Artificial Intelligence 277 (2019) 103172

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Approximate verification of strategic abilities under imperfect information

Wojciech Jamroga [a,*], Michał Knapik [a], Damian Kurpiewski [a], Łukasz Mikulski [b]

[a] Institute of Computer Science, Polish Academy of Sciences, Poland
[b] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Poland

ARTICLE INFO

Article history:
Received 10 January 2018
Received in revised form 4 September 2019
Accepted 5 September 2019
Available online 11 September 2019

Keywords:
Strategic ability
Alternating-time temporal logic
Model checking
Imperfect information
Alternating μ-calculus
Approximate verification

ABSTRACT

Model checking of strategic ability under imperfect information is known to be hard. The complexity results range from **NP**-completeness to undecidability, depending on the precise setup of the problem. No less importantly, the usual fixpoint equivalences do not hold for imperfect information strategies, which seriously hampers incremental synthesis of winning strategies. In this paper, we propose translations of $\mathbf{ATL}_{\mathrm{ir}}$ formulae that provide lower and upper bounds for their truth values, and are cheaper to verify than the original specifications. Most interestingly, the lower approximation is provided by a fixpoint expression that uses a nonstandard variant of the next-step ability operator. We show the correctness of the translations, establish their computational complexity, and validate the approach by experiments with several benchmarks, including a scalable scenario of Bridge play. We also demonstrate that the approximations leave much room for optimizations; in particular, optimizing the data structures can produce a significant speedup. Finally, we show that our fixpoint approximations of $\mathbf{ATL}_{\mathrm{ir}}$ formulae can be combined with under- and overapproximations of models in the vein of *may/must abstractions*, providing very promising experimental results.

1. Introduction

Multi-agent systems describe interactions of multiple *autonomous agents* capable of making rational choices. More and more practical problems are being modeled and solved under paradigms related to multi-agent systems. Example applications include space mission planning and air control [26,37], defense and security [35,67], logistics and production planning [38,41], and many others (cf. [59] for a survey). Some model checking tools have been created or extended to accept models of multi-agent systems as input, most notably *Mocha* [3,4], *MCK* [34,70], and *MCMAS* [53,54]. Some tools aim at verification of programs specified in multi-agent programming languages (e.g., AgentSpeak in Jason [14]).

Many relevant properties of multi-agent systems refer to *strategic abilities* of agents and their groups. In particular, most functionality requirements can be specified as the ability of the authorized users to achieve their legitimate goals, or to complete their tasks. At the same time, many security properties can be phrased in terms of the inability of the unauthorized users to obtain their goals. Properties of this kind can be conveniently specified in *modal logics of strategic ability*, of which alternating-time temporal logic (**ATL**) [5,6] is probably the most popular. In its basic version, the logic allows to specify

* Corresponding author.
E-mail addresses: w.jamroga@ipipan.waw.pl (W. Jamroga), michal.knapik@ipipan.waw.pl (M. Knapik), damian.kurpiewski@ipipan.waw.pl (D. Kurpiewski), lukasz.mikulski@mat.umk.pl (Ł. Mikulski).

https://doi.org/10.1016/j.artint.2019.103172
0004-3702/

strategic properties of agents and their coalitions under the assumption of perfect information about the current state of affairs. That is, every agent is able to recognize the global state of the world in its entirety. As the assumption is rather unrealistic, there is a growing number of works that study the syntactic and semantic variants of **ATL** for agents with imperfect information, see [2] for an overview. The contributions are mainly theoretical, and include results concerning the conceptual soundness of a given semantics [1,2,28,39,43,50,66], meta-logical properties [19,40], and the complexity of model checking [17,31,40,45,46,66,68]. However, there is relatively little research on practical algorithms for reasoning and/or verification in scenarios where agents have a limited view of the world.

This is somewhat easy to understand, since model checking of **ATL** variants with imperfect information has been proved $\mathbf{\Delta_2^P}$- to **PSPACE**-complete for agents playing memoryless (a.k.a. positional) strategies [17,46,66] and **EXPTIME**-complete to undecidable for agents with perfect recall of the past [31,40]. This concurs with the results for solving imperfect information games and synthesis of winning strategies, which are also known to be hard [24,32,60]. Moreover, the imperfect information semantics of **ATL** does not admit simple fixpoint characterizations based on standard short-term ability operators [18,29]. Clearly, that makes incremental synthesis of strategies impossible, or at least difficult to achieve. Some early attempts at verification of **ATL** with imperfect information made their way into the MCMAS model-checker [52,53,55,63], but the issue was never at the heart of the tool. More dedicated attempts began to emerge only recently [21–23,42,61]. Up until now, experimental results confirm that the initial intuition was right: model checking of strategic modalities for imperfect information is hard, and dealing with it requires innovative algorithms and verification techniques.

One idea that has not been properly explored is that of alternating-time epistemic mu-calculus (**AEμC**) [18]. Verification of **AEμC** is between **P** and $\mathbf{\Delta_2^P}$ for its fragment with no alternation of fixpoint operators, the complexity being relative to the size of the largest epistemic neighborhood in the model, i.e., the largest cluster of states that provide the same observations to the players. For **ATL** with imperfect information, model checking is at least $\mathbf{\Delta_2^P}$-complete with respect to the size of the *whole* model. Moreover, for coalitions of up to 2 agents, model checking of **AEμC** is in **P**. Thus, using **AEμC** specifications instead of **ATL** formulae can make model checking significantly cheaper. Unfortunately, the usual fixpoint equivalences do *not* hold for **ATL** with imperfect information [19]. In fact, as we already mentioned, no truth-preserving translation to simple fixpoint formulae built on standard modal, epistemic, and short-term strategic operators can be constructed at all, neither for memoryless agents [18], nor for agents with perfect recall [29,30].

In this paper, we propose that in some instances, instead of the exact model checking, it suffices to provide a lower and an upper bound for the output. In particular, given a formula φ, we want to construct two translations $tr_L(\varphi)$ and $tr_U(\varphi)$ such that $tr_L(\varphi) \Rightarrow \varphi \Rightarrow tr_U(\varphi)$. If $tr_L(\varphi)$ is verified as true, then the original formula φ must also hold in the given model. Conversely, if $tr_U(\varphi)$ evaluates to false, then φ must also be false. The intuition for the upper bound is straightforward: instead of checking existence of an imperfect information strategy, we can look for a perfect information strategy that obtains the same goal. If the latter is false, the former must be false too. Finding a reasonable lower bound is nontrivial, but we construct one by means of a fixpoint expression in alternating epistemic mu-calculus. We begin by showing that the straightforward fixpoint translation does not work. Then, we propose how it can be modified to obtain guaranteed lower bounds. To this end, we alter the next-step operator in such a way that traversing the appropriate epistemic neighborhood is seen as an atomic activity. We show the correctness of the translations, establish their computational complexity, and validate the approach by experiments with some scalable scenarios.

The idea of approximate verification based on fixpoint translations does not rely in itself on any kind of optimization in the model checking algorithm. In fact, our first set of experiments is based on a completely straightforward implementation of the method. As it turns out, there is plenty of room for improvement, e.g., by optimizing operations on data structures. We show this by using a more efficient representation of state spaces, based on disjoint sets, and redoing some of the experiments. Finally, we observe that our proposal is similar to the idea of may/must abstraction [8,36,51], except that in our case the approximations are obtained by transforming formulae rather than models. We show that both kinds of approximation (of formulae and models) can be smoothly combined, yielding a general framework for model checking via computing under- and overapproximations.

The structure of the paper is as follows. We begin by introducing the relevant logics and their models in Section 2. Then, in Section 3, we propose and study fixpoint translations that produce correct under- and overapproximations of **ATL** formulae. Section 4 contains an experimental evaluation of the approximate algorithms for several benchmarks: a simple voting scenario and two variants of card play in the game of Bridge. In Section 5, we show that a simple optimization of data structures can lead to dramatically improved performance of the algorithm, in terms of both running time and memory consumption. Finally, we propose and study the general framework for approximate model checking in Section 6, define a may/must abstraction for **ATL** with imperfect information over explicit models, and evaluate our algorithms on abstractions of the Bridge play models. We conclude in Section 7.

Previous version of the article. Some of the ideas and results discussed here have been already presented in a preliminary form in the conference paper [47]. This article expands the conference version with revised proofs and new, more systematic experiments. It also introduces the optimizations based on disjoint sets, together with an experimental evaluation of the idea. Finally, we propose how fixpoint approximation can be combined with a variant of may/must abstraction for concurrent games, and study the performance and accurateness of the resulting algorithms.

2. Verifying strategic ability

In this section we provide an overview of the relevant variants of **ATL**, and the corresponding complexity results for model checking.

The central operator of **ATL** is $\langle\langle A\rangle\rangle\gamma$, expressing that coalition A has a strategy to ensure that the temporal property γ will hold. Many semantic variants of **ATL** have been proposed over the last 15 years, differing vastly in their assumptions about the agents' memory and observational capabilities, but also in the very concept of what *ability* means. This corresponds closely to the rich discourse on the topic in modern philosophy and AI (cf. [11,56–58,65], to name just a few relevant positions). Here, we focus on Schobbens' $\mathbf{ATL}_{\mathrm{ir}}$ as "the" logic of strategic ability under imperfect information. That is, we address the existence of *memoryless* conditional plans, and assume that a plan is successful if it achieves its goals from all the states that the coalition considers possible in the current state (*subjective ability*).[1]

The former choice is driven mainly by the difficulty of verification: model checking strategic ability with imperfect information and perfect recall ranges from **EXPTIME**-complete to undecidable, whereas for memoryless strategies it is "only" between **NP** and $\mathbf{\Delta_2^P}$.[2] It seems prudent to attack the simpler problem first, and then possibly move on to the harder one. The latter choice is more a matter of focus: both the "subjective" and the "objective" variants of ability are meaningful, but most authors seem more interested in the subjective approach, as it formalizes the notion of "knowing how to play." We believe that the techniques, developed in this paper, can be adapted (with reasonable effort) to fit the objective semantics.

We refer the interested reader to [44] for more details and an extensive discussion.

2.1. Models, strategies, outcomes

Models. We interpret **ATL** specifications over a variant of transition systems where transitions are labeled by combinations of actions, one per agent. Moreover, epistemic relations are used to indicate states that look the same to a given agent. Formally, an *imperfect information concurrent game structure* or *iCGS* [6,66,69] is given by $M = \langle \mathbb{A}\mathrm{gt}, St, Props, V, Act, d, o, \{\sim_a | a \in \mathbb{A}\mathrm{gt}\}\rangle$ which includes a nonempty finite set of all agents $\mathbb{A}\mathrm{gt} = \{1, \ldots, k\}$, a nonempty set of states St, a set of atomic propositions $Props$ and their valuation $V : Props \to 2^{St}$, and a nonempty finite set of (atomic) actions Act. The protocol function $d\colon \mathbb{A}\mathrm{gt} \times St \to 2^{Act} \setminus \{\emptyset\}$ defines nonempty sets of actions available to agents at each state; we will write $d_a(q)$ instead of $d(a, q)$, and define $d_A(q) = \prod_{a\in A} d_a(q)$ for each $A \subseteq \mathbb{A}\mathrm{gt}, q \in St$. Furthermore, o is a (deterministic) transition function that assigns the outcome state $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to each state q and tuple of actions $\langle\alpha_1, \ldots, \alpha_k\rangle$ such that $\alpha_i \in d(i, q)$ for $i = 1, \ldots, k$.

Every $\sim_a \subseteq St \times St$ is an epistemic equivalence relation with the intended meaning that, whenever $q \sim_a q'$, the states q and q' are indistinguishable to agent a. The iCGS is assumed to be *uniform*, in the sense that $q \sim_a q'$ implies $d_a(q) = d_a(q')$, i.e., the same choices are available in indistinguishable states. Note that perfect information can be modeled by assuming each $\sim_a$ to be the identity relation.

Example 1. Consider a very simple voting scenario with two agents: the voter v and the coercer c. The voter casts a vote for a selected candidate $i \in \{1, \ldots, n\}$ (action $vote_i$). Upon exit from the polling station, the voter can hand in a proof of how she voted to the coercer (action $give$) or refuse to hand in the proof (action ng). The proof may be a certified receipt from the election authorities, a picture of the ballot taken with a smartphone, etc.: anything that the coercer will consider believable. After that, the coercer can either punish the voter (action pun) or not punish (action np).

The iCGS M_{vote} modeling the scenario for $n = 2$ is shown in Fig. 1. Proposition vote_i labels states where the voter has already voted for candidate i. Proposition pun indicates states where the voter has been punished. The indistinguishability relation for the coercer is depicted by dotted lines.

Strategies. A *strategy* of agent $a \in \mathbb{A}\mathrm{gt}$ is a conditional plan that specifies what a is going to do in every possible situation. Formally, a *perfect information memoryless strategy* for a can be represented by a function $s_a\colon St \to Act$ satisfying $s_a(q) \in d_a(q)$ for each $q \in St$. An *imperfect information memoryless strategy* additionally satisfies $s_a(q) = s_a(q')$ whenever $q \sim_a q'$. Following [66], we refer to the former as Ir-*strategies*, and to the latter as ir-*strategies*.

A *collective x-strategy* s_A, for coalition $A \subseteq \mathbb{A}\mathrm{gt}$ and strategy type $x \in \{\mathrm{Ir}, \mathrm{ir}\}$, is a tuple of individual x-strategies, one per agent in A. The set of all such strategies is denoted by Σ_A^x. By $s_A|_a$ we denote the strategy of agent $a \in A$ selected from s_A.

Given two partial functions $f, f'\colon X \rightharpoonup Y$, we say that f' *extends* f (denoted $f \subseteq f'$) if, whenever $f(x)$ is defined, we have $f(x) = f'(x)$. A partial function $s'_a\colon St \rightharpoonup Act$ is called *a partial x-strategy for* a if s'_a is extended by some strategy $s_a \in \Sigma_a^x$. A collective partial x-strategy s_A is a tuple of partial x-strategies, one per agent in A.

Outcome paths. A *path* $\lambda = q_0q_1q_2\ldots$ is an infinite sequence of states such that there is a transition between each q_i, q_{i+1}. We use $\lambda[i]$ to denote the ith position on path λ (starting from $i = 0$) and $\lambda[i, j]$ to denote the part of λ between positions

[1] In contrast, *objective ability* looks only at the outcome paths starting from the current global state of the system.

[2] $\mathbf{\Delta_2^P} = \mathbf{P^{NP}}$ is the class of problems solvable in polynomial time by a deterministic Turing machine making adaptive calls to an oracle for problems in **NP**.

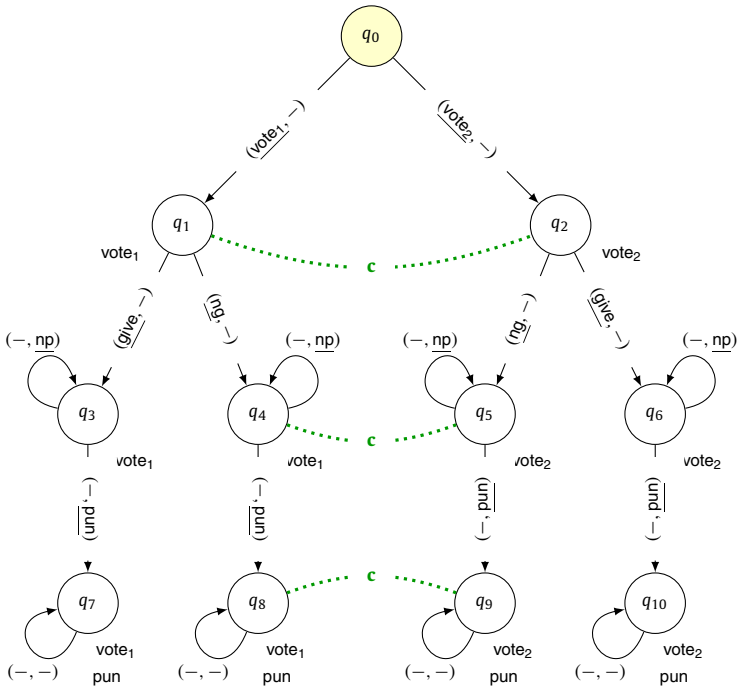

Fig. 1. A simple model of voting and coercion.

i and j. Function $out(q, s_A)$ returns the set of all paths that can result from the execution of a (complete) strategy s_A, beginning at state q. For agents not in A, path transitions can involve any action allowed by the protocol function. Formally:

$$out(q, s_A) = \{\lambda = q_0, q_1, q_2 \ldots \mid q_0 = q \text{ and for each } i = 0, 1, \ldots \text{ there exists } \langle \alpha^i_{a_1}, \ldots, \alpha^i_{a_k} \rangle \text{ such that } \alpha^i_a \in d_a(q_i) \text{ for every } a \in \mathbb{A}\text{gt, and } \alpha^i_a = s_A|_a(q_i) \text{ for every } a \in A, \text{ and } q_{i+1} = o(q_i, \alpha^i_{a_1}, \ldots, \alpha^i_{a_k})\}.$$

We will sometimes write $out^{\mathrm{Ir}}(q, s_A)$ instead of $out(q, s_A)$. Moreover, the function $out^{\mathrm{ir}}(q, s_A) = \bigcup_{a \in A} \bigcup_{q \sim_a q'} out(q', s_A)$ collects all the outcome paths that start from states that are indistinguishable from q to at least one agent in A.

