[2] A *collective strategy* S_A for a group of agents $A \subseteq \mathbb{A}\mathrm{gt}$ is a tuple of strategies, one per agent from A. A *path* λ in model M is an infinite sequence of states that can be reached by subsequent transitions, and refers to a possible course of action (or a possible computation) that may occur in the system; by $\lambda[i]$, we denote the ith position on path λ. Function $out(q, S_A)$ returns the set of all paths that may result from agents A executing strategy S_A from state q onward. Now, informally speaking, $M, q \models \langle\langle A\rangle\rangle\varphi$ iff there is a collective strategy S_A such that φ holds for every $\lambda \in out(q, S_A)$. In Section 2.3, we give a more precise semantic definition of ATL_{ir}, which is the main subject of our study.

[1] The logic to which such a syntactic restriction applies is sometimes called *"vanilla"* ATL (resp. "vanilla" CTL etc.).

[2] Note that in the original formulation of ATL [1, 2], strategies assign agents' choices to *sequences* of states, which suggests that agents can by definition recall the whole history of each game.

One of the most appreciated features of ATL is its model checking complexity – linear in the number of transitions in the model and the length of the formula. However, after a careful inspection, this result is not as good as it seems. This linear complexity is no more valid when we measure the size of models in *the number of states, actions and agents [9, 15]*, or when we represent systems with *concurrent programs [17]*. Still, we have the following.

Proposition 1 ([2]) *The* ATL *model checking problem is* PTIME*-complete, and can be done in time* $\mathbf{O}(ml)$*, where* m *is the number of transitions in the model and* l *is the length of the formula.*

2.2 Strategic Abilities under Incomplete Information

ATL and its models include no way of addressing uncertainty that an agent or a process may have about the current situation. Moreover, strategies in ATL can define different choices for any pair of different states, hence implying that an agent can recognize each (global) state of the system, and act accordingly. Thus, it can be argued that the logic is tailored for describing and analyzing systems in which every agent/process has *complete and accurate knowledge* about the current state of the system. This is usually not the case for most application domains, where a process can access its *local* state, but the state of the environment and the (local) states of other agents can be observed only partially.

One of the main challenges for a logic of strategic abilities under incomplete information is the question of how agents' knowledge should interfere with the agents' available strategies. The early approaches to "ATL with incomplete information" [2, Sec.7.2],[18, 19] did not handle this interaction in a completely satisfactory way (cf. [8, 16, 14]), which triggered a flurry of logics, proposed to overcome the problems [8, 11, 16, 14, 20, 7, 12]. Most of the proposals agree that only *uniform* strategies (i.e., strategies that specify the same choices in indistinguishable states) are really executable. However, in order to identify a successful strategy, the agents must consider not only the courses of actions starting from the current state of the system, but also those starting from states that are indistinguishable from the current one. There are many cases here, especially when group epistemics is concerned: the agents may have *common*, *ordinary* or *distributed* knowledge about a strategy being successful, or they may be hinted the right strategy by a distinguished member (the "boss"), a subgroup ("headquarters committee") or even another group of agents ("consulting company") etc. Most existing solutions treat only some of the cases (albeit rather in an elegant way), while the others offer a very general treatment of the problem at the expense of a complicated logical language (which is by no means elegant). We believe that an elegant and general solution has been recently proposed in the form of Constructive Strategic Logic [12, 13], but this claim is yet to be verified.

ATL$_{ir}$ stands out among the existing solutions for its simplicity. While by no means the most expressive, it can be treated as the "core", minimal ATL-based language for ability under incomplete information.

2.3 ATL$_{ir}$

ATL$_{ir}$ includes the same formulae as ATL, only the cooperation modalities are presented with a subscript: $\langle\langle A\rangle\rangle_{ir}$ to indicate that they address agents with imperfect *information* and imperfect *recall*. Formally, the recursive definition of ATL$_{ir}$ formulae is:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A\rangle\rangle_{ir}\bigcirc\varphi \mid \langle\langle A\rangle\rangle_{ir}\Box\varphi \mid \langle\langle A\rangle\rangle_{ir}\varphi\,\mathcal{U}\,\varphi$$

Again, we define $\langle\langle A\rangle\rangle_{ir}\Diamond\varphi \equiv \langle\langle A\rangle\rangle_{ir}\top\,\mathcal{U}\,\varphi$.

Models of ATL$_{ir}$, *imperfect information concurrent game structures* (i-CGS), can be presented as concurrent game structures augmented with a family of epistemic indistinguishability relations $\sim_a \subseteq St \times St$, one per agent $a \in \mathbb{A}\mathrm{gt}$. The relations describe agents' uncertainty: $q \sim_a q'$ means that, while the system is in state q, agent a considers it possible that it is in q' now. It is required that agents have the same choices in indistinguishable states. To recapitulate, i-CGS can be defined as tuples

$$M = \langle \mathbb{A}\mathrm{gt}, St, \Pi, \pi, Act, d, o, \sim_1, ..., \sim_k\rangle,$$

where:

- $\mathbb{A}\mathrm{gt} = \{1, ..., k\}$ is a finite nonempty set of all agents,
- St is a nonempty set of states,
- Π is a set of atomic propositions,
- $\pi : \Pi \to \mathcal{P}(St)$ is a valuation of propositions,
- Act is a finite nonempty set of (atomic) actions;
- function $d : \mathbb{A}\mathrm{gt} \times St \to \mathcal{P}(Act)$ defines actions available to an agent in a state; $d(a, q) \neq \emptyset$ for all $a \in \mathbb{A}\mathrm{gt}, q \in St$,
- o is a (deterministic) transition function that assigns outcome states to states and tuples of actions; that is, $o(q, \alpha_1, \ldots, \alpha_k) \in St$ for every $q \in St$ and $\langle\alpha_1, \ldots, \alpha_k\rangle \in d(1, q) \times \cdots \times d(k, q)$;
- $\sim_1, ..., \sim_k \subseteq St \times St$ are epistemic relations, one per agent. Every $\sim_a$ is assumed to be an equivalence. We require that $q \sim_a q'$ implies $d(a, q) = d(a, q')$.