2.2. Alternating-time temporal logic

Syntax. We use a variant of **ATL** that explicitly distinguishes between perfect and imperfect information abilities. Formally, the syntax is defined by the following grammar [5]:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle_x \gamma$$
$$\gamma ::= \mathbf{X}\varphi \mid \mathbf{G}\varphi \mid \varphi \,\mathbf{U}\, \varphi,$$

where $x \in \{\mathrm{Ir}, \mathrm{ir}\}$, $\mathsf{p} \in Props$ and $A \subseteq \mathbb{A}\mathrm{gt}$. Formulae γ are sometimes called *path subformulae* of **ATL**. We read $\langle\langle A \rangle\rangle_{\mathrm{ir}} \gamma$ as *"A can identify and execute a strategy that enforces γ,"* **X** as *"in the next state,"* **G** as *"now and always in the future,"* and **U** as *"until."* The perfect information modality $\langle\langle A \rangle\rangle_{\mathrm{Ir}} \gamma$ can be read as *"A might be able to bring about γ if allowed to make lucky guesses whenever uncertain."* We focus on the kind of ability expressed by $\langle\langle A \rangle\rangle_{\mathrm{ir}}$. The other strategic modality (i.e., $\langle\langle A \rangle\rangle_{\mathrm{Ir}}$) will prove useful when approximating $\langle\langle A \rangle\rangle_{\mathrm{ir}}$.

Semantics. The semantics of **ATL** can be defined as follows:

- $M, q \models \mathsf{p}$ iff $q \in V(\mathsf{p})$,
- $M, q \models \neg\varphi$ iff $M, q \not\models \varphi$,
- $M, q \models \varphi \wedge \psi$ iff $M, q \models \varphi$ and $M, q \models \psi$,
- $M, q \models \langle\langle A \rangle\rangle_x \mathbf{X}\varphi$ iff there exists $s_A \in \Sigma^x_A$ such that for all $\lambda \in out^x(q, s_A)$ we have $M, \lambda[1] \models \varphi$,
- $M, q \models \langle\langle A \rangle\rangle_x \mathbf{G}\varphi$ iff there exists $s_A \in \Sigma^x_A$ such that for all $\lambda \in out^x(q, s_A)$ and $i \in \mathbb{N}$ we have $M, \lambda[i] \models \varphi$,
- $M, q \models \langle\langle A \rangle\rangle_x \psi \,\mathbf{U}\, \varphi$ iff there exists $s_A \in \Sigma^x_A$ such that for all $\lambda \in out^x(q, s_A)$ there is $i \in \mathbb{N}$ for which $M, \lambda[i] \models \varphi$ and $M, \lambda[j] \models \psi$ for all $0 \leq j < i$.

The standard boolean operators (logical constants $\top$ and $\bot$, disjunction $\vee$, and implication $\rightarrow$) are defined as usual. We will often write $\langle A\rangle\varphi$ instead of $\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{X}\varphi$ to express one-step abilities under imperfect information. Additionally, we define *"now or sometime in the future"* as $\mathbf{F}\varphi \equiv \top\mathbf{U}\varphi$. It is easy to see that $M, q \models \langle\langle A\rangle\rangle_x\mathbf{F}\varphi$ if, and only if, there exists a collective strategy $s_A \in \Sigma_A^x$ such that, on each path $\lambda \in out^x(q, s_A)$, there is a state satisfying φ. In that case, we can also say that φ is *x-reachable from q*.

Example 2. Consider model M_{vote} from Example 1. The following formula expresses that the coercer can ensure that the voter will eventually either have voted for candidate i (presumably chosen by the coercer for the voter to vote for) or be punished: $\varphi_0 \equiv \langle\langle c\rangle\rangle_{\mathrm{ir}}\mathbf{F}\big(\neg\mathsf{vote}_\mathsf{i} \rightarrow \mathsf{pun}\big)$. We note that it holds in M_{vote}, q_0 for any $i = 1, 2$. A strategy for c that witnesses the property is $s_c(q_3) = np,\ s_c(q_4) = s_c(q_5) = s_c(q_6) = pun$ for $i = 1$, and symmetrically for $i = 2$.

Consequently, the formula $\varphi_1 \equiv \langle\langle v\rangle\rangle_{\mathrm{ir}}\mathbf{G}\big(\neg\mathsf{pun} \wedge \neg\mathsf{vote}_\mathsf{i}\big)$ saying that the voter can avoid voting for candidate i and being punished, is false in M_{vote}, q_0 for all $i = 1, 2$.

We refer to the syntactic fragment containing only $\langle\langle A\rangle\rangle_{\mathrm{ir}}$ modalities as $\mathbf{ATL}_{\mathrm{ir}}$, and to the one containing only $\langle\langle A\rangle\rangle_{\mathrm{Ir}}$ modalities as $\mathbf{ATL}_{\mathrm{Ir}}$.

Proposition 3 *([6,66,46]). Model checking* $\mathbf{ATL}_{\mathrm{Ir}}$ *is* **P***-complete and can be done in time* $O(|M| \cdot |\varphi|)$ *where* $|M|$ *is the number of transitions in the model and* $|\varphi|$ *is the length of the formula. Model checking* $\mathbf{ATL}_{\mathrm{ir}}$ *is* $\mathbf{\Delta_2^P}$*-complete with respect to* $|M|$ *and* $|\varphi|$.

2.3. Reasoning about knowledge

Having indistinguishability relations in the models, we can interpret knowledge modalities K_a in the standard way:

- $M, q \models K_a\varphi$ iff $M, q' \models \varphi$ for all q such that $q \sim_a q'$.

Intuitively, the meaning of $q \models K_a\varphi$ is that the observational capabilities of agent a allow the agent to conclude that φ holds in each state that is indistinguishable from q according to a.

The semantics of *"everybody knows" (E_A)* and *common knowledge (C_A)* are defined analogously by assuming the relation $\sim_A^E = \bigcup_{a\in A} \sim_a$ to aggregate individual uncertainty in A, and $\sim_A^C$ to be the transitive closure of $\sim_A^E$. By convention, we take $\sim_\emptyset^E$ and $\sim_\emptyset^C$ to be the identity relations. We also use $[q]_{\mathcal{R}} = \{q' \mid q\mathcal{R}q'\}$ to denote the image of q wrt relation $\mathcal{R}$.

Example 4. The following formulae hold in M_{vote}, q_0 for any $i = 1, 2$ by virtue of strategy s_c presented in Example 2:

- $\varphi_2 \equiv \langle\langle c\rangle\rangle_{\mathrm{ir}}\mathbf{F}\big((\neg K_c\mathsf{vote}_\mathsf{i}) \rightarrow \mathsf{pun}\big)$: The coercer has a strategy so that, eventually, the voter is punished unless the coercer has learnt that the voter voted as instructed;
- $\varphi_3 \equiv \langle\langle c\rangle\rangle_{\mathrm{ir}}\mathbf{G}\big((K_c\mathsf{vote}_\mathsf{i}) \rightarrow \neg\mathsf{pun}\big)$: Moreover, the coercer can guarantee that if he learns that the voter obeyed, then the voter will not be punished.

Note that the property expressed by the first formula reflects a rather strict requirement on the expectations of the coercer. Namely, in a system where the formula holds, the coercer enforces a strategy that punishes the voter at every reached state where he is *not certain* that the voter voted for the i-th candidate. As $q \models \neg K_c\mathsf{vote}_\mathsf{i} \rightarrow \mathsf{pun}$ implies $q \models \neg\mathsf{vote}_\mathsf{i} \rightarrow \mathsf{pun}$, we observe that φ_2 implies φ_0 of Example 2.

Remark 5. Note that $K_a\varphi$ is definable in $\mathbf{ATL}_{\mathrm{ir}}$ as $\langle\langle a\rangle\rangle_{\mathrm{ir}}\bot\mathbf{U}\varphi$. More generally, one can define "everybody knows" entirely in $\mathbf{ATL}_{\mathrm{ir}}$ by $E_A\varphi \equiv \langle\langle A\rangle\rangle_{\mathrm{ir}}\bot\mathbf{U}\varphi$.

Remark 6. The semantics of $\langle\langle A\rangle\rangle_{\mathrm{ir}}\gamma$, presented in Section 2.2, encodes the notion of "subjective" ability [50,66]. That is, the agents must have a successful strategy from all the states that they consider possible when the system is in state q. Then, they know that the strategy indeed obtains γ. The alternative notion of "objective" ability [19] requires the existence of a winning strategy from state q alone. We focus on the subjective interpretation, as it is more standard in game theory and **ATL**, mostly because it formalizes the notion of "knowing how to play."

Note that if $[q]_{\sim_A^E} = \{q\}$ and γ contains no nested strategic modalities, then the subjective and objective semantics of $\langle\langle A\rangle\rangle_{\mathrm{ir}}\gamma$ at q coincide. Moreover, if $\varphi, \varphi_1, \varphi_2$ contain no nested strategic modalities, then model checking $\langle\langle A\rangle\rangle_{\mathrm{ir}}\varphi_1\mathbf{U}\varphi_2$ and $\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{G}\varphi$ in M, q according to the objective semantics can be easily reduced to the subjective case by adding a spurious initial state q', with transitions to all states in $[q]_{\sim_A^E}$, controlled by a "dummy" agent outside A. We refer the interested reader to [62] for the details of the construction.

2.4. Alternating epistemic mu-calculus

It is well known that the modalities in $\mathbf{ATL}_{\mathrm{ir}}$ have simple fixpoint characterizations [6], and hence $\mathbf{ATL}_{\mathrm{ir}}$ can be embedded in a variant of μ-calculus with $\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{X}$ as the basic modality and no alternation of fixpoint operators. At the same time, the analogous variant of μ-calculus for imperfect information has incomparable expressive power to $\mathbf{ATL}_{\mathrm{ir}}$ [18]. In this section, we briefly present its syntax and semantics, and recall the relevant results.

Formally, *alternating epistemic μ-calculus (*$\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$*)* takes the next-time fragment of $\mathbf{ATL}_{\mathrm{ir}}$, possibly with epistemic modalities, and adds the least fixpoint operator μ. The greatest fixpoint operator ν is defined as dual to μ. Let $\mathcal{V}ars$ be a set of second-order variables ranging over 2^{St}. The language of $\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$ is defined by the following grammar:

$$\varphi ::= \mathsf{p} \mid Z \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle A\rangle\varphi \mid \mu Z(\varphi) \mid K_a\varphi,$$

where $\mathsf{p} \in \mathit{Props}$, $Z \in \mathcal{V}ars$, $a \in \mathbb{A}\mathrm{gt}$, $A \subseteq \mathbb{A}\mathrm{gt}$, and the formulae are Z-positive, i.e., each free occurrence of Z is in the scope of an even number of negations. We define $\nu Z(\varphi(Z)) \equiv \neg\mu Z(\neg\varphi(\neg Z))$, where $\varphi(\neg Z)$ denotes the result of substituting in φ all free occurrences of Z with $\neg Z$. A formula of $\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$ is *simple*[3] if in its negation normal form[4] it contains no occurrences of ν (resp. μ) on any syntactic path from an occurrence of μZ (resp. νZ) to a bound occurrence of Z. Disallowing the alternation of fixpoint operators simplifies the semantics of μ-calculus, and usually decreases its model checking complexity. Thus, similarly to [18], we consider here only the simple fragment of $\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$, denoted by $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$.

We evaluate the formulae of $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$ with respect to the valuations of $\mathcal{V}ars$, i.e., functions $\mathcal{V}\colon \mathcal{V}ars \to 2^{St}$. We denote the set of all the valuations of $\mathcal{V}ars$ by $\mathcal{V}als$. If $X \in \mathcal{V}ars$, $Z \subseteq St$, and $\mathcal{V} \in \mathcal{V}als$, then by $\mathcal{V}[X := Z]$ we denote the valuation of $\mathcal{V}ars$ such that $\mathcal{V}[X := Z](Y) = \mathcal{V}(Y)$ for $Y \neq X$ and $\mathcal{V}[X := Z](X) = Z$.

The denotational semantics of $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$ assigns to each formula φ the set of states $[\![\varphi]\!]^M_{\mathcal{V}}$ where φ is true under the valuation $\mathcal{V} \in \mathcal{V}als$:

- $[\![\mathsf{p}]\!]^M_{\mathcal{V}} = V(\mathsf{p})$,
- $[\![Z]\!]^M_{\mathcal{V}} = \mathcal{V}(Z)$,
- $[\![\neg\varphi]\!]^M_{\mathcal{V}} = St \setminus [\![\varphi]\!]^M_{\mathcal{V}}$,
- $[\![\varphi \wedge \psi]\!]^M_{\mathcal{V}} = [\![\varphi]\!]^M_{\mathcal{V}} \cap [\![\psi]\!]^M_{\mathcal{V}}$,
- $[\![\langle A\rangle\varphi]\!]^M_{\mathcal{V}} = \{q \in St \mid \exists s_A \in \Sigma_A\, \forall \lambda \in out^{\mathrm{ir}}_M(q, s_A)\, \lambda[1] \in [\![\varphi]\!]^M_{\mathcal{V}}\}$,
- $[\![\mu Z(\varphi)]\!]^M_{\mathcal{V}} = \bigcap\{Q \subseteq St \mid [\![\varphi]\!]^M_{\mathcal{V}[Z:=Q]} \subseteq Q\}$,
- $[\![K_a\varphi]\!]^M_{\mathcal{V}} = \{q \in St \mid \forall_{q'}(q \sim_a q' \text{ implies } q' \in [\![\varphi]\!]^M_{\mathcal{V}})\}$,

where $\varphi \in \mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$, $\mathsf{p} \in \mathit{Props}$, $Z \in \mathcal{V}ars$, $A \subseteq \mathbb{A}\mathrm{gt}$, and $a \in \mathbb{A}\mathrm{gt}$. If φ is a sentence, i.e., it contains no free variables, then its validity does not depend on the valuation $\mathcal{V}$, and we write $M, q \models \varphi$ instead of $q \in [\![\varphi]\!]^M_{\mathcal{V}}$.

Example 7. Consider the $\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$ formula $\mu Z.\big((\neg\mathsf{pun} \to \mathsf{vote}_{\mathsf{i}}) \vee \langle c\rangle Z\big)$, i.e., the "naive" fixpoint translation of the formula $\langle\langle c\rangle\rangle_{\mathrm{ir}}\mathbf{F}\big(\neg\mathsf{pun} \to \mathsf{vote}_{\mathsf{i}}\big)$ from Example 2. The fixpoint computation produces the whole set of states St. Thus, in particular, $M_{vote}, q_0 \models \mu Z.\big((\neg\mathsf{pun} \to \mathsf{vote}_{\mathsf{i}}) \vee \langle c\rangle Z\big)$.

Proposition 8 *([18]). Model checking* $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$ *with strategic modalities $\langle\langle A\rangle\rangle$ for $|A| \leq 2$ is* $\mathbf{P}$*-complete and can be done in time $O(|\sim| \cdot |\varphi|)$, where $|\sim|$ is the size of the largest equivalence class among $\sim_1, \ldots, \sim_k$, and $|\varphi|$ is the length of the formula. For $|A| \geq 3$, the problem is between* $\mathbf{NP}$ *and* $\boldsymbol{\Delta}^{\mathbf{P}}_{\mathbf{2}}$ *with respect to $|\sim|$ and $|\varphi|$.*

Thus, simple alternating epistemic μ-calculus can be an attractive alternative to $\mathbf{ATL}_{\mathrm{ir}}$ from the complexity point of view. Unfortunately, formulae of $\mathbf{ATL}_{\mathrm{ir}}$ admit no universal translations to $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$.

Proposition 9 *([18]).* $\mathbf{ATL}_{\mathrm{ir}}$ *and* $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$ *have incomparable expressive power and incomparable distinguishing power.*

The proof that $\mathbf{sAE}\boldsymbol{\mu}\mathbf{C}$ does not cover the expressive power of $\mathbf{ATL}_{\mathrm{ir}}$ uses formulae of type $\langle\langle a\rangle\rangle\mathbf{F}\mathsf{p}$, but it is easy to construct a similar argument for $\langle\langle a\rangle\rangle\mathbf{G}\mathsf{p}$. In consequence, long-term strategic modalities of $\mathbf{ATL}_{\mathrm{ir}}$ do not have simple fixpoint characterizations in terms of the next-step strategic modalities $\langle A\rangle$. We note in passing that an analogous result was proved in [29, Theorem 11] for $\mathbf{ATL}_{\mathrm{iR}}$, i.e., the variant of $\mathbf{ATL}$ with imperfect information and perfect recall strategies. Namely, it is shown that under these assumptions $\langle\langle a\rangle\rangle\mathbf{F}\mathsf{p}$ cannot be expressed as a formula of epistemic μ-calculus. Further results [29, 30] confirm that even richer variants of μ-calculus do not cover the full expressive power of $\mathbf{ATL}_{\mathrm{iR}}$.

[3] Usually, such formulae are called *alternation-free* to emphasize that they can broken down into state subformulae containing no alternation of fixpoint operators. We do not use that terminology to avoid confusion with the alternation of strategic quantifiers behind the $\langle\langle A\rangle\rangle$ and $\langle A\rangle$ operators.

[4] Negation normal form of an $\mathbf{AE}\boldsymbol{\mu}\mathbf{C}$ formula ψ is a logically equivalent formula ψ' expressed in a version of the syntax of the logic extended with $\nu Z(\varphi)$ (instead of deriving this operator from μ) and such that negations appear only in front of propositions.

3. Fixpoint approximation of strategic abilities

The main idea in this paper is that sometimes, instead of the exact model checking, it suffices to provide a lower and an upper bound for the output. In particular, given a formula φ, we want to construct two translations $tr_L(\varphi)$ and $tr_U(\varphi)$ such that $tr_L(\varphi) \Rightarrow \varphi \Rightarrow tr_U(\varphi)$. If $tr_L(\varphi)$ is verified as true, then the original formula φ must also hold in the given model. Conversely, if $tr_U(\varphi)$ evaluates to false, then φ must also be false. Clearly, such an enterprise only makes sense if the truth values for the translations are easier to compute than for the original formula. To achieve that, we will build our approximations of $\langle\langle A\rangle\rangle_{\mathrm{ir}}$ on fixpoint-definable properties. That is, we look for translations that map the formulae of $\mathbf{ATL}_{\mathrm{ir}}$ to an appropriate variant of alternating μ-calculus.

Notation. In the rest of this section, we use φ, ψ to denote arbitrary formulae of $\mathbf{ATL}_{\mathrm{ir}}$, and γ to denote arbitrary path subformulae of $\mathbf{ATL}_{\mathrm{ir}}$. Moreover, we assume that M is an iCGS, and q is a state in M (unless explicitly stated otherwise).

3.1. Lower bounds for abilities

The complexity of model checking for $\mathbf{sAE}\mu\mathbf{C}$ looks more attractive than that of $\mathbf{ATL}_{\mathrm{ir}}$ [6]. Unfortunately, the expressivity results cited in Section 2.4 imply that there is no translation to simple fixpoint formulae based on standard epistemic and short-term strategic operators, that would capture *exactly* the meaning of all $\mathbf{ATL}_{\mathrm{ir}}$ modalities. It might be possible, however, to come up with a translation tr_L that provides a *lower bound* of the actual strategic abilities, i.e., such that $M, q \models tr_L(\langle\langle A\rangle\rangle_{\mathrm{ir}}\gamma)$ implies $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\gamma$. In other words, a translation which can only reduce, but never enhance the abilities of the coalition.

We begin by investigating the "naive" fixpoint translation that mimics the one for $\mathbf{ATL}_{\mathrm{Ir}}$, and show that it does not work. Then, we propose how to alter the semantics of the next-time modality so that a general lower bound can be obtained. We focus first on reachability goals, expressed by formulae $\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$, and then extend the approach to the other modalities.

3.2. Trying it simple for reachability goals

We start with the simplest translation, analogous to that of [6]:

$$tr_{L1}(\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi) = \mu Z.(\varphi \vee \langle A\rangle Z).$$

Unfortunately, this translation provides neither a lower nor an upper bound. For the former, use model M_1 in Fig. 2A, and observe that $M_1, q_0 \models \mu Z.(\mathrm{p} \vee \langle 1\rangle Z)$, which follows from the fact that $M, q \models \varphi$ implies $M, q \models \mu Z.(\varphi \vee \langle A\rangle Z)$, for all $A \subseteq \mathbb{A}\mathrm{gt}$. On the other hand $M_1, q_0 \not\models \langle\langle 1\rangle\rangle_{\mathrm{ir}}\mathbf{F}\mathrm{p}$, as the only path starting from q_1 loops at the source, never reaching p. For the latter, take model M_2 in Fig. 2B, and observe that $M_2, q_0 \models \langle\langle 1\rangle\rangle_{\mathrm{ir}}\mathbf{F}\mathrm{p}$ via the only possible strategy (note that the sequence $q_1'q_2'$ is never visited when starting from q_0). However, $M_2, q_0 \not\models \mu Z.(\mathrm{p} \vee \langle 1\rangle Z)$, as no strategy can enforce p from $[q_1]_{\sim_1} = \{q_1, q_1'\}$.[5] As a consequence, we get the following:

Proposition 10. *$M, q \models \mu Z.(\varphi \vee \langle A\rangle Z)$ does not imply $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$. The converse implication does not hold either.*

Let us now consider a slightly stronger fixpoint specification:

$$tr_{L2}(\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi) = \mu Z.(E_A\varphi \vee \langle A\rangle Z).$$

The new translation works for the empty coalition and for single agents, but not for coalitions of multiple players:

Proposition 11. *Let $A \subseteq \mathbb{A}\mathrm{gt}$ and $q \in St$. The following holds:*

1. *$M, q \models \mu Z.(E_\emptyset\varphi \vee \langle\emptyset\rangle Z)$ iff $M, q \models \langle\langle\emptyset\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$;*
2. *If $|A| = 1$, then $M, q \models \mu Z.(E_A\varphi \vee \langle A\rangle Z)$ implies $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$, but the converse does not universally hold[6];*
3. *If $|A| > 1$, then $M, q \models \mu Z.(E_A\varphi \vee \langle A\rangle Z)$ does not imply $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$. The converse does not hold either.*

Proof. The outline of the proof is as follows. The first case follows from known results on games with perfect information. The second case is more involved: a strategy that witnesses $M, q \models \langle\langle 1\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$ is built step-by-step while computing the fixed point $\mu Z.(E_A\varphi \vee \langle 1\rangle Z)$. The key observation here is that for a single-agent these computations operate on the epistemic classes of $\sim_1$ which is an equivalence relation. As the classes are disjoint, there is no possibility of a conflict due to the lack of uniformity, once a new part of a strategy has been synthesized over a class. The procedure of incremental synthesis cannot be extended to the relation of "everybody knows" for larger coalitions, as shown in the third case.

[5] Model M_2 is taken from [18] where it was used to prove that $\mathbf{sAE}\mu\mathbf{C}$ is not expressive enough to subsume $\mathbf{ATL}_{\mathrm{ir}}$.

[6] Note that, for $A = \{a\}$, $E_A\varphi$ is equivalent to $K_a\varphi$.

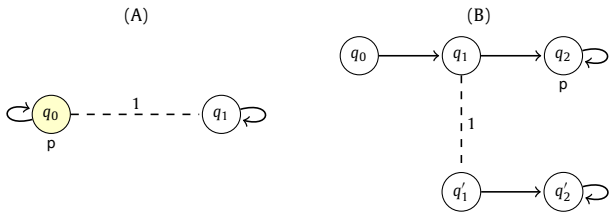

Fig. 2. Counterexamples for tr_{L1}: (A) M_1; (B) M_2.

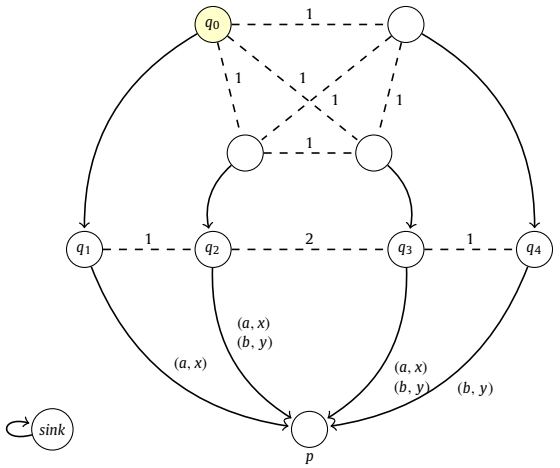

Fig. 3. M_3: a counterexample for tr_{L2}.

Case 1: follows from the fact that for the empty coalition the ir-reachability is equivalent to the Ir-reachability, which in turn has the standard fixpoint characterization in **AμC** [6].

Case 2: Let us assume that $A = \{a\}$ for some $a \in \mathbb{A}\mathrm{gt}$. We define the sequence $\{F_j\}_{j\in\mathbb{N}}$ of **sAEμC** formulae such that $F_0 = K_a\varphi$ and $F_{j+1} = F_0 \vee \langle a\rangle F_j$, for all $j \geq 0$. From Kleene fixed-point theorem we have $[\![\mu Z.(K_a\varphi \vee \langle a\rangle Z)]\!] = \bigcup_{j=0}^{\infty}[\![F_j]\!]$, and $\{[\![F_j]\!]\}_{j\in\mathbb{N}}$ is a non-decreasing monotone sequence of subsets of St. Now, we prove that for each $j \in \mathbb{N}$ there exists a partial strategy s_a^j such that $dom(s_a^j) = [\![F_j]\!]$, $\forall q \in dom(s_a^j)\ \forall \lambda \in out^{\mathrm{ir}}(q, s_a^j)\ \exists k \leq j\ \lambda[k] \models \varphi$, and $s_a^j \subseteq s_a^{j+1}$. The proof is by induction on j. We constructively build s_a^{j+1} from s_a^j for each $j \in \mathbb{N}$. The base case is trivial. For the inductive step, firstly observe that for each $j \in \mathbb{N}$ if $q \in [\![F_j]\!]$, then $[q]_{\sim_a} \subseteq [\![F_j]\!]$. As $\sim_a$ is an equivalence relation, for each $q \in [\![F_{j+1}]\!]$ either $[q]_{\sim_a} \subseteq [\![F_j]\!]$ or $[q]_{\sim_a} \subseteq [\![F_{j+1}]\!] \setminus [\![F_j]\!]$. In the first case we put $s_a^{j+1}(q) = s_a^j(q)$. In the second case, we know that there exists a strategy s_a^q such that $\forall \lambda \in out^{\mathrm{ir}}(q, s_a^q)\ \lambda[1] \in [\![F_j]\!]$. We thus put $s_a^{j+1}(q') = s_a^q(q')$ for all $q' \in [q]_{\sim_a}$, which concludes the inductive proof.

We finally define the partial strategy $s_a = \bigcup_{j\in\mathbb{N}} s_a^j$. For each $q \in St$ such that $M, q \models \mu Z.(K_a\varphi \vee \langle a\rangle Z)$, either $M, q \models \varphi$ or φ is reached along each path consistent with any extension of s_a to a full strategy.

For the converse implication, take model M_2 in Fig. 2B, and observe that $M_2, q_0 \models \langle\langle 1\rangle\rangle_{\mathrm{ir}}\mathbf{F}\mathsf{p}$ but $M_2, q_0 \not\models \mu Z.(K_1\mathsf{p} \vee \langle 1\rangle Z)$.

Case 3: Consider the iCGS M_3 presented in Fig. 3. We assume that $d_1(q) = \{a, b\}$ and $d_2(q) = \{x, y\}$, for $q \in \{q_1, q_2, q_3, q_4\}$. In the remaining states the protocols allow only one action. For clarity, we omit from the figure the transitions leaving the states q_1, q_2, q_3, and q_4, leading to state *sink*. Assume now $\varphi \equiv p$. Note that $M_3, q_0 \models \mu Z.(E_{\{1,2\}}\varphi \vee \langle\{1, 2\}\rangle Z)$ and $M_3, q_0 \not\models \langle\langle 1, 2\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$. For larger coalitions A, we extend the model with a sufficient number of spurious idle agents.

For the other direction, use the counterexample from Case 2, extended with appropriately many spurious agents. This concludes the proof of the case and of the whole proposition. □

According to Propositions 10 and 11, translation tr_{L2} provides lower bounds for $\mathbf{ATL}_{\mathrm{ir}}$ verification only in a limited number of instances. Also, the bound is rather loose, as the following example demonstrates.

Example 12. Consider the single-agent iCGS M_4 presented in Fig. 4A. The sole available strategy, in which agent 1 selects always action a, enforces eventually reaching p, i.e., $M_4, q_0 \models \langle\langle 1\rangle\rangle_{\mathrm{ir}}\mathbf{F}\mathsf{p}$. On the other hand, $M_4, q_0 \not\models \mu Z.(K_1\mathsf{p} \vee \langle 1\rangle Z)$. This is because the next-step operator in $\mathbf{ATL}_{\mathrm{ir}}$ requires reaching p simultaneously from *all* the states indistinguishable from q_0, whereas p is reached from q_0, q_1 in one and two steps, respectively.

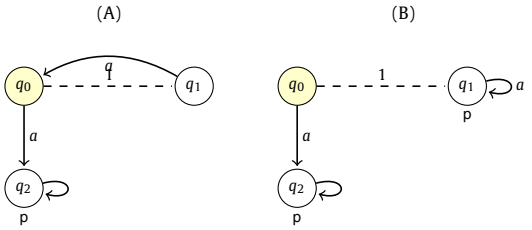

Fig. 4. Lower bounds are not tight: (A) M_4; (B) M_5.

3.3. Steadfast next step operator

To obtain a tighter lower bound, and one that works universally, we introduce a new modality. $\langle A\rangle^\bullet$ can be seen as a semantic variant of the next-step ability operator $\langle A\rangle$ where: (i) agents in A look for a short-term strategy that succeeds from the "common knowledge" neighborhood of the current state (rather than from the "everybody knows" neighborhood), and (ii) they are allowed to "steadfastly" pursue their goal in a variable number of steps within the indistinguishability class. In this section, we propose the semantics of $\langle A\rangle^\bullet$ and show how to revise the lower bound. Some additional insights are provided in Section 3.5.

We begin by defining the auxiliary function *Reach* so that $q \in Reach_M(s_A, Q, \varphi)$ collects all $q \in Q$ such that all the paths executing s_A from q eventually reach φ without leaving Q, except possibly for the last step:

$$Reach_M(s_A, Q, \varphi) = \{q \in Q \mid \forall \lambda \in out(q, s_A)\ \exists i\ .\ M, \lambda[i] \models \varphi \text{ and } \forall 0 \le j < i\ .\ \lambda[j] \in Q\}.$$

In other words, every outcome path must stay in Q until it reaches φ.

The *steadfast next-step operator* $\langle A\rangle^\bullet$ is defined as follows:

- $M, q \models \langle A\rangle^\bullet \varphi$ iff there is $s_A \in \Sigma_A^{\mathrm{ir}}$ such that $Reach_M(s_A, [q]_{\sim_A^C}, \varphi) = [q]_{\sim_A^C}$.

That is, φ must be reached this way from every state in $[q]_{\sim_A^C}$. Now we can propose our ultimate attempt at the lower bound for reachability goals:

$$tr_{L3}(\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi) = \mu Z.(E_A\varphi \vee \langle A\rangle^\bullet Z),$$

with the following result.

Proposition 13. *If $M, q \models \mu Z.(E_A\varphi \vee \langle A\rangle^\bullet Z)$, then $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$. The converse does not universally hold.*

Proof. The proof is similar to the proof of the second case of Proposition 11, namely, we synthesize a strategy witnessing $M, q \models \langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\varphi$ during iterative computation of the fixpoint $\mu Z.(E_A\varphi \vee \langle A\rangle^\bullet Z)$. The key observation here is that the operator $\langle A\rangle^\bullet Z$ selects those states for which the entire common knowledge neighborhood for group A is enforced into Z. Similarly to Proposition 11.2, we utilize the fact that common knowledge is an equivalence relation, therefore a partial strategy defined over the neighborhood during a step of the computation will not conflict with partial strategies created at any further moment.

Formally, we define a sequence $\{F_j\}_{j\in\mathbb{N}}$ of **sAEμC** formulae such that $F_0 = E_A\varphi$ and $F_{j+1} = F_0 \vee \langle A\rangle^\bullet F_j$, for all $j \ge 0$. We also use a sequence $\{H_j\}_{j\in\mathbb{N}}$ with $H_j = \langle A\rangle^\bullet F_j$. From Kleene fixed-point theorem we have $[\![\mu Z.(E_A\varphi \vee \langle A\rangle^\bullet Z)]\!] = \bigcup_{j=0}^{\infty}[\![F_j]\!] = [\![F_0]\!] \cup \bigcup_{j=0}^{\infty}[\![H_j]\!]$. Observe that, as $\sim_A^C$ is an equivalence relation, we have for each $q \in St$ and $j \in \mathbb{N}$ that if $[q]_{\sim_A^C} \cap [\![H_j]\!] \neq \emptyset$, then $[q]_{\sim_A^C} \subseteq [\![H_j]\!]$.

We prove that for each $j \in \mathbb{N}$ there exists a partial strategy s_A^j such that $dom(s_A^j) = [\![H_j]\!]$, $\forall q \in dom(s_A^j)\ \forall \lambda \in out^{\mathrm{ir}}(q, s_A^j)\ \exists k \in \mathbb{N}\ .\ \lambda[k] \models E_A\varphi$, and $s_A^j \subseteq s_A^{j+1}$. The proof is by induction on j. In the base case of $H_0 = \langle A\rangle^\bullet E_A\varphi$ observe that if $q \in [\![H_0]\!]$, then there exists a partial strategy $s_A^{0,q}$ with $dom(s_A^{0,q}) = [q]_{\sim_A^C}$ such that every $\lambda \in out^{\mathrm{ir}}(q, s_A^{0,q})$ stays in $[q]_{\sim_A^C}$ until it reaches a state where $E_A\varphi$ holds. We can now define $s_A^0 = \bigcup_{[q]_{\sim_A^C} \in St/\sim_A^C} s_A^{0,q}$ which is uniform, and reaches $E_A\varphi$ on all execution paths. For the inductive step, we divide the construction of s_A^{j+1} in two cases. Firstly, if $q \in [\![H_j]\!]$, then we put $s_A^{j+1}(q) = s_A^j(q)$. Secondly, let $q \in [\![H_{j+1}]\!] \setminus [\![H_j]\!]$. In this case there exists a partial strategy $s_A^{j+1,q}$ with $dom(s_A^{j+1,q}) = [q]_{\sim_A^C}$ such that each outcome $\lambda \in out^{\mathrm{ir}}(q, s_A^{j+1,q})$ stays in $[q]_{\sim_A^C}$ until it reaches a state q' such that either $q' \models E_A\varphi$ or $q' \in [\![H_j]\!]$. In the latter, from the inductive assumption we know that following s_A^{j+1} always leads to

reaching $E_A\varphi$ without leaving $[\![H_j]\!]$. We thus take $s_A^{j+1} = \bigcup_{[q]_{\sim_A^C} \in St/\sim_A^C} s_A^{j+1,q}$ which, again, is uniform, and reaches $E_A\varphi$ on all execution paths. This concludes the inductive part of the proof.