Again, a (memoryless) strategy of agent a is a conditional plan that specifies what a is going to do in every possible state. An executable plan must prescribe the same choices for indistinguishable states. Therefore ATL$_{ir}$ restricts the strategies that can be used by agents to the set of so called uniform strategies. A *uniform strategy* of agent a is defined as a function $s_a : St \to Act$, such that: (1) $s_a(q) \in d(a, q)$, and (2) if $q \sim_a q'$

then $s_a(q) = s_a(q')$. A *collective strategy* for a group of agents $A = \{a_1, ..., a_r\}$ is a tuple of strategies $S_A = \langle s_{a_1}, ..., s_{a_r}\rangle$, one per agent from A. A collective strategy is uniform if it contains only uniform individual strategies. Again, function $out(q, S_A)$ returns the set of all paths that may result from agents A executing strategy S_A from state q onward: (the notation $S_A(a)$ stands for the strategy s_a of agent a in the tuple $S_A = \langle s_{a_1}, ..., s_{a_r}\rangle$)

$out(q, S_A) = \{\lambda = q_0q_1q_2... \mid q_0 = q$ and for every $i = 1, 2, ...$ there exists a tuple of agents' decisions $\langle \alpha_1^{i-1}, ..., \alpha_k^{i-1}\rangle$ such that $\alpha_a^{i-1} = S_A(a)(q_{i-1})$ for each $a \in A$, $\alpha_a^{i-1} \in d(a, q_{i-1})$ for each $a \notin A$, and $o(q_{i-1}, \alpha_1^{i-1}, ..., \alpha_k^{i-1}) = q_i\}$.

The semantics of ATL$_{ir}$ formulae is defined as follows:

$M, q \models p$ iff $q \in \pi(p)$ (for $p \in \Pi$);

$M, q \models \neg\varphi$ iff $M, q \not\models \varphi$;

$M, q \models \varphi \wedge \psi$ iff $M, q \models \varphi$ and $M, q \models \psi$;

$M, q \models \langle\langle A\rangle\rangle_{ir} \bigcirc \varphi$ iff there exists a uniform strategy S_A such that, for every $a \in A$, $q' \in St$ such that $q \sim_a q'$, and $\lambda \in out(S_A, q')$, we have $M, \lambda[1] \models \varphi$;

$M, q \models \langle\langle A\rangle\rangle_{ir} \Box \varphi$ iff there exists a uniform strategy S_A such that, for every $a \in A$, $q' \in St$ such that $q \sim_a q'$, and $\lambda \in out(S_A, q')$, we have $M, \lambda[i]$ for every $i \geq 0$;

$M, q \models \langle\langle A\rangle\rangle_{ir} \varphi \mathcal{U} \psi$ iff there exist a uniform strategy S_A such that, for every $a \in A$, $q' \in St$ such that $q \sim_a q'$, and $\lambda \in out(S_A, q')$, there is $i \geq 0$ for which $M, \lambda[i] \models \psi$, and $M, \lambda[j] \models \varphi$ for every $0 \leq j < i$.

That is, $\langle\langle A\rangle\rangle_{ir}\varphi$ if coalition A has a uniform strategy, such that for every path *that can possibly result from execution of the strategy*, φ is the case. This is a strong statement, because many paths can result. It suffices that at least one of the agents in A considers some states q, q' equivalent: Then all such paths have to be considered.

Note that the universal path quantifier A from CTL can be expressed in ATL$_{ir}$ as $\langle\langle\emptyset\rangle\rangle_{ir}$.

Example 1 (Gambling robots) *Two robots (a and b) play a simple card game. The deck consists of Ace, King and Queen (A, K, Q). Normally, it is assumed that A is the best card, K the second best, and Q the worst. Therefore A beats K and Q, K beats Q, and Q beats no card. At the beginning of the game, the "environment" agent deals a random card to both robots (face down), so that each player can see his own hand, but he does not know the card of the other player. Then robot a can exchange his card for the one remaining in the deck (action $exch$), or he can keep the current one ($keep$). At the same time, robot b can change the priorities of the cards, so that Q becomes better than A (action chg) or he can do nothing (nop), i.e. leave the priorities unchanged. If*

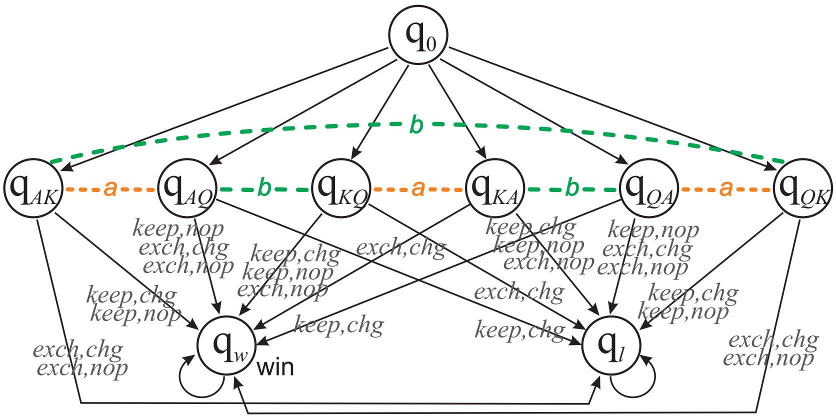

Figure 1: Gambling Robots game

a has a better card than b after that, then a win is scored, otherwise the game ends in a "losing" state. A CGS *M_1 for the game is shown in Figure 1.*

It is easy to see that $M_1, q_0 \models \neg\langle\langle a\rangle\rangle_{ir}\Diamond\mathsf{win}$, because, for every a's (uniform) strategy, if it guarantees a win in e.g. state q_{AK} then it fails in q_{AQ} (and similarly for other pairs of indistinguishable states). Let us also observe that $M_1, q_0 \models \neg\langle\langle a, b\rangle\rangle_{ir}\Diamond\mathsf{win}$ (in order to win, a must exchange his card in state q_{QK}, so he must exchange his card in q_{QA} too (by uniformity), and playing $exch$ in q_{QA} leads to the losing state. On the other hand, $M_1, q_{AQ} \models \langle\langle a, b\rangle\rangle_{ir}\bigcirc\mathsf{win}$ (a winning strategy: $s_a(q_{AK}) = s_a(q_{AQ}) = s_a(q_{KQ}) = keep$, $s_b(q_{AQ}) = s_b(q_{KQ}) = s_b(q_{AK}) = nop$; q_{AK}, q_{AQ}, q_{QK} are the states that must be considered by a and b in q_{AQ}). Still, $M_1, q_{AK} \models \neg\langle\langle a, b\rangle\rangle_{ir}\bigcirc\mathsf{win}$.

Schobbens [16] proved that ATL$_{ir}$ model checking is $\mathbf{NP}$-hard and $\mathbf{\Delta_2^P}$-easy. He also conjectured that the problem might be $\mathbf{\Delta_2^P}$-complete. We prove that it is indeed the case in Section 3.

3 Model Checking ATL$_{ir}$

Schobbens [16] proved that ATL$_{ir}$ model checking is intractable: more precisely, it is $\mathbf{NP}$-hard and $\mathbf{\Delta_2^P}$-easy (i.e., can be solved through a polynomial number of calls to an oracle for some problem in $\mathbf{NP}$) when the size of the model is defined in terms of the number of transitions. He also conjectured that the problem might be $\mathbf{\Delta_2^P}$-complete. In this section, we close the gap and prove that it is $\mathbf{\Delta_2^P}$-hard, and hence indeed $\mathbf{\Delta_2^P}$-complete. The proof proceeds by a reduction of the $\mathbf{SNSAT}$ problem to ATL$_{ir}$ model checking, presented in Section 3.2.

We have already investigated the complexity of ATL$_{ir}$ model checking in [10], concluding that the problem is $\mathbf{NP}$-complete. Unfortunately, our claim was incorrect: we want to set it right in this paper.

3.1 Existing Results

Model checking ATL_{ir} has been proved to be $\mathbf{NP}$-hard and $\mathbf{\Delta_2^P}$-easy in the number of transitions and the length of the formula [16]. The membership in $\mathbf{\Delta_2^P}$ was demonstrated through the following observation. If the formula to be model checked is of the form $\langle\langle A\rangle\rangle_{ir}\varphi$ (φ being $\bigcirc\psi$, $\Box\psi$ or $\psi_1\,\mathcal{U}\,\psi_2$), where φ contains no more cooperation modalities, then it is sufficient to guess a strategy for A, "trim" the model by removing all transitions that will never be executed (according to this strategy), and model check CTL formula $\mathsf{A}\varphi$ in the resulting model. Thus, model checking an arbitrary ATL_{ir} formula can be done by checking the subformulae iteratively, which requires a polynomial number of calls to an $\mathbf{NP}$ algorithm.[3]

The $\mathbf{NP}$-hardness follows from a reduction of the well known $\mathbf{SAT}$ problem. Here, we present a reduction which is somewhat different from the one in [16]. We will adapt it in Section 3.2 to prove $\mathbf{\Delta_2^P}$-hardness. In $\mathbf{SAT}$, we are given a CNF formula $\varphi \equiv C_1 \wedge \ldots \wedge C_n$ involving k propositional variables from set $X = \{x_1, ..., x_k\}$. Each clause C_i can be written as $C_i \equiv x_1^{s_{i,1}} \vee \ldots \vee x_k^{s_{i,k}}$, where $s_{i,j} \in \{+, -, 0\}$; x_j^+ denotes a positive occurrence of x_j in C_i, x_j^- denotes an occurrence of $\neg x_j$ in C_i, and x_j^0 indicates that x_j does not occur in C_i. The problem asks if $\exists X.\varphi$, that is, if there is a valuation of $x_1, ..., x_k$ such that φ holds.