Finally, we build a partial strategy $s_A = \bigcup_{j\in\mathbb{N}} s_A^j$, whose any extension is such that for each $q \in St$, if $M, q \models \mu Z.(E_A\varphi \vee \langle A\rangle^\bullet Z)$, then a state in which $E_A\varphi$ holds is eventually reached along each outcome path $\lambda \in out^{\mathrm{ir}}(q, s'_A)$. This concludes the proof of the implication.

To see that the converse does not hold, consider model M_5 in Fig. 4B. We have that $M_5, q_0 \models \langle\!\langle 1\rangle\!\rangle_{\mathrm{ir}}\mathbf{F}\mathsf{p}$, but $M_5, q_0 \not\models \mu Z.(K_1\mathsf{p} \vee \langle 1\rangle^\bullet Z)$. □

Thus, tr_{L3} indeed provides a lower bound for reachability goals expressed in $\mathbf{ATL}_{\mathrm{ir}}$.

3.4. Lower bounds for "always" and "until"

So far, we have concentrated on reachability goals, expressed with the strategic operator $\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{F}$. We now extend the translation and the result in Proposition 13 to all the modalities of $\mathbf{ATL}_{\mathrm{ir}}$:

$$tr_{L3}(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi) = \nu Z.(C_A\varphi \wedge \langle A\rangle^\bullet Z),$$
$$tr_{L3}(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi) = \mu Z.\big(E_A\varphi \vee (C_A\psi \wedge \langle A\rangle^\bullet Z)\big).$$

Theorem 14.

1. *If* $M, q \models \nu Z.(C_A\varphi \wedge \langle A\rangle^\bullet Z)$, *then* $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi$;
2. *If* $M, q \models \mu Z.\big(E_A\varphi \vee (C_A\psi \wedge \langle A\rangle^\bullet Z)\big)$, *then* $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi$.

Proof. The proof exploits techniques similar to the proofs of Propositions 11 and 13. The main difference concerns the first case, where a strategy witnessing $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi$ is built *decrementally*, while computing the fixpoint $M, q \models \nu Z.(C_A\varphi \wedge \langle A\rangle^\bullet Z)$. Again, we employ the observation that the relation of common knowledge partitions the state space into disjoint equivalence classes. At the j-th step we preserve a partial uniform strategy defined over a set-theoretic sum $[\![G_j]\!]$ of such classes that enforces φ along the first i steps of each of its outcomes. At the $(j+1)$-th step we obtain $[\![G_{j+1}]\!]$ by selecting those classes that enforce $[\![G_j]\!]$ from itself.

Case 1: Let us define the sequence $\{G_j\}_{j\in\mathbb{N}}$ of formulae such that $G_0 = C_A\varphi$ and $G_{j+1} = G_0 \wedge \langle A\rangle^\bullet G_j$, for all $j \geq 0$. From Kleene fixed-point theorem, we have $[\![\nu Z.(C_A\varphi \wedge \langle A\rangle^\bullet Z)]\!] = \bigcap_{j=0}^{\infty}[\![G_j]\!]$. It suffices to prove that for each $j \in \mathbb{N}$ there exists a strategy s_A^j such that $\forall q \in [\![G_j]\!]\ \forall \lambda \in out^{\mathrm{ir}}(q, s_A^j)\ \forall 0 \leq k \leq j\ .\ \lambda[k] \models \varphi$. The proof is by induction on j, with the trivial base case. Assume that the inductive assumption holds for some $j \in \mathbb{N}$. From the definition of the steadfast next-step operator we can define for each equivalence class $[q]_{\sim_A^C} \in [\![G_{j+1}]\!]/\sim_A^C$ a partial strategy $s_A^{q,j+1}$ such that $\forall q' \in [q]_{\sim_A^C}\ \forall \lambda \in out^{\mathrm{ir}}(q, s_A^{q,j+1})\ .\ \lambda[1] \in [\![G_j]\!]$. We now construct

$$s_A^{j+1} = \bigcup_{[q]_{\sim_A^C} \in [\![G_{j+1}]\!]/\sim_A^C} s_A^{q,j+1} \cup s_A^j|_{[\![C_A\varphi]\!]\setminus[\![G_j]\!]}.$$

Intuitively, s_A^j enforces that a path leaving each $q \in [\![G_{j+1}]\!]$ stays within $[\![C_A\varphi]\!]$ for at least j steps. Moreover, $s_A^j \subseteq s_A^{j+1}$ for all j. Thus, $s_A = \bigcup_{j\in\mathbb{N}} s_A^j$ enforces that a path leaving each $q \in \bigcup_{j\in\mathbb{N}}[\![G_j]\!]$ stays within $[\![C_A\varphi]\!]$ for infinitely many steps, which concludes the proof. Note that the correctness of the construction relies the fact that $\sim_A^C$ is an equivalence relation.

Case 2: The proof is analogous to that of Proposition 13. □

Remark 15. In fact, a closer inspection of the above proof shows that a stronger result can be obtained:

- $M, q \models tr_{L3}(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi)$ implies $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}C_A\varphi$;
- $M, q \models tr_{L3}(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi)$ implies $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}(C_A\psi)\,\mathbf{U}\,(E_A\varphi)$.

3.5. Discussion & properties

Theorem 14 shows that $tr_{L3}(\varphi)$ provides a correct lower bound of the value of φ for all formulae of $\mathbf{ATL}_{\mathrm{ir}}$. In this section, we discuss the tightness of the approximation from the theoretical point of view. In particular, we argue in Section 3.5.1 that the use of a non-standard next-step ability operator is justified, as it allows to obtain strictly tighter approximations than the standard one. Moreover, we present a partial characterization of models for which the lower bound is tight by giving a necessary condition (Section 3.5.2, Proposition 18). It shows that if the lower bound verification for $\langle\!\langle A\rangle\!\rangle\gamma$ returns "true," then the agents in A must have a *recomputable* strategy s_A to enforce γ – in the sense that they will keep knowing that s_A is winning for γ at any point of executing s_A.

An empirical evaluation will be presented in Section 4.

3.5.1. Comparing tr_{L2} and tr_{L3} for Reachability Goals

Translation tr_{L3} updates tr_{L2} by replacing the standard next-step ability operator $\langle A\rangle$ with the "steadfast next-step ability" $\langle A\rangle^\bullet$. The difference between the semantics of $\langle A\rangle\varphi$ and $\langle A\rangle^\bullet\varphi$ is twofold. First, $\langle A\rangle\varphi$ looks for a winning short-term strategy in the "everybody knows" neighborhood of a given state (i.e., $[q]_{\sim_A^E}$), whereas $\langle A\rangle^\bullet\varphi$ looks at the "common knowledge" neighborhood (i.e., $[q]_{\sim_A^C}$). Secondly, $\langle A\rangle^\bullet$ allows to "zig-zag" across $[q]_{\sim_A^C}$ until a state satisfying φ is found.

Actually, the first change would suffice to provide a universally correct lower bound for $\mathbf{ATL}_{\mathrm{ir}}$. The second update makes it *more useful* in models where agents may not see the occurrence of some action, such as M_4 of Fig. 4A. To see this formally, we show that tr_{L3} provides a strictly tighter approximation than tr_{L2} on singleton coalitions:

Proposition 16. *If $M, q \models \mu Z.(K_a\varphi \vee \langle a\rangle Z)$, then $M, q \models \mu Z.(K_a\varphi \vee \langle a\rangle^\bullet Z)$. The converse does not universally hold.*

Proof. It suffices to observe that $M, q \models \langle a\rangle \mathrm{p}$ implies $M, q \models \langle a\rangle^\bullet \mathrm{p}$, for any $\mathrm{p} \in \mathit{Props}$. Note that this is true only for single-agent coalitions. For the converse, notice that in the iCGS M_4 of Fig. 4A we have $M_4, q_0 \models \mu Z.(K_1\mathrm{p} \vee \langle 1\rangle^\bullet Z)$ and $M_4, q_0 \not\models \mu Z.(K_1\mathrm{p} \vee \langle 1\rangle Z)$. □

On the other hand, if agent a always sees whenever an action occurs, then tr_{L2} and tr_{L3} coincide for a's abilities. Formally, let us call iCGS M *lockstep for a* if, whenever there is a transition from q to q' in M, we have $q \sim_a q'$. The following is straightforward.

Proposition 17. *If M is lockstep for a, then $M, q \models \langle a\rangle\varphi$ iff $M, q \models \langle a\rangle^\bullet\varphi$. In consequence, $M, q \models tr_{L2}(\langle\!\langle a\rangle\!\rangle \mathbf{F}\varphi)$ iff $M, q \models tr_{L3}(\langle\!\langle a\rangle\!\rangle \mathbf{F}\varphi)$.*

3.5.2. When is the lower bound tight?

An interesting question is: what is the subclass of iCGS's for which tr_{L3} is tight, i.e., the answer given by the approximation is exact? We address the question only partially here. In fact, we characterize a subclass of iCGS's for which tr_{L3} is certainly *not* tight, by the necessary condition below.

Let $\gamma \equiv \mathbf{G}\psi$ or $\gamma \equiv \psi_1 \mathbf{U}\, \psi_2$ for some $\psi, \psi_1, \psi_2 \in \mathbf{ATL}_{\mathrm{ir}}$. We say that strategy $s_A \in \Sigma_A^{\mathrm{ir}}$ is *winning for γ from q* if it obtains γ for all paths in $out^{\mathrm{ir}}(q, s_A)$. Moreover, for such s_A, let $RR(q, s_A, \gamma)$ be the set of *relevant reachable states of s_A in the context of γ*, defined as follows: $RR(q, s_A, \mathbf{G}\psi)$ is the set of states that occur anywhere in $out^{\mathrm{ir}}(q, s_A)$; $RR(q, s_A, \psi_1 \mathbf{U}\, \psi_2)$ is the set of states that occur anywhere in $out^{\mathrm{ir}}(q, s_A)$ before the first occurrence of ψ_2.

Proposition 18. *Let M be an iCGS, $q \in St_M$. Furthermore, suppose that $M, q \models tr_{L3}(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\gamma)$, i.e., the lower bound translation of $\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\gamma$ returns true in M, q. Then, there is a strategy $s_A \in \Sigma_A^{\mathrm{ir}}$ which is winning for γ from every $q' \in RR(q, s_A, \gamma)$.*

Proof. Straightforward from the fact that $tr_{L3}(\varphi)$ is a fixpoint formula, and model checking of $tr_{L3}(\varphi)$ produces a winning strategy for γ from q. □

Conversely, the approximation is *not* tight if there are winning strategies, but each of them reaches an intermediate state q' from which no winning follow-up strategy can be computed. This can only happen if some states in the epistemic neighborhood $[q']_{\sim_A^E}$ are not reachable by s_A. In consequence, the agents in A forget relevant information that comes solely from the fact that they are executing s_A. We will use Proposition 18 in Section 4 to show that the few benchmarks existing in the literature are not amenable to our approximations. The complete characterization of applicability for our approximation scheme is left for future work.

3.6. Upper bound

In a given model, it is always the case that $\Sigma_A^{\mathrm{ir}} \subseteq \Sigma_A^{\mathrm{Ir}}$. Thus, whatever coalition A can achieve according to the ir-semantics, they can also achieve it according to the Ir-semantics. Conversely, if the agents have no perfect information strategy to achieve γ, they cannot have an imperfect information strategy to obtain the same. We use this observation to define a simple upper bound for formulae of $\mathbf{ATL}_{\mathrm{ir}}$[7]:

Proposition 19. *Let M be an iCGS and $q \in St_M$ a state in M. Then, $M, q \models \langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\gamma$ implies $M, q \models E_A\langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}\gamma$.*

Proof. Straightforward from the semantics. □

[7] The same observation was used in [21] to obtain a preliminary pruning of the model before model checking formulae of $\mathbf{ATL}_{\mathrm{ir}}$.

3.7. Approximation semantics for $\mathbf{ATL}_{\mathrm{ir}}$

Based on Theorem 14 and Proposition 19, we propose the *lower approximation* tr_L and the *upper approximation* tr_U for all the formulae of $\mathbf{ATL}_{\mathrm{ir}}$ as follows:

$$
\begin{aligned}
tr_L(p) &= p,\\
tr_L(\neg\varphi) &= \neg tr_U(\varphi),\\
tr_L(\varphi\wedge\psi) &= tr_L(\varphi)\wedge tr_L(\psi),\\
tr_L(\langle A\rangle\varphi) &= \langle A\rangle tr_L(\varphi),\\
tr_L(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi) &= \nu Z.(C_A tr_L(\varphi)\wedge\langle A\rangle^{\bullet}Z),\\
tr_L(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi) &= \mu Z.\big(E_A tr_L(\varphi)\vee(C_A tr_L(\psi)\wedge\langle A\rangle^{\bullet}Z)\big).
\end{aligned}
$$

$$
\begin{aligned}
tr_U(p) &= p,\\
tr_U(\neg\varphi) &= \neg tr_L(\varphi),\\
tr_U(\varphi\wedge\psi) &= tr_U(\varphi)\wedge tr_U(\psi),\\
tr_U(\langle A\rangle\varphi) &= E_A\langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}\mathbf{X}tr_U(\varphi),\\
tr_U(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi) &= E_A\langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}\mathbf{G}tr_U(\varphi),\\
tr_U(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi) &= E_A\langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}tr_U(\psi)\,\mathbf{U}\,tr_U(\varphi).
\end{aligned}
$$

It should be noted that, in computing the upper approximation, every application of rules for $tr_L(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\mathbf{G}\varphi)$ and $tr_L(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi\,\mathbf{U}\,\varphi)$ selects a fresh variable to be bound by ν and μ, respectively.

Theorem 20. *For any iCGS M, state q in it, and $\mathbf{ATL}_{\mathrm{ir}}$ formula φ:*

$$M,q\models tr_L(\varphi)\ \Rightarrow\ M,q\models\varphi\ \Rightarrow\ M,q\models tr_U(\varphi).$$

Proof. Straightforward induction on the structure of φ. □

Theorem 21. *If φ includes only coalitions of size at most 1, then model checking $tr_L(\varphi)$ and $tr_U(\varphi)$ can be done in time $O(|M|\cdot|\varphi|)$. In the general case, the problem of model checking $tr_L(\varphi)$ and $tr_U(\varphi)$ is between* **NP** *and* $\mathbf{\Delta_2^P}$ *wrt* $\max_{A\in\varphi}(|\sim_A^C|)$ *and* $|\varphi|$.

Proof. Firstly, note that both translations work in linear time and produce formulae of size linear in the size of the original. The proof follows by induction on the structure of φ. The interesting cases are $tr_U(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi)$ and $tr_L(\langle\!\langle A\rangle\!\rangle_{\mathrm{ir}}\psi)$, where ψ contains no strategic modalities. For the former, recall that model checking of $\langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}\psi$ is in **P** wrt to the size of the model and the formula [6]. For the latter, it suffices to determine the model checking complexity for formulae $\langle A\rangle^{\bullet}\psi$, where ψ contains no strategic modalities. The hardness results for $\langle A\rangle^{\bullet}\psi$ carry over to model checking of arbitrary formulae. For the upper complexity bounds, observe that the model checking for arbitrary formulae can be done by iterative computation of fixpoints, and recursive model checking of subformulae (bottom-up), with polynomially many iterations and recursive calls. Thus, if $\langle A\rangle^{\bullet}\psi$ can be model-checked in **P**, the overall procedure is also in **P**. If $\langle A\rangle^{\bullet}\psi$ can be model-checked in **NP**, the overall procedure is in $\mathbf{\Delta_2^P}$.

We further divide the proof into subcases, depending on the size of the coalition.

Case $|A|=0$: Let M' be exactly as M plus an additional fresh proposition p that holds only in state q. Then, model checking $M,q\models\langle\emptyset\rangle^{\bullet}\psi$ is equivalent to model checking the **CTL** formula $\mathsf{A}(\mathsf{p}\,\mathbf{U}\,\psi)$ in M',q, which is doable in polynomial time.

Case $|A|=1$: Let $a\in\mathbb{A}\mathrm{gt}$ and M'_α be the model M with the actions of agent a fixed to α inside the epistemic neighborhood $[q]_{\sim_a}$, plus an additional fresh proposition $\mathsf{p_a}$ that holds in the states of $[q]_{\sim_a}$. Then, model checking $M,q\models\langle a\rangle^{\bullet}\psi$ reduces to checking if the **CTLK** formula $K_a\mathsf{A}(\mathsf{p_a}\,\mathbf{U}\,\psi)$ holds in any pointed model (M'_α,q), which is doable in polynomial time (the number of possible α's is bounded by $O(|M|)$, and the cost of verifying a single pointed model is polynomial).

Case $|A|\geq 2$: We adapt the proof of [18, Theorem 12]. First, we show that model checking $M,q\models\langle A\rangle^{\bullet}\psi$ is in **NP**. This can be demonstrated by the following algorithm: (1) label the states in the common knowledge neighborhood $[q]_{\sim_A^C}$ by a fresh proposition p_A^C, (2) guess a memoryless strategy for A in $[q]_{\sim_A^C}$, (2) prune M according to the strategy, obtaining model M', and (3) model-check the **CTLK** formula $M',q\models C_A\mathsf{A}(\mathsf{p}_A^C\,\mathbf{U}\,\psi)$. Clearly, since model checking **CTLK** is in **P**, the algorithm runs in nondeterministic polynomial time.