We construct the corresponding i-CGS M_φ as follows. There are two players: verifier $\mathbf{v}$ and refuter $\mathbf{r}$. The refuter decides at the beginning of the game which clause C_i will have to be satisfied: it is done by proceeding from the initial state q_0 to a "clause" state q_i. At q_i, verifier decides (by proceeding to a "proposition" state $q_{i,j}$) which of the literals $x_j^{s_{i,j}}$ from C_i will be attempted. Finally, at $q_{i,j}$, verifier attempts to prove C_i by declaring the underlying propositional variable x_j true (action $\top$) or false (action $\bot$). If she succeeds (i.e., if she executes $\top$ for x_j^+, or executes $\bot$ for x_j^-), then the system proceeds to the "winning" state $q_\top$. Otherwise, the system stays in $q_{i,j}$. Additionally, "proposition" states referring to the same variable are indistinguishable for verifier, so that she has to declare the same value of x_j in all of them within a uniform strategy. A sole ATL_{ir} proposition yes holds only in the "winning" state $q_\top$. Obviously, states corresponding to literals x_j^0 can be omitted from the model.

Speaking more formally, $M_\varphi = \langle \mathbb{A}\text{gt}, St, \Pi, \pi, Act, d, o, \sim_1, ..., \sim_k\rangle$, where:

- $\mathbb{A}\text{gt} = \{\mathbf{v}, \mathbf{r}\}$,
- $St = \{q_0\} \cup St_{cl} \cup St_{prop} \cup \{q_\top\}$, where $St_{cl} = \{q_1, \ldots, q_n\}$, and $St_{prop} = \{q_{1,1}, \ldots, q_{1,k}, \ldots, q_{n,1}, \ldots, q_{n,k}\}$;
- $\Pi = \{\mathsf{yes}\}$, $\quad \pi(\mathsf{yes}) = \{q_\top\}$,
- $Act = \{1, ..., \max(k, n), \top, \bot\}$,

[3] The algorithm from [10] can be also used to demonstrate the upper bounds for the complexity of this problem.

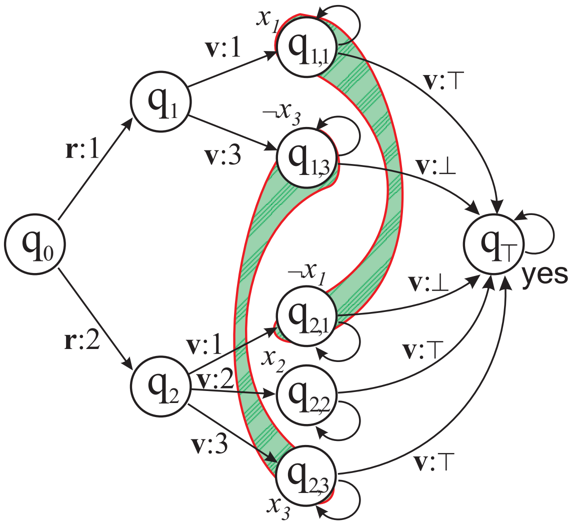

Figure 2: An i-CGS for checking satisfiability of $\varphi \equiv (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

- $d(\mathbf{v}, q_0) = d(\mathbf{v}, q_\top) = \{1\}$, $\quad d(\mathbf{v}, q_i) = \{1, ..., k\}$,
  $d(\mathbf{v}, q_{i,j}) = \{\top, \bot\}$,
  $d(\mathbf{r}, q) = \{1, ..., n\}$ for $q = q_0$ and $\{1\}$ otherwise;
- $o(q_0, 1, i) = q_i$, $\quad o(q_i, j, 1) = q_{i,j}$,
  $o(q_{i,j}, \top, 1) = q_\top$ if $s_{i,j} = +$, and $q_{i,j}$ otherwise,
  $o(q_{i,j}, \bot, 1) = q_\top$ if $s_{i,j} = -$, and $q_{i,j}$ otherwise;
- $q_0 \sim_{\mathbf{v}} q$ iff $q = q_0$, $\quad q_i \sim_{\mathbf{v}} q$ iff $q = q_i$, $\quad q_{i,j} \sim_{\mathbf{v}} q$ iff $q = q_{i',j}$.

As an example, model M_φ for $\varphi \equiv (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$ is presented in Figure 2.

Theorem 2 *φ is satisfiable iff* $M_\varphi, q_0 \models \langle\langle \mathbf{v} \rangle\rangle_{ir} \Diamond \mathsf{yes}$.

Proof. Firstly, if there is a valuation that makes φ true, then for every clause C_i one can choose a literal out of C_i that is made true by the valuation. The choice, together with the valuation, corresponds to a uniform strategy for $\mathbf{v}$ such that, for all possible executions, $q_\top$ is achieved at the end.

Conversely, if $M_\varphi, q_0 \models \langle\langle \mathbf{v} \rangle\rangle_{ir} \Diamond \mathsf{yes}$, then there is a strategy $s_{\mathbf{v}}$ such that $q_\top$ is achieved for all paths from $out(q_0, s_{\mathbf{v}})$. But then the valuation, which assigns propositions $x_1, ..., x_k$ with the same values as $s_{\mathbf{v}}$, satisfies φ. ■

Both the number of states and transitions in M_φ are linear in the length of φ, and the construction of M requires linear time too. Thus, the model checking problem for

ATL$_{ir}$ is $\mathbf{NP}$-hard. Note that it is $\mathbf{NP}$-hard even for formulae with a single cooperation modality, and turn-based models with at most two agents.[4]

We already investigated the complexity of ATL$_{ir}$ model checking in [10], concluding that the problem was $\mathbf{NP}$-complete. Unfortunately, our claim was incorrect: the error occurred in the way we handled negation in our model checking algorithm (cf. [15]). Still, as observed by Laroussinie, Markey and Oreiby in [15], our algorithm *is* correct for "positive ATL$_{ir}$" – i.e., ATL$_{ir}$ without negation. Thus, the following holds.