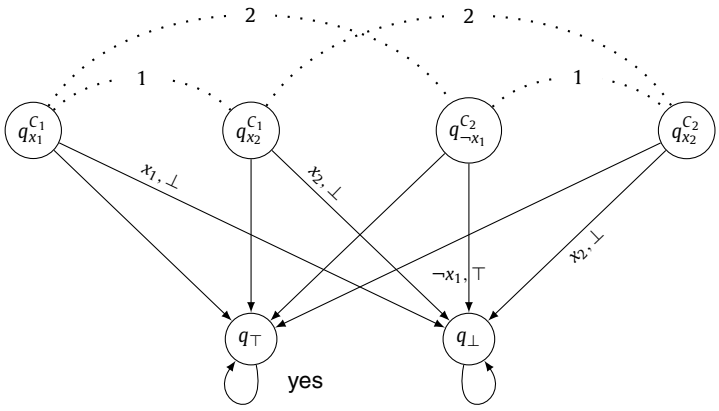

Fig. 5. Model M_Φ for $\Phi \equiv C_1 \wedge C_2, C_1 \equiv x_1 \vee x_2, C_2 \equiv \neg x_1 \vee x_2$. Only transitions leading to $q_\bot$ are labeled; the other combinations of actions lead to $q_\top$.

For the **NP**-hardness, we adapt the SAT reduction in [18, Proposition 11]. Given a Boolean formula Φ in CNF, we construct a 2-agent iCGS M_Φ as follows. Each literal l in clause ξ of Φ is associated with a state q_l^ξ. At state q_l^ξ, player 1 indicates a literal from ξ, and player 2 decides on the valuation of the underlying Boolean variable. If 1 indicated a "wrong" literal $l' \neq l$ then the system proceeds to state $q_\top$ where proposition yes holds. The same happens if 1 indicated the "right" literal (l) and 2 selected the valuation that makes l true. Otherwise the system proceeds to the "sink" state $q_\bot$. Player 1 must select literals uniformly within clauses, so $q_l^\xi \sim_1 q_{l'}^{\xi'}$ iff $\xi = \xi'$. Player 2 is to select uniform valuations of variables, i.e., $q_l^\xi \sim_2 q_{l'}^{\xi'}$ iff $var(l) = var(l')$ where $var(l)$ is the variable contained in l. An example of the construction is presented in Fig. 5.

Then, Φ is satisfiable iff $M_\Phi, q \models \langle\langle 1,2 \rangle\rangle^\bullet$yes for an arbitrary "literal" state q. □

Thus, our approximations offer computational advantage over exact $\mathbf{ATL}_{\mathrm{ir}}$ in cases when the common knowledge neighborhoods for coalition A are significantly smaller than the whole model. This happens when the members of A have similar knowledge, and especially when the coalition consists of a single agent. An interested reader may note the analogy to a number of results for solving multi-player games with imperfect information and perfect recall, which is undecidable in general but has been proved decidable for verifying abilities of individual agents [24], coalitions with exactly the same knowledge [40], hierarchical knowledge [12,13], and knowledge coming from publicly visible actions [10].

3.8. Approximation of abilities under perfect recall

In this paper, we focus on approximating abilities based on memoryless strategies. Approximations might be equally useful for $\mathbf{ATL}_{\mathrm{iR}}$ (i.e., the variant of **ATL** using uniform perfect recall strategies). However, the high intractability of $\mathbf{ATL}_{\mathrm{iR}}$ model checking suggests that a substantial extension will be needed to come up with satisfactory approximations.

Note, on the other hand, that if A have a successful memoryless strategy, then they also have a winning perfect recall strategy. In consequence, as long as our lower bounds are correct for $\mathbf{ATL}_{\mathrm{ir}}$, they are also correct for $\mathbf{ATL}_{\mathrm{iR}}$. Moreover, it is well known that the semantics of $\mathbf{ATL}_{\mathrm{IR}}$ and $\mathbf{ATL}_{\mathrm{Ir}}$ coincide [6,66]. Thus, our upper bound is also correct for $\mathbf{ATL}_{\mathrm{iR}}$.

Finally, we observe that the benchmark in Section 4.2 is a *model of perfect recall*, i.e., the states of the model explicitly encode the agents' full memory of their past observations. In consequence, the memoryless and perfect recall semantics of **ATL** coincide in the model. The experimental results suggest that, for such models, verification of perfect recall abilities can be much improved by using the approximations proposed here.

4. Experimental evaluation

Theorems 20 and 21 validate the approximation semantics theoretically. In this section, we back up the theoretical results by looking at how well the approximations work in practice. We address two issues: the *performance* and the *accuracy* of the approximations.

4.1. Existing benchmarks

The only publicly available tool that provides verification of **ATL** with imperfect information is MCMAS [55,63,52,53]. We note, however, that imperfect information strategies are not really at the heart of the model checker, the focus being on verification of **CTLK** and **ATLK** with perfect information strategies. More dedicated attempts produced so far only experimental algorithms, with preliminary performance results reported in [61,22,42,23,62,21]. Because of that, there are few benchmarks for model checking $\mathbf{ATL}_{\mathrm{ir}}$, and few experiments have actually been conducted.

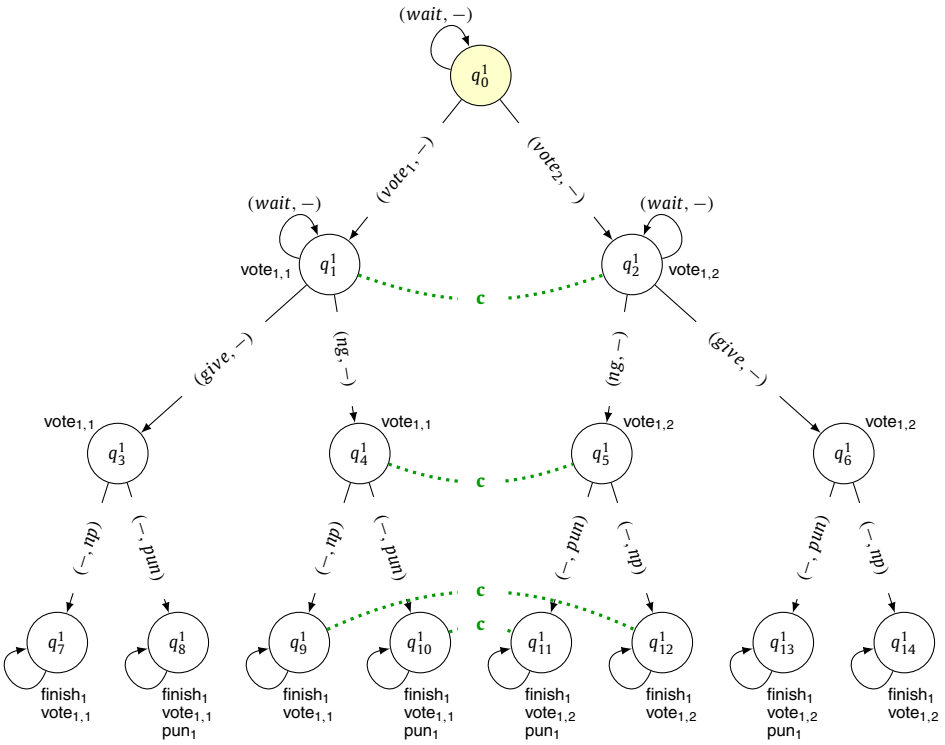

Fig. 6. A voter module for the experiments.

The classes of models typically used to estimate the performance of $\mathbf{ATL}_{\mathrm{ir}}$ model checking are **TianJi** [63,22] and **Castles** [61]. The properties to be verified are usually reachability properties, saying that Tian Ji can achieve a win over the king (in **TianJi**), or that a given coalition of workers can defeat another castle (for **Castles**). We observe that both **TianJi** and **Castles** *do not satisfy* the necessary condition in Proposition 18. This is because the states of the model do not encode some relevant information about the actions that have been already played by the coalition. Thus, even one step before winning the game, the players take into account also some (possibly losing) states that couldn't be reached by the strategy they are executing.

This means that the **sAEμC** approximations, proposed in this paper, are not useful for **TianJi** and **Castles**. It also means that the benchmarks arguably do not capture realistic scenarios. We usually do not want to assume agents to forget *their own actions* from a few steps back. In the remainder, we propose several new benchmarks that can be used to evaluate our approximation scheme.

Finally, we note that most experiments reported in the literature use very simple input formulae (no nested strategic modalities; singleton coalitions or groups of agents with identical indistinguishability relations). This is usually a matter of focus: one cannot verify everything at once, so it seems natural to start with structurally simplest instances. Here, we follow the trend.

4.2. Verifying the simple voting scenario

For the first benchmark, we adapt the simple voting scenario in Example 1. The model consists of $k+1$ agents (k voters $v_1, \ldots, v_k$, and 1 coercer c). The module of voter v_i implements the transition structure from Fig. 1, with three modifications. First, the voter can at any state execute the "idle" action *wait*. In consequence, synchronous voting as well as interleaving of votes is allowed. Secondly, in states $q_3, \ldots, q_6$, the coercer's action *np* ("no punishment") leads to an additional final state $(q'_7, \ldots, q'_{10})$, labeled accordingly. Thirdly, the old and new leaves in the structure (i.e., $q_7, \ldots, q_{10}, q'_7, \ldots, q'_{10}$) are labeled with an additional atomic proposition finish_i. The resulting model is the synchronous product of k agent modules, each being an instance of the structure presented in Fig. 6. In the synchronous product, a global state is a tuple of local states. Two global states are indistinguishable for an agent if they agree on the agent's local component, following the standard assumption that the agent can only observe her local state.

As specifications, we use formulae saying that: (i) the coercer can force the voter to vote for candidate 1 or else the voter is punished, and (ii) the voter can avoid voting for candidate 1 and being punished (cf. Example 2). Note, however, that the model used for the experiments is an unconstrained product of the voter modules. Thus, it includes also paths that were absent in the CEGS M_{vote} from Example 1 (in particular, ones where a voter executes *wait* all the time). To deal with this, we modify the specifications from Example 2 so that they discard such paths:

Table 1
Experimental results for simple voting model (formula φ_1).

k	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	result	tverif	result		
1	15	0.001	0.0001	True	0.0001	True	100%	0.006
2	225	0.015	0.001	True	0.0008	True	100%	14.79
3	3375	0.29	0.09	True	0.019	True	100%	timeout
4	50625	8.248	14.90	True	0.47	True	100%	timeout
5	662529	157.977	5626	True	9.14	True	100%	timeout
6	memout							timeout

Table 2
Experimental results for simple voting (φ_2).

k	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	result	tverif	result		
1	15	0.001	0.00005	False	0.00003	False	100%	0.005
2	225	0.015	0.0003	False	0.0002	False	100%	0.02
3	3375	0.29	0.0083	False	0.004	False	100%	0.04
4	50625	8.248	0.522	False	0.084	False	100%	0.12
5	662529	157.977	101.88	False	1.35	False	100%	0.16
6	memout							0.17

1. $\varphi_1 \equiv \langle\langle c \rangle\rangle_{\mathrm{ir}} \mathbf{G}\big((\mathsf{finish}_i \wedge \neg \mathsf{pun}_i) \rightarrow \mathsf{vote}_{i,1}\big)$ which always holds in the voting scenario,
2. $\varphi_2 \equiv \langle\langle v_i \rangle\rangle_{\mathrm{ir}} \mathbf{F}\big(\mathsf{finish}_i \wedge \neg \mathsf{pun}_i \wedge \neg \mathsf{vote}_{i,1}\big)$ which is always false.

The results of the experiments for formula φ_1 are shown in Table 1, and for φ_2 in Table 2. The columns present the following information:

- the parameter of the model (i.e., the number of voters k),
- the size of the state space (#states),
- the generation time for models (tgen),
- the time and output of verification (tverif, result) for model checking of the lower approximation $tr_L(\varphi)$,
- ...and similarly for the upper approximation $tr_U(\varphi)$;
- the percentage of cases where the bounds have matched (Match), and
- the total running time of the exact $\mathbf{ATL}_{\mathrm{ir}}$ model checking for φ (tg+tv).

The running times are given in seconds. *Timeout* indicates that the process did not terminate in 48 hours (!). *Memout* shows that the process ran out of RAM, and could not complete the verification task.

The computation of the lower and upper approximations was done with a straightforward implementation (in Python 3) of the fixpoint model checking algorithm for $\mathbf{sAE}\mu\mathbf{C}$ and $\mathbf{ATL}_{\mathrm{Ir}}$, respectively. We used the explicit representation of models, and the algorithms were not optimized in any way. The exact $\mathbf{ATL}_{\mathrm{ir}}$ model checking was done with MCMAS 1.3.0 in such a way that the underlying CEGS of the ISPL code was isomorphic to the explicit models used to compute approximations. The subjective semantics of $\mathbf{ATL}_{\mathrm{ir}}$ was obtained by using the option *-atlk 2* and setting the initial states as the starting indistinguishability class for the proponent. All the tests were conducted on a computer with an Intel Core i7-6700 CPU with dynamic clock speed of 2.60 – 3.50 GHz, 32 GB RAM, running 64bit Ubuntu 16.04 Linux.

Discussion of the results. The exact model checking with MCMAS performed well on the inputs where no winning strategy existed (formula φ_2), but was very bad at finding the existing winning strategy for formula φ_1. In that case, our approximations offered huge speedup. Moreover, the approximations actually found the winning strategy in all the tested instances, thus producing fully conclusive output. This might be partly due to the fact that the indistinguishability relations are rather sparse in the models, and in consequence the models are relatively close to perfect information.

4.3. Bridge endplay

To evaluate our scheme on a broader class of structures, we use bridge play scenarios of a type often considered in bridge handbooks and magazines. The task is to find a winning strategy for the declarer, usually depicted at the South position (**S**), in the k-endplay of the game, see Fig. 7 for an illustration. The deck consists of $4n$ cards in total (n in each suit),[8] and the initial state captures each player holding k cards in their hand, after having played $n - k$ cards. This way we

[8] In real bridge, $n = 13$, but we keep it variable to study how the model checking algorithms scale up for different sizes of the input model.

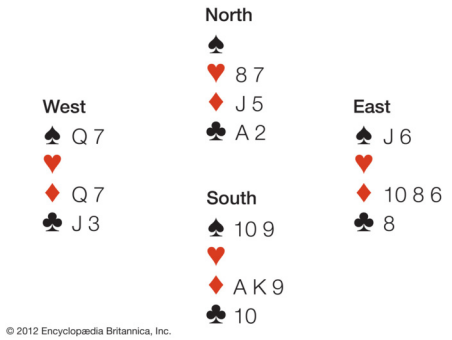

Fig. 7. Example 6-endplay in bridge.

Table 3
Experimental results: solving endplay in bridge.

(n,k)	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	%true	tverif	%false		
(1, 1)	11	0.0007	0.00007	100%	0.00004	0%	100%	0.12
(2, 2)	346	0.011	0.0008	100%	0.0003	0%	100%	2.42 h*
(3, 3)	12953	0.73	0.07	85%	0.01	15%	100%	timeout
(4, 4)	617897	35.19	348.37	90%	0.72	10%	100%	timeout
(5, 5)*	2443467	132.00	8815.73	100%	4.216	0%	100%	timeout

obtain a family of models, parameterized by the possible values of (n,k). A NoTrump contract is being played; the declarer wins if she takes more than $k/2$ tricks in the endplay.

The players' cards are played sequentially (clockwise). **S** plays first at the beginning of the game. Each next trick (i.e., the set of four played cards, one per player) is opened by the player who won the latest trick. The declarer handles her own cards and the ones of the dummy (**N**). The opponents (**W** and **E**) handle their own hands each. The cards of the dummy are visible to everybody; the other hands are only seen by their owners. Each player remembers the cards that have already been played, including the ones that were used up before the initial state of the k-endplay. That is, the local state of a player contains: the current hand of the player, the current hand of the dummy, the cards from the deck that were already used up in the previous tricks, the status of the current trick, i.e., the sequence of pairs *(player, card)* for the cards already played within the trick (alternatively, the sequence of cards already played within the trick, plus who started the trick); and the current score (which team has won how many tricks so far). The models are described more systematically in Appendix A. Moreover, our model generators, together with some example generated models, are available online at https://github.com/blackbat13/ATLFormulaChecker/tree/master/bridge_model.

Notice that, since we will only look at the strategic abilities of the declarer (**S**), the epistemic relations of the other players (**W**, **E**) are irrelevant in our experiments.

We observe the following properties of the model. First, it is turn-based (with the "idle" action *wait* that players use when another player is laying down a card). Secondly, players have imperfect information, since they cannot infer the hands of the other players. The missing information is relevant: anybody who has ever played bridge or poker knows how much the limited knowledge of the opponents' hands decreases one's chances of winning the game. Thirdly, this is a model of imperfect recall. The players do not remember in which order the cards have been played so far, and who had what cards.[9] For example, after a trick collecting ♣10, ♣J, ♣A, and ♣8 in the endplay of Fig. 7, the declarer will know that those cards have been played, but not who played which of them. Finally, the model is lockstep (everybody sees when a transition happens), and thus tr_{L2} and tr_{L3} coincide on singleton coalitions.

The results of the experiments for formula $\varphi \equiv \langle\langle \mathbf{S} \rangle\rangle_{\mathrm{ir}} \mathbf{F} \mathsf{win}$ are shown in Table 3. The columns present the following information:

- the parameters of the model (n,k),
- the size of the state space (#states),
- the generation time for models (tgen),
- the verification time (tverif) and the percentage of instances for which the output of approximate verification has been conclusive for the lower approximation tr_L (%true) and the upper approximation tr_U (%false);
- the percentage of cases where the bounds have matched (Match), and

[9] This reflects the capabilities of middle-level bridge players: they usually remember what has been played, but not in which order and by whom. Advanced players remember also who played what, and masters remember the whole history of the play.

Table 4
Experimental results for absent-minded declarer.

(n,k)	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	%true	tverif	%false		
(1, 1)	19	0.001	0.0001	100%	0.0001	0%	100%	9.68 h⋆
(2, 2)	713	0.04	0.01	100%	0.004	0%	100%	timeout
(3, 3)	52843	5.18	18.61	65%	0.58	15%	80%	timeout
(4, 4)	memout							timeout

Table 5
Absent-minded declarer, approximation tr_{L2}.

(n,k)	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	%true	tverif	%false		
(1, 1)	19	0.002	<0.0001	0%	<0.0001	0%	0%	14.93 h⋆
(2, 2)	735	0.05	0.001	0%	0.004	10%	10%	timeout
(3, 3)	60563	6.34	0.04	0%	0.58	35%	35%	timeout
(4, 4)	memout							timeout

- the total running time of the exact $\mathbf{ATL}_{\mathrm{ir}}$ model checking with MCMAS (tg+tv).

The times are given in seconds, except where indicated. The experiments were implemented and run in the same environment as for the voting scenario in Section 4.2. Again, we ran the experiments for up to 48h per instance. The results in each row are averaged over 20 randomly generated instances, except for ($\star$) where only 1 hand-crafted instance was used.[10]

Discussion of results. In the experiments, our approximations offered a dramatic speedup. The exact model checking of φ was infeasible except for the simplest models (hundreds of states), even with an optimized symbolic model checker like MCMAS. In contrast, our bounds were verified for models up to millions of states. Moreover, our approximations obtained an astonishing level of accuracy: the bounds matched in 100% of the analyzed instances, thus producing fully conclusive output.

We also emphasize that the algorithms computing our bounds have been implemented without any optimizations. This suggests that there is still much room for improving the performance. A simple optimization of data structures is discussed in Section 5, with running times improved by several orders of magnitude.

4.4. Bridge endplay by absentminded declarer

In the bridge endplay models, the players always see when a move is made. Thus, for singleton coalitions, the steadfast next-time operator $\langle a \rangle^{\bullet}$ coincides with the standard next-time abilities expressed by $\langle a \rangle$. In order to better assess the performance of our lower bound, we have considered a variant of the scenario where the declarer is absentminded and does not see the cards being laid on the table until the end of each trick. Moreover, she can play her and the dummy's cards at any moment, even in parallel with one of the opponents. This results in larger indistinguishability classes for **S**, but also in a general increase of the number of states and transitions in the model.[11]

The results of the experiments are shown in Table 4. Note that, for this class of models, the bounds do not match as tightly as before. Still, the approximation was conclusive in an overwhelming majority of instances. Moreover, it grossly outperformed the exact model checking which was (barely) possible only in the trivial case of $n = 1$.

The models are not turn-based, not lockstep, and not of perfect recall. Since they are not lockstep, approximations tr_{L2} and tr_{L3} do not have to coincide. In Table 5, we present the experimental results obtained with tr_{L2}, which show that the improved approximation tr_{L3} provides tighter lower bounds also from the practical point of view.

5. Optimizations

The experimental results presented in Section 4 were obtained with a straightforward implementation of the fixpoint algorithms that compute the lower and upper bounds. This, potentially, leaves room for various optimizations. In this section, we show that the space for optimizing the performance of the algorithms is indeed vast.

We propose three possible sources of improvement. First, we observe that the computation of the lower bound operates in fact on epistemic equivalence classes rather than individual states in the model. In consequence, it suffices to represent

[10] In case of the approximate model checking for random (5, 5) models, the program was not even able to complete model generation due to memout. For the exact model checking of random (2, 2) models, timeout was obtained in most instances.

[11] The model generators and example generated models for the absentminded declarer are also available at https://github.com/blackbat13/ATLFormulaChecker/tree/master/bridge_model.

which equivalence classes are connected by which transitions. Secondly, we propose how to improve operations on sets of states, using the technique of *disjoint sets*. Finally, we show how to further optimize the memory consumption for verification of the bridge model, based on the observation that the iCGS forms a tree where the next "stratum" depends only on the one that immediately precedes it. Note that, while the last optimization relies on the specific structure of the benchmark, the first two are completely independent from the application domain.

On top of that, we implemented the optimized algorithm in C++ which offered better control of the data management than Python.

5.1. Reduction based on epistemic equivalence classes

Consider the fixpoint algorithm computing the lower bound of $M, q \models \langle\langle A \rangle\rangle_{\mathrm{ir}} \gamma$. The crucial part of the algorithm is the computation of the pre-image of the current set of "candidate" states Q. That is, the algorithm looks for the states q' such that $M, q' \models \langle A \rangle^{\bullet} Q$. We observe now that this property is invariant with respect to the common knowledge relation $\sim_A^C$. In other words, either all the states in the common knowledge neighborhood $[q']_{\sim_A^C}$ are in the pre-image, or none of them. In consequence, when searching for one-step strategies, it suffices to consider outgoing transitions per equivalence class of $\sim_A^C$, and not for each global state separately.

Our first optimization consists of an abstraction of the transition space to transitions between equivalence classes of $\sim_A^C$, and an abstraction of the action space by available one-step strategies of the coalition. This means that we keep the information about individual states only to produce the epistemic classes and transitions between them. After the model has been generated, all the relevant information is "moved" to the representative of the class. In particular, the outgoing transitions from any state in class $[q']_{\sim_A^C}$ are moved to its representative.

The epistemic abstraction allows to obtain a significant speedup of the lower bound computation, especially for singleton coalitions. This is because we "pre-compile" the explicit model in a way that reduces the complexity of checking whether all the states in $[q']$ are in the pre-image of Q. The complexity decreases from quadratic with respect to the size of $[q']$ in the naive implementation, to constant in the optimized one.

5.2. Optimization of data structures based on disjoint sets

The fixpoint algorithms, used for computing the lower and the upper bounds in Section 4, heavily depend on the representation of *sets* of states in the model. The sets naturally arise as epistemic equivalence classes and their unions, cf. the epistemic abstraction of transitions proposed in Section 5.1. To reduce the memory needed to store such sets, and speed up the operations of adding elements to a set and merging two sets, we use the technique of *disjoint-set data structures*, also known as *merge-find sets* [33]. In the approach, each set of states is stored as an arbitrary tree (see Fig. 8 for an illustration). The root of the tree serves as the representative.

The three key functions, operating on this data structure, are: *make_set* (to create singleton sets at the beginning), *find* (to find the representative of the set where a given element belongs), and *union* (to merge two sets). Every member of the set is equipped with a single pointer indicating its parent and, by transitivity, pointing to the representative of the set. The root is used not only for identifying the set (by the *find* operation), but also to store all the relevant data related to the set of states.

This is especially important for transitions between states. We already mentioned that it is sufficient to look at the sets of all outgoing transitions for each epistemic class as a whole. Similarly, for incoming transitions, it matters only whether they point to the union of epistemic classes whose pre-image is currently being computed. Thanks to that, we can take an advantage of the disjoint-set structure and the fact that every set is represented by one root. The idea is to store information about all the transitions from (and into) the set only in its root. To further optimize this procedure, we reassign a transition from an arbitrary state only when the transition is used. Thus, the reassignment is only made after the algorithm calls function *find* to ask for the representative (root) of the endpoint of the transition. Thanks to the postponed update of transitions, an efficient implementation of the disjoint-set data structure reduces the complexity of merging two sets to *amortized near-constant-time*, i.e., the inverse of the Ackermann function.

Further details:

- Function *make_set(s)* is used during the model generation phase. It takes node s, and adds two extra fields: one for storing the pointer to the parent, the other to store an approximate height of the tree rooted in this node (initially 1).
- Function *find(s)* is implemented together with path compression. That is, while searching for s, the algorithm updates the parent pointers of every node found on the path from s to the root r; the new value of each pointer being r. Note that *find(s)* returns the representative r, which can be used not only to check whether two elements are in the same set, but also to update the connection if it initially pointed to an element different from r.
- The implementation of function *union*(s_1, s_2) uses the approximate heights of the trees rooted in s_1 and s_2. As the new representative, the node with the larger value is chosen. In case of equal values, an arbitrary node is selected, and the algorithm adds 1 to its approximate height. In order to speed up further computations, we also reassign the starting points of all the outgoing transitions to the new root. For incoming transitions, we use *lazy update*. That is, we postpone

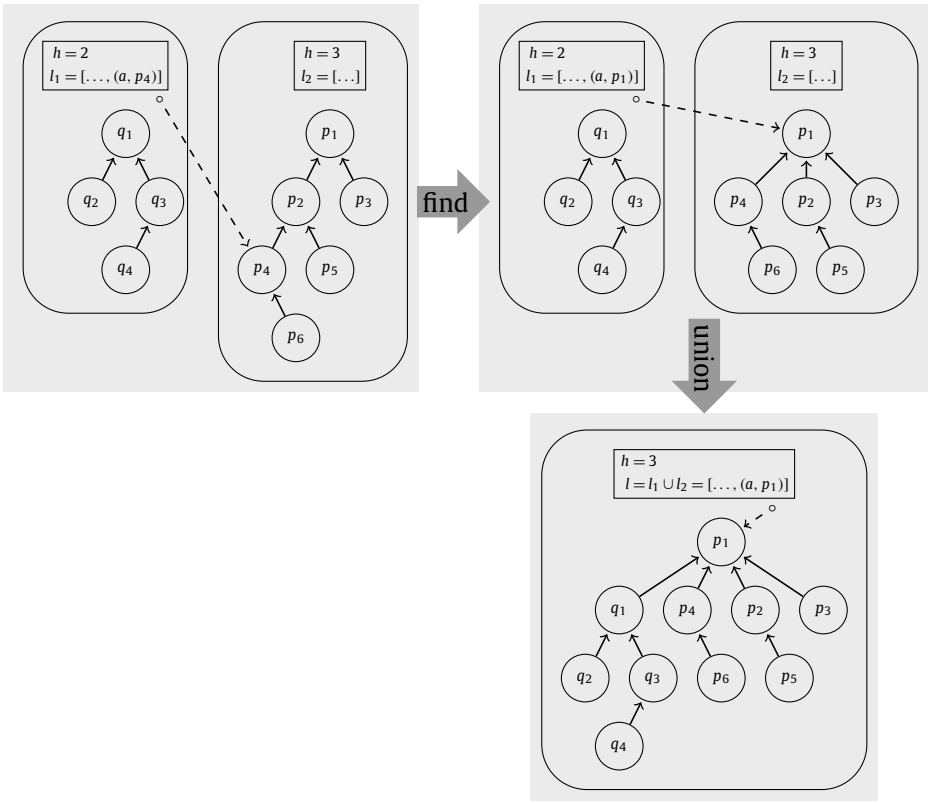

Fig. 8. An example use of a merge-find structure while adding an epistemic class to the set of epistemic classes with a guaranteed winning strategy. The two stages of executing *union*(q_1, p_4) (finding the root of an epistemic class and joining two sets of classes) are shown. Note that an update of the arc labeled by a (between the pair of structures that contain q_1 and p_4) is also shown.

the reassignment of the ending points for transitions until the transition is used for the first time – see the description of function *find*.

5.3. Dynamic discarding of irrelevant submodels

The optimizations presented in Sections 5.1 and 5.2 can be applied in the approximate verification of any input models. Moreover, they offer an improvement in both the time and the space complexity of the computation. Unfortunately, preliminary tests showed that it was not enough to complete experiments for arbitrary $(5, 5)$ models of bridge endplay due to the memory bottleneck. This is because the size of the tree-like structure for bridge grows exponentially with the increase of the main parameter. The epistemic abstraction from Section 5.1 significantly reduces the size of the graph, but the whole model of global states is being generated *before* the abstraction is applied. More precisely, the model generation algorithm (so far) works by simulating the game, step by step, from the initial states. All the information about the previously generated states is stored in the memory, and used to generate subsequent states and transitions. This is needed in order to check, when generating a new state, if it has not been already added to the model. The question is: can we avoid it, e.g., by "zooming" in and out on parts of the model that are relevant at a given stage of the computation?

For the bridge endplay benchmark the answer is yes. Observe that the models of bridge endplay have *layered structure*, where the states of the next layer have only incoming transitions from the previous layer. Moreover, the epistemic indistinguishability classes never traverse across layers, see Fig. 9 for an illustration. In consequence, the epistemic abstraction can be done on the fly, and as soon as the states in layer l are completely generated, the algorithm can discard the explicit representation of states and transitions from level $l-1$, keeping only their epistemic equivalence classes that will be used in the verification phase. This reduces the memory requirements of model generation from the size of the explicit state graph to doubled diameter of the graph, more precisely the maximal aggregate width of two subsequent layers.

5.4. Experimental results for the optimized algorithm

The experiments in Section 4 were conducted using a straightforward, one can even say naive, implementation of the fixpoint approximation algorithms. Already for that implementation, the results looked very promising, scaling up to millions of states. In this section, we show that one can develop and apply various optimization techniques, both domain-independent

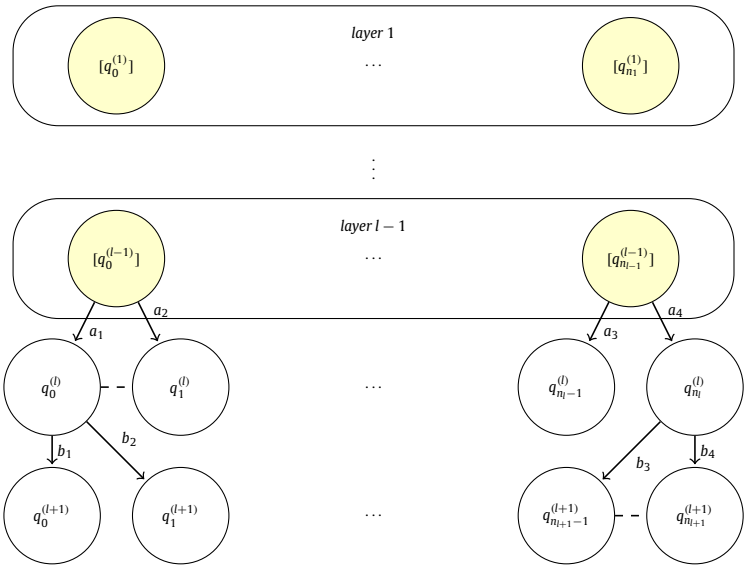

Fig. 9. Intermediate shape of the state space for a bridge game during the model generation.

Table 6
Results of the optimized algorithm for the bridge model.

(n,k)	#states	tgen	Lower approx.		Upper approx.		Match	Exact (tg+tv)
			tverif	%true	tverif	%false		
$(1,1)$	11	<0.0001	<0.0001	100%	<0.0001	0%	100%	0.12
$(2,2)$	346	<0.0001	<0.0001	56%	<0.0001	44%	100%	2.42 h*
$(3,3)$	12953	0.06	<0.0001	62%	<0.0001	38%	100%	timeout
$(4,4)$	617897	4.64	0.56	62%	0.26	38%	100%	timeout
$(5,5)^\star$	2443467	34.00	3.0	100%	2.0	0%	100%	timeout
$(5,5)$	15190971	124.00	8.5	50%	6.0	50%	100%	timeout
$(6,6)^\star$	70094091	3779.00	667.0	100%	78.0	0%	100%	timeout

and domain-specific, that complement the general approximation scheme. We have focused on optimizing the data structures and operations on them. Most likely, other kinds of optimizations can be applied as well; we plan to study the issue in the future.

The optimized algorithms were implemented in C++ which offered better control of data management than Python. We ran the experiments in the same way and in the same environment as before. The results of the experiments with the optimized algorithms are shown in Table 6. When compared to Table 3, one can see further dramatic speedup and a lower memory usage. The algorithms were able to generate and verify a model with over 70 million states in less than 75 minutes. Perhaps more importantly, the verification of the handpicked $(5,5)^\star$ model (which marked the limit of our capability in Section 4) ran almost 3000 times (!) faster than with the straightforward implementation. This strongly suggests that the potential for further improvement is still large.

Our optimized algorithms reached their limit with randomly generated $(6,6)$ models, facing memout during model generation. A possible way to overcome the limitation is by state abstraction of the model. We discuss the idea in Section 6.

Symbolic verification. It should be noted that the optimizations discussed in this section are *not* based on a symbolic representation of the state space. The symbolic approach that triggered the success of model checking in verification of large models is typically based on Binary Decision Diagrams (BDDs) [20,16] that are also behind the engine of MCMAS [53]. Our preliminary attempts in this direction have been disappointing. We suspect that the reason lies in the difficulty of ensuring that a selected one-step strategy is uniform in a given common knowledge neighborhood. This might have required a finer analysis of the state space than is feasible by means of a compressed representation using BDDs. We leave a more thorough study of the topic for future work.

6. Combining fixpoint approximation and abstraction

Abstraction [27,25] is a technique that can significantly reduce the size of the model, and hence also the time needed to complete the model checking. The idea is to cluster similar states in the model (called henceforth the *concrete model*) into equivalence classes that would serve as states in the new model (the *abstract model*). Similarly, we group similar concrete actions into abstract actions. For temporal logic, *may/must abstractions* are often used, typically combined with 3-valued

semantics [15,36]. May abstraction potentially adds (but never removes!) paths from the system, so that it consistently underapproximates universal formulae ("for all paths") and overapproximates existential ones ("there is a path"). Must abstraction does the opposite. If lucky, they produce the output of the verification, but in general can give inconclusive answers. Thus, they work in a very similar way to our fixpoint approximations, the only difference being that abstractions transform models rather than formulae.

We propose that, in case the verification problem is very hard computationally, it can be beneficial to combine both kinds of approximations. That is, one can transform the model *and* the formula; we only need to take care that both sides of the approximation produce the same kind of bound (either lower or upper one). We begin by proposing a variant of abstraction for $\mathbf{ATL}_{\mathrm{ir}}$, that has been strongly inspired by the three-valued abstraction schemes for strategic ability in concurrent games [8,7]. Then, we show how to combine it with the fixpoint approximation from Section 3, and present an experimental evaluation.