Proposition 3 *Model checking of "positive* ATL$_{ir}$*" is* $\mathbf{NP}$*-complete with respect to the number of transitions in the model and the length of the formula.*

The $\mathbf{\Delta_2^P}$-hardness for the full ATL$_{ir}$ is proved in Section 3.2.

3.2 Model Checking ATL$_{ir}$ Is Indeed $\mathbf{\Delta_2^P}$-complete

Let us first recall (after [15]) the definition of $\mathbf{SNSAT}$, a typical $\mathbf{\Delta_2^P}$-hard problem.

Definition 1 (SNSAT)
Input: *p sets of propositional variables $X_r = \{x_{1,r}, ..., x_{k,r}\}$, p propositional variables z_r, and p Boolean formulae φ_r in CNF, with each φ_r involving only variables in $X_r \cup \{z_1, ..., z_{r-1}\}$, with the following requirement:*

$z_r \equiv$ *there exists an assignment of variables in X_r such that φ_r is true.*

We will also write, by abuse of notation, $z_r \equiv \exists X_r\ \varphi_r(z_1, ..., z_{r-1}, X_r)$.
Output: *The truth-value of z_p (i.e., $\top$ or $\bot$).*

Let n be the maximal number of clauses in any $\varphi_1, ..., \varphi_p$ from the given input. Now, each φ_r can be written as:

$$\varphi_r \equiv C_1^r \wedge \ldots \wedge C_n^r, \text{ and } C_i^r \equiv x_{1,r}^{s_{i,1}^r} \vee \ldots \vee x_{k,r}^{s_{i,k}^r} \vee z_1^{s_{i,k+1}^r} \vee \ldots z_{r-1}^{s_{i,k+r-1}^r}.$$

Again, $s_{i,j}^r \in \{+, -, 0\}$; x^+ denotes a positive occurrence of x, x^- denotes an occurrence of $\neg x$, and x^0 indicates that x does not occur in the clause. Similarly, $s_{i,k+j}^r$ defines the "sign" of z_j in clause C_i^r. Given such an instance of $\mathbf{SNSAT}$, we construct a sequence of concurrent game structures M_r for $r = 1, ..., p$ in a similar way to the construction in Section 3.1. That is, clauses and variables $x_{i,r}$ are handled in exactly the same way as before. Moreover, if z_i occurs as a positive literal in φ_r, we embed M_{φ_i} in M_r, and add a transition to the initial state q_0^i of M_i. If $\neg z_i$ occurs in φ_r, we do almost the same: the only difference is that we split the transition into two steps, with a state neg_i^r (labeled with an ATL$_{ir}$ proposition neg) added in between.

More formally, $M_r = \langle \mathbb{A}\mathrm{gt}, St^r, \Pi, \pi^r, Act^r, d^r, o^r, \sim_1^r, ..., \sim_k^r \rangle$, where:

[4] In fact, it is $\mathbf{NP}$-hard even for models with a single agent, although the construction must be a little different to demonstrate this.

- $\mathbb{A}\mathrm{gt} = \{\mathbf{v}, \mathbf{r}\}$,
- $St^r = \{q_0^r, q_1^r, \dots, q_n^r, q_{1,1}^r, \dots, q_{n,k}^r, neg_1^r, \dots, neg_{r-1}^r, q_\top\} \cup St^{r-1}$,
- $\Pi = \{\mathsf{yes}, \mathsf{neg}\}$, $\quad \pi^r(\mathsf{yes}) = \{q_\top\}$, $\quad \pi^r(\mathsf{neg}) = \{neg_i^j \mid i, j = 1, ..., r\}$,
- $Act^r = \{1, ..., \max(k + r - 1, n), \top, \bot\}$,
- $d^r(\mathbf{v}, q_0^r) = d^r(\mathbf{v}, neg_i^r) = d^r(\mathbf{v}, q_\top) = \{1\}$, $\quad d^r(\mathbf{v}, q_i^r) = \{1, ..., k + r - 1\}$,
  $d^r(\mathbf{v}, q_{i,j}^r) = \{\top, \bot\}$,
  $d^r(\mathbf{r}, q) = \{1, ..., n\}$ for $q = q_0^r$ and $\{1\}$ for the other $q \in St^r$.
  For $q \in St^{r-1}$, we simply include the function from M_{r-1}: $d^r(a, q) = d^{r-1}(a, q)$;
- $o^r(q_0^r, 1, i) = q_i^r$, $\quad o^r(q_i^r, j, 1) = q_{i,j}^r$ for $j \leq k$,
  $o^r(q_i^r, k+j, 1) = q_0^{r-1}$ if $s_{i,k+j}^r = +$, and $o^r(q_i^r, k+j, 1) = neg_j^r$ if $s_{i,k+j}^r = -$,
  $o^r(neg_j^r, 1, 1) = q_0^{r-1}$,
  $o^r(q_{i,j}^r, \top, 1) = q_\top$ if $s_{i,j}^r = +$, and $q_{i,j}^r$ otherwise,
  $o^r(q_{i,j}^r, \bot, 1) = q_\top$ if $s_{i,j}^r = -$, and $q_{i,j}^r$ otherwise.
  For $q \in St^{r-1}$, we include the transitions from M_{r-1}: $o^r(q, \alpha) = o^{r-1}(q, \alpha)$;
- $q_0^r \sim_{\mathbf{v}} q$ iff $q = q_0^r$, $\quad q_i^r \sim_{\mathbf{v}} q$ iff $q = q_i^r$, $\quad q_{i,j}^r \sim_{\mathbf{v}} q$ iff $q = q_{i',j}^r$.
  For $q, q' \in St^{r-1}$, we include the tuples from M_{r-1}: $q \sim_{\mathbf{v}}^r q'$ iff $q \sim_{\mathbf{v}}^{r-1} q'$.