6.1. State and action abstraction for $\mathbf{ATL}_{\mathrm{ir}}$

Here, we semi-formally explain the idea of the abstraction. The technical details are presented in Appendix B. Similarly to may/must abstractions, we define the *lower abstraction* $\mathcal{A}^L_A(M)$ *of* M *with respect to* A and the *upper abstraction* $\mathcal{A}^U_A(M)$ *of* M *with respect to* A. $\mathcal{A}^L_A(M)$ underapproximates A's abilities in M, and $\mathcal{A}^U_A(M)$ overapproximates A's abilities in M. In both cases, the general structure is that of a nondeterministic iCGS where the opponents of A have been removed from the model. The actions of the agents in $\mathbb{A}\mathrm{gt} \setminus A$ are incorporated into the nondeterministic transition function.

In order to obtain the abstractions, we start with an *abstraction generator* $\mathcal{A} = (\mathcal{A}_S, \mathcal{A}_{Ac})$ with the following components: $\mathcal{A}_S$ that "clusters" concrete states into abstract states, and $\mathcal{A}_{Ac}$ that maps concrete actions into abstract actions. The lower abstraction is obtained as follows:

- We pessimistically fix the valuation of propositions: p holds in the abstract state $\widehat{q}$ if it holds in every concrete state $q' \in \widehat{q}$.
- Two abstract states are indistinguishable if any of their concrete states are.
- Protocols: we look at all the concrete states that appear in *all the abstract states indistinguishable from* $\widehat{q}$, and take the actions that appear in *all of their protocols*, modulo the renaming of actions with $\mathcal{A}_{Ac}$.
- The nondeterministic transition function: $\widehat{q_2} = \widehat{o}(\widehat{q_1}, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$ in the abstract model $\mathcal{A}^L_A(M)$ if there exists a pair of concrete states $q_1 \in \widehat{q_1}$ and $q_2 \in \widehat{q_2}$ in M with a transition from q_1 to q_2, labeled accordingly. That is, we aim at the *may* abstraction of paths.

The upper abstraction is constructed as follows:

- The valuation of propositions is optimistic, i.e., p holds in the abstract state $\widehat{q}$ if it holds in at least one concrete state $q' \in \widehat{q}$.
- Two abstract states are indistinguishable if all of their concrete states are.
- Protocols: we look at all the concrete states that appear in *at least one abstract state indistinguishable to* $\widehat{q}$, and take the actions that appear in *any of the protocols there*, modulo the renaming of actions with $\mathcal{A}_{Ac}$.
- Transitions: $\widehat{q_2} = \widehat{o}(\widehat{q_1}, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$ in the abstract model $\mathcal{A}^U_A(M)$ if, for every concrete state $q_1 \in \widehat{q_1}$, there is some $q_2 \in \widehat{q_2}$ with a transition from q_1 to q_2, labeled (after renaming) by the tuple of actions $(\widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$. This way, we obtain the *must* abstraction of paths.

Again, the details of the construction, and a formal statement of its correctness, can be found in Appendix B. We will show an example of the abstraction in Section 6.4.

6.2. Approximation semantics for $\mathbf{ATL}_{\mathrm{ir}}$ in abstract models

The abstractions transform a given iCGS in such a way that one of them underapproximates, and the other one overapproximates the abilities of a given coalition A, and hence also the truth value of all the formulae $\langle\langle A\rangle\rangle\gamma$ with no nested strategic modalities. Lifting the scheme to arbitrary formulae of $\mathbf{ATL}_{\mathrm{ir}}$ is straightforward, one only needs to take care of negation. We do it by *tagging* the formula recursively with a superscript, either L or U. Later, the formulae tagged with L will be evaluated in the lower abstraction of M, and the ones tagged with U will be evaluated in the upper abstraction of M. For example, the lower abstraction of $\langle\langle A\rangle\rangle_{\mathrm{ir}}\mathbf{F}\neg\langle\langle B\rangle\rangle_{\mathrm{ir}}\mathbf{G}(\mathrm{p}_1 \wedge \neg\mathrm{p}_2)$ should produce the following tags: $\langle\langle A\rangle\rangle^L_{\mathrm{ir}}\mathbf{F}\neg\langle\langle B\rangle\rangle^U_{\mathrm{ir}}\mathbf{G}(\mathrm{p}_1^U \wedge \neg\mathrm{p}_2^L)$. That is, the first strategic modality should be interpreted in $\mathcal{A}^L_A(M)$ and the second one in $\mathcal{A}^U_B(M)$. For the atomic propositions, the set of agents generating the abstraction is irrelevant. Thus, they can be interpreted e.g. in $\mathcal{A}^U_\emptyset(M)$ and $\mathcal{A}^L_\emptyset(M)$, respectively.

Formally, tagging for the lower abstraction is defined recursively by:

$TR_L(p) = p^L$, indicating that we will look at the lower abstraction for p;

$$TR_L(\langle\langle A\rangle\rangle_{\mathrm{ir}}\varphi) = \langle\langle A\rangle\rangle^L_{\mathrm{ir}} TR_L(\varphi)\text{: analogously;}$$
$$TR_L(\neg\varphi) = \neg TR_U(\varphi)\text{: negation switches between approximations;}$$
$$TR_L(\varphi \wedge \psi) = TR_L(\varphi) \wedge TR_L(\psi), \quad TR_L(\mathbf{X}\varphi) = \mathbf{X}TR_L(\varphi),$$
$$TR_L(\mathbf{G}\varphi) = \mathbf{G}TR_L(\varphi), \quad \text{and} \quad TR_L(\psi\,\mathbf{U}\,\varphi) = TR_L(\psi)\,\mathbf{U}\,TR_L(\varphi),$$
$$\text{i.e., the translation distributes over the other operators.}$$

Similarly for the upper abstraction:

$$TR_U(p) = p^U;$$
$$TR_U(\langle\langle A\rangle\rangle_{\mathrm{ir}}\varphi) = \langle\langle A\rangle\rangle^U_{\mathrm{ir}} TR_U(\varphi);$$
$$TR_U(\neg\varphi) = \neg TR_L(\varphi);$$
$$TR_U(\varphi \wedge \psi) = TR_U(\varphi) \wedge TR_U(\psi), \quad TR_U(\mathbf{X}\varphi) = \mathbf{X}TR_U(\varphi),$$
$$TR_U(\mathbf{G}\varphi) = \mathbf{G}TR_U(\varphi), \quad \text{and} \quad TR_U(\psi\,\mathbf{U}\,\varphi) = TR_U(\psi)\,\mathbf{U}\,TR_U(\varphi).$$

Now, we define the lower and upper approximation semantics $\models^L$ and $\models^U$. Given are: an iCGS M and an abstraction generator $\mathcal{A}$. We define a family of iCGS's: $M^x_A = \mathcal{A}^x_A(M)$, one for every $x \in \{L, U\}$ and $A \subseteq \mathbb{A}\mathrm{gt}$. To the semantics in Section 2, we add the following clauses for the new formulae p^x, $\langle\langle B\rangle\rangle^x_{\mathrm{ir}}\varphi$:

- $M^y_A, \widehat{q} \models p^x$ iff $M^x_A, \widehat{q} \models p$;
- $M^y_A, \widehat{q} \models \langle\langle B\rangle\rangle^x_{\mathrm{ir}}\varphi$ iff $M^x_B, \widehat{q} \models \langle\langle B\rangle\rangle_{\mathrm{ir}}\varphi$.

Finally, we define:

- $M, q \models^L \varphi$ iff $M^L_{\emptyset}, \mathcal{A}_S(q) \models TR_L(\varphi)$;
- $M, q \models^U \varphi$ iff $M^U_{\emptyset}, \mathcal{A}_S(q) \models TR_U(\varphi)$.

Theorem 22. *For every formula φ of* **ATL**$_{\mathrm{ir}}$*, we have*

$$M, q \models^L \varphi \quad \Rightarrow \quad M, q \models \varphi \quad \Rightarrow \quad M, q \models^U \varphi.$$

Proof. Straightforward induction on the structure of the formula, based on Propositions 24 and 25 (see Appendix B). □

6.3. Generalized approximation semantics: combining approximation on models and formulae

We have presented two methods of approximating the output of model checking for formulae of **ATL**$_{\mathrm{ir}}$. One, proposed and studied in Sections 3–4, transforms formulae into ones that provide a lower and an upper bound on the truth value of the original formula. The other one, discussed in the preceding subsections, produces analogous transformations of models. Here, we point out that both methods can be combined in a straightforward manner. To this end, we extend the tagging schemes of strategic modalities to the steadfast next step operator, and make the tagging distribute over fixpoint operators:

$$TR_L(\langle A\rangle^\bullet\varphi) = \langle A\rangle^{\bullet L} TR_L(\varphi);$$
$$TR_L(Z) = Z;$$
$$TR_L(\mu Z.\varphi) = \mu Z.TR_L(\varphi) \quad \text{and} \quad TR_L(\nu Z.\varphi) = \nu Z.TR_L(\varphi),$$

and analogously for TR_U. We also add the following clause:

- $M^y_A, \widehat{q} \models \langle B\rangle^{\bullet x}\varphi$ iff $M^x_B, \widehat{q} \models \langle B\rangle^\bullet\varphi$.

The extension to **AEμC** is straightforward. Now, the result of Theorem 22 can be extended to the combination of abstraction and fixpoint approximation:

Theorem 23. *For every formula φ of* **ATL**$_{\mathrm{ir}}$*, we have*

$$M, q \models^L tr_L(\varphi) \quad \Rightarrow \quad M, q \models \varphi \quad \Rightarrow \quad M, q \models^U tr_U(\varphi).$$

Table 7
Verification with abstraction: results for the bridge model.

(n,k)	AL	#states	Lower abstraction			Upper abstraction			Match
			tgen	tverif	%true	tgen	tverif	%false	
(5, 5)	1	1328192	258	1.4	0%	330	1.0	0%	0%
(5, 5)	2	2481220	323	2.8	0%	412	3.4	0%	0%
(5, 5)	3	6223840	420	5.4	0%	500	7.6	0%	0%
(5, 5)	4	9124109	490	8.25	50%	612	9.25	25%	75%
(5, 5)	5	15190971	124	8.5	50%	124	6.0	50%	100%

Table 8
Further results for abstractions of the bridge model.

(n,k)	AL	#states	Lower abstraction			Upper abstraction			Match
			tgen	tverif	%true	tgen	tverif	%false	
(6, 6)*	1	272113	680	<1	0%	1203	<1	0%	0%
(6, 6)*	2	408127	714	1	0%	989	1	0%	0%
(6, 6)*	3	1420924	739	2	0%	993	2	0%	0%
(6, 6)*	4	6925594	874	11	100%	1064	9	0%	100%
(6, 6)*	5	13977070	1126	25	100%	1371	21	0%	100%
(6, 6)*	6	70094091	3779	667	100%	3779	78	0%	100%

Proof. Straightforward induction on the structure of the formula, based on Theorem 14, Proposition 19, Proposition 24, and Proposition 25. □

In Section 6.5, we will see to what extent the combined scheme allows to verify larger instances of the bridge endplay scenario.

6.4. Lower and upper abstractions for the bridge model

In order to perform experimental evaluation of how abstraction combines with the fixpoint approximation of formulae, we go back to the bridge endplay scenario in Section 4.3. A natural abstraction, often used by human players, is to ignore the ranks of cards below a particular level R. That is, the player registers the rank and the suit of each card from R up, and only the suit for low cards below R. For example, if $R = J$ and the hand of the player is ♠$AK92$ ◇10 ♣$J653$, then the abstraction produces ♠$AKxx$ ◇x ♣$Jxxx$ (meaning: the Ace, the King, and two low cards in Spades, one low card in Diamonds, and the Jack plus three low cards in Clubs). Similarly, an opponent's action of playing ◇8 will appear as ◇x in the abstract model. This way, we obtain a scalable family of abstractions for every bridge endplay model M, that allows to "zoom" in and out on the model, depending on the need.

Formally, our abstraction generators are based on a family of deck simplifiers $simpl_r : Deck_n \to Deck_r$, $1 \leq r \leq n$, defined by:

$$simpl_r((rank, suit)) = \begin{cases} (rank, suit) & \text{if } rank > n - r + 1 \\ (n - r + 1, suit) & \text{else} \end{cases}$$

That is, the upper $r-1$ ranks are kept intact, and all the lower ones are clustered together. Note that, for $r = 1$, all the ranks collapse. Conversely, for $r = n$, the function does not change the deck, and hence the endplay model is left unchanged. Now, $\mathcal{A}^r = (\mathcal{A}^r_S, \mathcal{A}^r_{Ac})$ with:

- $\mathcal{A}^r_S((hands, tricks, next, board, lead, history, clock, suit)) = (simpl_r(hands), tricks, next, simpl_r(board), lead, simpl_r(history), clock, suit)$,[12]
- $\mathcal{A}^r_{Ac}(c) = simpl_r(c)$, and
- $\mathcal{A}^r_{Ac}(wait) = wait$.

6.5. Experiments: combining abstraction and fixpoint approximations

We have conducted three series of experiments with abstractions of the bridge endplay models. First, we re-approached the largest models that we managed to verify in the previous sections. We looked at the balance between accuracy and performance, obtained by different granularity levels of abstraction. The results are shown in Tables 7 and 8. Table 7 presents the output of experiments for randomly generated models (5, 5), i.e., models of full play with each player holding initially

[12] We recall that a detailed description of the bridge model can be found in Appendix A.

Table 9
Further results for abstractions of the bridge model.

(n,k)	AL	#states	Lower abstraction			Upper abstraction			Match
			tgen	tverif	%true	tgen	tverif	%false	
(13, 1)	6	11	<1	<1	65%	<1	<1	15%	80%
(13, 2)	6	243	<1	<1	45%	<1	<1	20%	65%
(13, 3)	6	12504	<1	0.1	55%	<1	<1	15%	70%
(13, 4)	6	545323	10	0.5	50%	12	0.45	30%	80%
(13, 5)	6	6406684	562	7.2	20%	646	9.2	0%	20%

5 cards. We studied all possible levels of abstraction from $r = 1$ (all ranks of cards removed) up. The results for $r = 5$ (full model, no abstraction) form the baseline; we repeat them after Section 5.4. Table 8 presents an analogous study for the handpicked model (6, 6), the same as in Section 5.4.

Finally, we model checked existence of winning strategies in k-endplay with the full deck of cards ($n = 13$), for a fixed level of abstraction ($r = 6$) and different values of k. The results are shown in Table 9. This series of experiments is especially interesting, as it best captures what happens in reality. Middle-level human players deal with the complexity of the full deck of cards by discerning between ranks from A down to 10, and abstracting away from the "low ranks" between 9 and 2.

In all the tables, the columns present the following information:

- the parameters of the model (n, k), i.e., the number of cards per suit in the deck (n) and the initial number of cards per player (k);
- the level of the abstraction (AL$= r$);
- the size of the state space after abstraction (#states);
- the generation time for the lower and the upper abstraction of the model (tgen),
- the time and the output of verification (tverif, %true, %false) for model checking of the lower and the upper abstraction,
- the percentage of cases where the bounds have matched (Match).

The times are given in seconds. The experiments were run in exactly the same environment as before. We used the optimized version of our fixpoint algorithm, presented in Section 5. The results in each row are averaged over 20 randomly generated instances, except for ($\star$) where only 1 hand-crafted instance was used.

Discussion of the results. The first two series of experiments show, unsurprisingly, that stronger abstraction saves much of the verification time, but also removes from the model information that can be vital for the verification output. In particular, too strong abstraction removes information needed to identify more sophisticated winning strategies. Of course, when combining *two* approximation techniques, one should expect the gap between the bounds to be significant. Still, the gap in our experiments was clearly due to abstraction, and not because of the fixpoint approximation (this is evident from the baseline results). On a more positive note, the results in Table 8 demonstrate that, for some models, conclusive model checking with abstraction can be done faster by an order of magnitude, compared to model checking without abstraction. Note also that conclusive verification for the $(6, 6)^\star$ model was possible without discerning half of the card ranks in the deck (abstraction level $r = 4$).

The results in the last series are especially hopeful. With abstraction, we were able to verify models which had been beyond our reach in their full form. The accuracy of the approximation ranged between 20% and 80%. This means that combining approximation on formulae and approximation on models allows for model checking of some instances that are too complex for either of the approximation methods alone.

7. Conclusions

Synthesis of winning strategies for imperfect information games is well known to be hard. Similarly, verification of strategic properties in scenarios with imperfect information is difficult, both theoretically and in practice. In this paper, we suggest that model checking of logics like $\mathbf{ATL}_{\mathrm{ir}}$ can be in some cases performed by computing an under- and an overapproximation of the $\mathbf{ATL}_{\mathrm{ir}}$ specification, and comparing if the bounds match.

We propose such approximations, prove their correctness, and show that, for singleton coalitions, their values can be computed in polynomial time. We also propose novel benchmarks for experimental validation. No less importantly, we report very promising experimental results, in performance as well as accuracy of the output. To our best knowledge, this is the first successful attempt at approximating strategic abilities under imperfect information by means of fixpoint methods. Finally, we demonstrate that there is much room for improvement, both in the sense of optimizing the algorithms and enhancing the applicability of the method. For the former, we propose a number of refinements, most notably an optimization of operations on data structures based on disjoint sets. For the latter, we show that combining our method with *may/must abstraction* allows to verify some instances that are too complex for either of the approximation methods alone.