As an example, model M_3 for $\varphi_3 \equiv (x_3 \vee \neg z_2) \wedge (\neg x_3 \vee \neg z_1)$, $\varphi_2 \equiv z_1 \wedge \neg z_1$, $\varphi_1 \equiv (x_1 \vee x_2) \wedge \neg x_1$, is presented in Figure 3.

Theorem 4 *Let*

$$\begin{aligned}\Phi_1 &\equiv \langle\langle \mathbf{v} \rangle\rangle_{ir} (\neg \mathsf{neg})\, \mathcal{U}\, \mathsf{yes},\\ \Phi_i &\equiv \langle\langle \mathbf{v} \rangle\rangle_{ir} (\neg \mathsf{neg})\, \mathcal{U}\, (\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A} \bigcirc \neg \Phi_{i-1})).\end{aligned}$$

Now, for all r: z_r is true iff $M_r, q_0^r \models \Phi_r$.

Before we prove the theorem, we state an important lemma.

Lemma 5 *For $i \geq r$: $M_r, q_0^r \models \Phi_i$ iff $M_r, q_0^r \models \Phi_{i+1}$.*

Proof (induction on r).

1. For $r = 1$: $M_1, q_0^1 \models \Phi_i$ iff $M_1, q_0^1 \models \langle\langle \mathbf{v} \rangle\rangle_{ir} \Diamond \mathsf{yes}$ iff $M_1, q_0^1 \models \Phi_{i+1}$, because M_1 does not include states that satisfy neg.

2. For $r > 1$: $M_r, q_0^r \models \Phi_{i+1} \equiv \langle\langle \mathbf{v} \rangle\rangle_{ir} (\neg \mathsf{neg})\, \mathcal{U}\, (\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A} \bigcirc \neg \Phi_i))$ iff $\exists s_{\mathbf{v}} \forall \lambda \in out(q_0^r, s_{\mathbf{v}}) \exists u \forall w \leq u. \big((M_r, \lambda[u] \models \mathsf{yes}$ *or* $M_r, \lambda[u] \models \mathsf{neg} \wedge \mathsf{A} \bigcirc \neg \Phi_i)$ *and* $(M_r, \lambda[w] \models \neg \mathsf{neg})\big)$. [*]
   However, each state satisfying neg has exactly one outgoing transition, so $M_r, \lambda[u] \models \mathsf{neg} \wedge \mathsf{A} \bigcirc \neg \Phi_i$ is equivalent to $M_r, \lambda[u] \models \mathsf{neg}$ *and* $M_r, \lambda[u+1] \models \neg \Phi_i$.

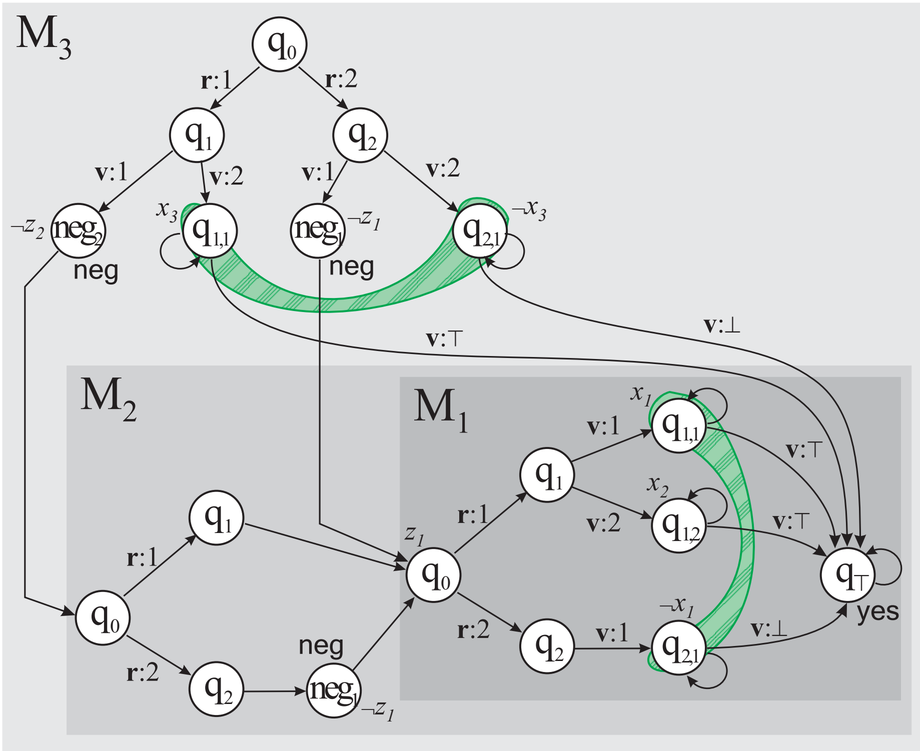

Figure 3: An i-CGS for the reduction of **SNSAT**

Thus, [*] iff $\exists s_{\mathbf{v}}\forall\lambda \in out(q_0^r, s_{\mathbf{v}})\exists u\forall w \leq u.\big((M_r, \lambda[u] \models \mathsf{yes}$ *or* $M_r, \lambda[u] \models \mathsf{neg}$ *and* $M_r, \lambda[u+1] \models \neg\Phi_i)$ *and* $(M_r, \lambda[w] \models \neg\mathsf{neg})\big)$ [**].

Note that, by the construction of M_r, $\lambda[u+1]$ must refer to the initial state q_0^j of some "submodel" M_j, $j < r \leq i$. Thus, $M_r, \lambda[u+1] \models \neg\Phi_i$ iff $M_j, q_0^j \models \neg\Phi_i$ iff (by induction) $M_j, q_0^j \models \neg\Phi_{i-1}$ iff $M_j, \lambda[u+1] \models \neg\Phi_{i-1}$.

So, [**] iff $\exists s_{\mathbf{v}}\forall\lambda \in out(q_0^r, s_{\mathbf{v}})\exists u\forall w \leq u.\big((M_r, \lambda[u] \models \mathsf{yes}$ *or* $M_r, \lambda[u] \models \mathsf{neg}$ *and* $M_r, \lambda[u+1] \models \neg\Phi_{i-1})$ *and* $(M_r, \lambda[w] \models \neg\mathsf{neg})\big)$ iff $M_r, q_0^r \models \langle\langle \mathbf{v}\rangle\rangle_{ir}(\neg\mathsf{neg})\,\mathcal{U}\,(\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A}\bigcirc\neg\Phi_{i-1})) \equiv \Phi_i$.

■

Proof of Theorem 4 (induction on r).

1. For $r = 1$: we use the proof of Theorem 2.
2. For $r > 1$:

For the implication from left to right (⇒): let z_r be true: then, there is a valuation of X_r such that φ_r holds. We construct $s_{\mathbf{v}}$ as in the proof of Theorem 2. In case that some x_i^s has been "chosen" in clause C_i^r, we are done. In case that some z_j^- has been "chosen" in clause C_i^r (note: j must be smaller than i), we have (by induction) that $M_j, q_0^j \models \neg\Phi_j$. By Lemma 5, also $M_j, q_0^j \models \neg\Phi_r$, and hence $M_r, q_0^j \models \neg\Phi_r$. So we can make the same choice (i.e., z_j^-) in $s_{\mathbf{v}}$, and this will lead to state neg_j^r, in which it holds that $\mathsf{neg} \wedge \mathsf{A}\bigcirc\neg\Phi_r$.

In case that some z_j^+ has been "chosen" in clause C_i^r, we have (by induction) that $M_j, q_0^j \models \Phi_j$, and hence, by Lemma 5, $M_j, q_0^j \models \Phi_r$. That is, there is a strategy $s'_{\mathbf{v}}$ in M_j such that $(\neg\mathsf{neg})\,\mathcal{U}\,(\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A}\bigcirc\neg\Phi_{r-1}))$ holds for all paths from $out(q_0^j, s'_{\mathbf{v}})$. As the states in M_j have no epistemic links to states outside of it, we can merge $s'_{\mathbf{v}}$ into $s_{\mathbf{v}}$.

For the other direction (⇐): let $M_r, q_0^r \models \Phi_r \equiv \langle\langle\mathbf{v}\rangle\rangle_{ir}(\neg\mathsf{neg})\,\mathcal{U}\,(\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A}\bigcirc\neg\Phi_{r-1}))$. We take the strategy $s_{\mathbf{v}}$ that enforces $(\neg\mathsf{neg})\,\mathcal{U}\,(\mathsf{yes} \vee (\mathsf{neg} \wedge \mathsf{A}\bigcirc\neg\Phi_{r-1}))$. We first consider the clause C_i^r for which a "propositional" state is chosen by $s_{\mathbf{v}}$. The strategy defines a uniform valuation for X_r that satisfies these clauses. For the other clauses, we have two possibilities:

- $s_{\mathbf{v}}$ chooses q_0^j in the state corresponding to C_i^r. Neither yes nor neg have been encountered on this path yet, so we can take $s_{\mathbf{v}}$ to demonstrate that $M_r, q_0^j \models \Phi_r$, and hence $M_j, q_0^j \models \Phi_r$. By Lemma 5, also $M_j, q_0^j \models \Phi_j$. By induction, z_j must be true, and hence clause C_i^r is satisfied.
- $s_{\mathbf{v}}$ chooses neg_j^r in the state corresponding to C_i^r. Then, it must be that $M_r, neg_j^r \models \mathsf{A}\bigcirc\neg\Phi_{r-1}$, and hence $M_j, q_0^j \models \neg\Phi_{r-1}$. By Lemma 5, also $M_j, q_0^j \models \neg\Phi_j$. By induction, z_j must be false, and hence clause C_i^r (containing $\neg z_j$) is also satisfied.

■

Thus, in order to determine the value of z_p, it is sufficient to model check Φ_p in M_p, q_0^p. Note that model M_p consists of $O(|\varphi|p)$ states and $O(|\varphi|p)$ transitions, where $|\varphi|$ is the maximal length of formulae $\varphi_1, ..., \varphi_p$. Moreover, the length of formula Φ_p is linear in p, and the construction of M_p and Φ_p can be also done in time $O(|\varphi|p)$ and $O(p)$, respectively. In consequence, we obtain a polynomial reduction of $\mathbf{SNSAT}$ to ATL$_{ir}$ model checking.

Theorem 6 *Model checking* ATL$_{ir}$ *is* $\mathbf{\Delta_2^P}$*-complete with respect to the number of transitions in the model, and the length of the formula. The problem is* $\mathbf{\Delta_2^P}$*-complete even for turn-based models with at most two agents.*

4 Conclusions

In this paper we proved that model checking of ATL$_{ir}$ formulae is $\mathbf{\Delta_2^P}$-hard, and therefore $\mathbf{\Delta_2^P}$-complete. Thus, we close an existing gap (between $\mathbf{NP}$-hardness and $\mathbf{\Delta_2^P}$-easiness) in the work of Schobbens [16], and at the same time correct our own claim from [10]. The gap between $\mathbf{NP}$ and $\mathbf{\Delta_2^P}$ is not terribly large, so the result might seem a minor one – although, technically, it was not that trivial to prove it. On the other hand, its importance goes well beyond model checking of ATL$_{ir}$. In fact, Theorem 6 yields immediate corollaries with $\mathbf{\Delta_2^P}$-completeness of other logics like ATOL, "Feasible ATEL", CSL etc., and $\mathbf{\Delta_2^P}$-hardness of ETSL.

We thank Nils Bulling for checking our proofs.

References

[1] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 100–109. IEEE Computer Society Press, 1997.

[2] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. *Journal of the ACM*, 49:672–713, 2002.

[3] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proceedings of Logics of Programs Workshop*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, 1981.

[4] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 995–1072. Elsevier Science Publishers, 1990.

[5] V. Goranko. Coalition games and alternating temporal logics. In J. van Benthem, editor, *Proceedings of TARK VIII*, pages 259–272. Morgan Kaufmann, 2001.

[6] V. Goranko and W. Jamroga. Comparing semantics of logics for multi-agent systems. *Synthese*, 139(2):241–280, 2004.

[7] A. Herzig and N. Troquard. Knowing how to play: Uniform choices in logics of agency. In *Proceedings of AAMAS'06*, 2006.

[8] W. Jamroga. Some remarks on alternating temporal epistemic logic. In B. Dunin-Keplicz and R. Verbrugge, editors, *Proceedings of Formal Approaches to Multi-Agent Systems (FAMAS 2003)*, pages 133–140, 2003.

[9] W. Jamroga and J. Dix. Do agents make model checking explode (computationally)? In M. Pěchouček, P. Petta, and L.Z. Varga, editors, *Proceedings of CEEMAS 2005*, volume 3690 of *Lecture Notes in Computer Science*, pages 398–407. Springer Verlag, 2005.

[10] W. Jamroga and J. Dix. Model checking strategic abilities of agents under incomplete information. In M. Coppo, E. Lodi, and G.M. Pinna, editors, *Proceedings of ICTCS 2005*, volume 3701 of *Lecture Notes in Computer Science*, pages 295–308. Springer Verlag, 2005.

[11] W. Jamroga and W. van der Hoek. Agents that know how to play. *Fundamenta Informaticae*, 63(2–3):185–219, 2004.

[12] W. Jamroga and Thomas Ågotnes. Constructive knowledge: What agents can achieve under incomplete information. Technical Report IfI-05-10, Clausthal University of Technology, 2005.

[13] Wojciech Jamroga and Thomas Ågotnes. What agents can achieve under incomplete information. In *Proceedings of AAMAS'06*, 2006.

[14] G. Jonker. Feasible strategies in Alternating-time Temporal Epistemic Logic. Master thesis, University of Utrecht, 2003.

[15] F. Laroussinie, N. Markey, and G. Oreiby. Expressiveness and complexity of ATL. Technical Report LSV-06-03, CNRS & ENS Cachan, France, 2006.

[16] P. Y. Schobbens. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2), 2004.

[17] W. van der Hoek, A. Lomuscio, and M. Wooldridge. On the complexity of practical ATL model checking. In P. Stone and G. Weiss, editors, *Proceedings of AAMAS'06*, pages 201–208, 2006.

[18] W. van der Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In C. Castelfranchi and W.L. Johnson, editors, *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-02)*, pages 1167–1174. ACM Press, New York, 2002.

[19] W. van der Hoek and M. Wooldridge. Cooperation, knowledge and time: Alternating-time Temporal Epistemic Logic and its applications. *Studia Logica*, 75(1):125–157, 2003.

[20] S. van Otterloo and G. Jonker. On Epistemic Temporal Strategic Logic. *Electronic Notes in Theoretical Computer Science*, XX:35–45, 2004. Proceedings of LCMAS'04.