The results open up multiple avenues for future work. On one hand, we plan to look for even tighter fixpoint approximations of $\mathbf{ATL}_{\mathrm{ir}}$ formulae. Secondly, our approximation scheme is rather inefficient for abilities of coalitions, especially ones

whose members have largely orthogonal views of the state of the system; this should be improved. As noted in Section 5, the problem of designing specialized data structures for dealing with the requirement of uniformity of strategies is still open and needs to be addressed. Another interesting topic is a better approximation of abilities for agents with perfect recall. Last but not least, we would like to apply the methodology, possibly combined with model reduction schemes such as in [9,49], to verify coercion-related properties of actual e-voting protocols. We have already made the first step in [48] where the framework of fixpoint approximation is used to verify coercion resistance in a simplified version of the SELENE voting protocol [64]. The results reported in [48] empirically confirm that the approach based on fixpoint approximations of strategic abilities often produces conclusive results (i.e., the approximations are tight). Moreover, a detailed comparison with the state-of-the-art algorithms implemented in the MCMAS model checker show a consistent superiority of the approximate verification.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

Wojciech Jamroga, Michał Knapik, and Damian Kurpiewski acknowledge the support of the National Centre for Research and Development (NCBR), Poland, under the PolLux project VoteVerif (POLLUX-IV/1/2016). We also thank Sasha Rubin and the anonymous reviewers of AAMAS and AIJ for their helpful comments.

Appendix A. Bridge model: the details

Our iCGS for the bridge endplay is constructed as follows.

Agents and states. Let us first define the set of cards used in the play (i.e., the *deck*). Recall the main parameters (n, k) of the model, with n being the number of cards per suit, and k the initial number of cards per hand. The set of possible *ranks* is $Ranks_n = \{1, \ldots, n\}$, where n represents the Ace, $n-1$ stands for the King, etc. The set of *suits* is $Suits = \{1, \ldots, 4\}$, with 1 for the Spades (♠), 2 for the Hearts (♡), and so on. Now, $Deck_n = Ranks_n \times Suits$. The four *positions* at the table are labeled after the four cardinal directions $\mathbf{S}, \mathbf{W}, \mathbf{N}, \mathbf{E}$, and represented by the set $Pos = \{0, \ldots, 3\}$. The set of *agents* is $\mathbb{A}\mathrm{gt} = \{\mathbf{S}, \mathbf{W}, \mathbf{E}\}$, with the declarer ($\mathbf{S}$) handling the cards at the South and the North positions, and $\mathbf{W}, \mathbf{E}$ handling their own hands. The set of global states St of the iCGS consists of tuples

$$(hands, tricks, next, board, lead, history, clock, suit),$$

each representing a different state of the play. The components of the tuple store the following information:

- $hands = (hand_{\mathbf{S}}, hand_{\mathbf{W}}, hand_{\mathbf{N}}, hand_{\mathbf{E}})$, with pairwise disjoint sets $hand_i \subseteq Deck_n$. Internally, this is represented by four arrays describing cards that each player holds, where -1 means an already played card. Initially, the cards in each hand are sorted in the ascending order;
- $history \subseteq Deck_n$: an array containing all the already played cards, sorted in the ascending order;
- $tricks \in \{0, \ldots, k\} \times \{0, \ldots, k\}$: two numbers, describing the current intermediate result of the game (how many tricks each team has won);
- $suit \in Suits \cup \{-1\}$: the suit of the current round or -1 if yet undetermined;
- $lead \in Pos$: the number of the position from which the current round started;
- $next \in Pos$: the number of the position from which the next card will be played;
- $board = (card_1, \ldots, card_4)$: the cards currently lying on the table, with $card_i \in Deck_n \cup \{-1\}$, and -1 denoting that the card on this position has not been played yet;
- $clock \in \{0, \ldots, 4\}$: the stage of the current round ($0, \ldots, 3$ for playing the cards from subsequent positions, 4 for collecting the cards and computing the results of the round).

Parts of example bridge models $(1, 1)$ and $(2, 2)$ are shown in Figs. A.10 and A.11.

Actions and transitions. The actions available to each player are: $d_i(q) = hand_i(q)$ if it is player i's turn to play a card, and $d_i(q) = \{wait\}$ otherwise. An example transition is depicted in Fig. A.10.

Indistinguishability relation. We recall that the only relevant epistemic relation for the model checking experiments in this paper is $\sim_{\mathbf{S}}$, i.e., the relation for the declarer. Two states are indistinguishable to $\mathbf{S}$ if the values of all variables except for the hands of the opponents are exactly the same, and the opponents' hands are of the same size (contain the same number of cards). For instance, an indistinguishable state to the one in Fig. A.11 can be obtained by swapping the Ace of Spades and the King of Clubs between the hands of the opponents, i.e., the one with **hands:** [[−1, A♡], [K♣, −1], [K♠, −1], [A♣, A♠]].

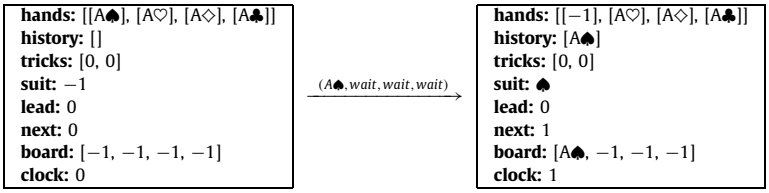

Fig. A.10. Example initial state of a $(1, 1)$ bridge model, with an opening transition.

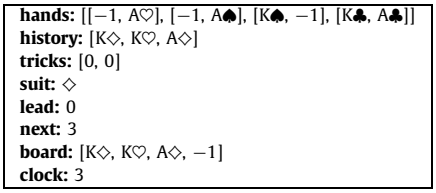

Fig. A.11. A state of bridge endplay for $n = 2, k = 2$.

Appendix B. State and action abstraction: the details

We present the details of the state and action abstraction for $\mathbf{ATL}_{\mathrm{ir}}$ over iCGS. The idea is to group states (respectively, actions) into abstract states (resp. abstract actions) according to a given equivalence relation. We begin by defining the equivalence relations, based on mappings that we call abstraction generators. Then, we propose how to construct the lower and the upper abstraction of models, that preserve the lower (resp. upper) bounds of strategic abilities for a given coalition.

B.1. Abstraction generators

Given are: a countable set of atomic propositions *Props*, an imperfect information concurrent game structure $M = (\mathbb{A}\mathrm{gt}, St, Act, d, o, V, \{\sim_a | a \in \mathbb{A}\mathrm{gt}\})$ satisfying all the standard requirements, and a coalition $A = \{a_1, \ldots, a_{|A|}\} \subseteq \mathbb{A}\mathrm{gt}$. Moreover, we are given the sets AST and AAC from which we will draw abstract states and actions.

In order to obtain the abstraction, we start with an *abstraction generator* $\mathcal{A} = (\mathcal{A}_S, \mathcal{A}_{Ac})$ with the following components:

- $\mathcal{A}_S : St \to AST$ "clusters" concrete states into abstract states. Note: $\mathcal{A}_S$ does not have to preserve V, i.e., it can cluster together states with different valuations of propositions;
- $\mathcal{A}_{Ac} : \mathbb{A}\mathrm{gt} \times St \times Act \to AAC$ maps concrete actions into abstract actions. Note that the way an actual action is mapped may depend on the agent who executes it, and the state where it is executed.

We also lift the generators to sets in the natural way: $\mathcal{A}_S(Q) = \{\mathcal{A}_S(q) \mid q \in Q\}$, $\mathcal{A}_{Ac}(a, q, \Theta) = \{\mathcal{A}_{Ac}(a, q, \alpha) \mid \alpha \in \Theta\}$, $\mathcal{A}_{Ac}(a, Q, \Theta) = \{\mathcal{A}_{Ac}(a, q, \Theta) \mid q \in Q\}$, and so on.

Note that the state and action abstraction generators $\mathcal{A}_S$ and $\mathcal{A}_{Ac}$ define equivalence relations on the sets St and Act, respectively. We write $[q] = \{q' \mid \mathcal{A}_S(q') = \mathcal{A}_S(q)\}$ and $[\alpha]_{a,q} = \{\alpha' \mid \mathcal{A}_{Ac}(a, q, \alpha') = \mathcal{A}_{Ac}(a, q, \alpha)\}$ to denote their abstraction classes. Moreover, we will sometimes identify the abstract states (resp. actions) with the corresponding abstraction classes. That is, we can e.g. write $[q]$ and $\mathcal{A}_S(q)$ interchangeably by a slight abuse of notation.

B.2. Lower and upper abstractions of models: general structure

We will now define the *lower abstraction* $\mathcal{A}^L_A(M)$ *of* M *with respect to* A, as well as the *upper abstraction* $\mathcal{A}^U_A(M)$ *of* M *with respect to* A, that approximate A's abilities in M. In both cases, the general structure is that of a nondeterministic iCGS[13]:

$$\widehat{M} = (\widehat{\mathbb{A}\mathrm{gt}}, \widehat{St}, \widehat{Act}, \widehat{d}, \widehat{o}, \widehat{V}, \{\widehat{\sim}_a \mid a \in \widehat{\mathbb{A}\mathrm{gt}}\}),$$

where:

- $\widehat{\mathbb{A}\mathrm{gt}} = A$,
- $\widehat{St} = \mathcal{A}_S(St)$, and
- $\widehat{Act} = \mathcal{A}_{Ac}(\mathbb{A}\mathrm{gt}, St, Act)$.

[13] We define *nondeterministic imperfect information concurrent game structures* as a simple extension of iCGS where: (i) protocols $d_a(q)$ can be empty, and (ii) the transition function o returns a subset of St rather than a single state (moreover, the subset can be empty).

That is, we remove from the model all the opponents, keeping only agents in A. The actions of the other agents will be "amalgamated" into the nondeterministic transition function. Secondly, we cluster states according to function $\mathcal{A}_S$, and actions according to function $\mathcal{A}_{Ac}$. Note that the way concrete actions are grouped into abstract actions may depend on who executes the action and where it is executed.

B.3. Lower abstraction: the specifics

For the lower approximation of A's abilities, we pessimistically fix the valuation of propositions: $\widehat{V}(\widehat{q}) = \bigcap_{q' \in \mathcal{A}_S^{-1}(\widehat{q})} V(q')$. That is, p holds in the abstract state $\widehat{q}$ iff it holds in every concrete state $q' \in \widehat{q}$.

The indistinguishability relations are given as follows: $\widehat{q_1} \widehat{\sim}_a \widehat{q_2}$ iff $q_1 \sim_a q_2$ for some $q_1 \in \widehat{q_1}, q_2 \in \widehat{q_2}$. In other words, two abstract states are indistinguishable if any of their concrete states are. Moreover, the protocols of the coalition members in the abstract model are defined as:

$$\widehat{d}_a(\widehat{q}) = \bigcap_{q' \in \mathrm{img}(\widehat{q}, \widehat{\sim}_a)} \mathcal{A}_{Ac}(a, q', d_a(q')).$$

That is, we look at all the concrete states that appear in *all the abstract states indistinguishable from* $\widehat{q}$, and take the actions that appear in *all of their protocols*, modulo the renaming of actions with $\mathcal{A}_{Ac}$. Notice that the renaming takes place *before* the intersection.

Now, the nondeterministic transition function can be defined as:

$$\widehat{o}(\widehat{q}, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|}) = \bigcup_{q' \in \mathcal{A}_S^{-1}(\widehat{q})} \mathcal{A}_S\left(succ(q', \mathcal{A}_{Ac}^{-1}(a_1, q', \widehat{\alpha}_1), \ldots, \mathcal{A}_{Ac}^{-1}(a_{|A|}, q', \widehat{\alpha}_{|A|}))\right)$$

for every $(\widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|}) \in \widehat{d}_A(\widehat{q})$. In other words, $\widehat{q_2} = \widehat{o}(\widehat{q_1}, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$ in the abstract model $\mathcal{A}_A^L(M)$ iff there exists a pair of concrete states $q_1 \in \widehat{q_1}$ and $q_2 \in \widehat{q_2}$ in M with a transition from q_1 to q_2, labeled accordingly.

We observe that:

1. We do not require the state abstraction generator $\mathcal{A}_S$ to be consistent with V (i.e., it is not required that $\mathcal{A}_S(q) = \mathcal{A}_S(q')$ implies $V(q) = V(q')$). Instead, we use the pessimistic (or skeptical) interpretation of atomic propositions in $\mathcal{A}_A^L(M)$. That is, making p true in $\mathcal{A}_A^L(M)$ is no easier than in M. In consequence, achieving $\langle\langle A\rangle\rangle \mathbf{X}\mathsf{p}$, $\langle\langle A\rangle\rangle \mathbf{G}\mathsf{p}$, and $\langle\langle A\rangle\rangle \mathsf{p}_1 \mathbf{U} \mathsf{p}_2$ can be only *harder* in the abstract model.
2. For every $a \in A$, $\widehat{d}_a$ is uniform by construction.
3. It may happen that $\widehat{d}_a(\widehat{q}) = \emptyset$. Then, the set of A's collective strategies is also empty, and in consequence all the formulae $\langle\langle A\rangle\rangle \gamma$ are false in $\mathcal{A}_A^L(M)$. This is not really a problem for us, since we will only use $\mathcal{A}_A^L(M)$ to compute the lower bound of A's abilities, cf. Proposition 24.
4. The construction consistently: (i) decreases the truth values of atomic propositions, (ii) increases the uncertainty of the agents in A, (iii) constrains their available choices, and (iv) applies a "may" abstraction to the transitions consistent with any remaining strategy.

In consequence, we obtain the following.

Proposition 24. *Let γ be an* **LTL** *formula containing no negations. Then, $\mathcal{A}_A^L(M), \mathcal{A}_S(q) \models \langle\langle A\rangle\rangle \gamma$ implies $M, q \models \langle\langle A\rangle\rangle \gamma$.*

B.3.1. Upper abstraction: the specifics

The valuation of propositions in $\mathcal{A}_A^U(M)$ is optimistic: $\widehat{V}(\widehat{q}) = \bigcup_{q' \in \mathcal{A}_S^{-1}(\widehat{q})} V(q')$. That is, p holds in the abstract state $\widehat{q}$ iff it holds in at least one concrete state $q' \in \widehat{q}$. The indistinguishability relations are given as: $\widehat{q_1} \widehat{\sim}_a \widehat{q_2}$ iff $q_1 \sim_a q_2$ for all $q_1 \in \widehat{q_1}, q_2 \in \widehat{q_2}$. In other words, two abstract states are indistinguishable if all of their concrete states are. The protocols of the coalition members in the abstract model are defined as:

$$\widehat{d}_a(\widehat{q}) = \bigcup_{q' \in \mathrm{img}(\widehat{q}, \widehat{\sim}_a)} \mathcal{A}_{Ac}(a, q', d_a(q')).$$

That is, we look at all the concrete states that appear in *at least one abstract state indistinguishable to* $\widehat{q}$, and take all the actions that appear in *any of the protocols there*, modulo the renaming of actions with $\mathcal{A}_{Ac}$. Notice, again, that the renaming takes place before the intersection.

Finally, the nondeterministic transition function is defined as:

$$\widehat{o}(\widehat{q}, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|}) = \bigcap_{q' \in \mathcal{A}_S^{-1}(\widehat{q})} \mathcal{A}_S\left(succ(q', \mathcal{A}_{Ac}^{-1}(a_1, q', \widehat{\alpha}_1), \ldots, \mathcal{A}_{Ac}^{-1}(a_{|A|}, q', \widehat{\alpha}_{|A|}))\right)$$

for every $(\widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|}) \in \widehat{d}_A(\widehat{q})$. In other words, $\widehat{q}_2 = \widehat{o}(\widehat{q}_1, \widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$ in the abstract model $\mathcal{A}^U_A(M)$ iff, for every concrete state $q_1 \in \widehat{q}_1$, there is some $q_2 \in \widehat{q}_2$ with a transition from q_1 to q_2, labeled (after renaming) by the tuple of actions $(\widehat{\alpha}_1, \ldots, \widehat{\alpha}_{|A|})$.

We observe that:

1. For any atomic proposition p, if it holds in q, then it also holds in $\mathcal{A}_S(q)$. This amounts to the optimistic (or credulous) interpretation of atomic propositions in $\mathcal{A}^U_A(M)$. In consequence, making p true in $\mathcal{A}^U_A(M)$ is no harder than in M. Achieving $\langle\langle A\rangle\rangle \mathbf{X}\mathsf{p}$, $\langle\langle A\rangle\rangle \mathbf{G}\mathsf{p}$, and $\langle\langle A\rangle\rangle \mathsf{p}_1 \mathbf{U}\, \mathsf{p}_2$ can be only made *easier* in the abstract model.
2. For every $a \in A$, $\widehat{d}_a$ is uniform by construction.
3. It may happen that, at some state $\widehat{q}$, there are no outgoing transitions. A subtler possibility is that $\widehat{q}$ has no outgoing transitions consistent with a given available strategy s_A of A. Note that this is not a problem as long as our semantics of $\langle\langle A\rangle\rangle \gamma$ looks only at the *infinite* outcome paths of s_A (rather than *maximal* ones). Then, all the runs going through (and hence ending in) $\widehat{q}$ will not be included in $out(\widehat{q}', s_A)$, which can only make $\langle\langle A\rangle\rangle \gamma$ *easier* to achieve from $\widehat{q}'$. In the extreme case, we get $out(\widehat{q}', s_A) = \emptyset$, which means that $\langle\langle A\rangle\rangle \gamma$ holds in $\widehat{q}'$ regardless of the path formula γ.
4. The construction consistently: (i) increases the truth values of atomic propositions, (ii) decreases the uncertainty of the agents in A, (iii) expands their choices, and (iv) applies a "must" abstraction to the transitions consistent with the resulting strategies.

In consequence, we obtain the following.

Proposition 25. *Let γ be an* **LTL** *formula with no negations. Then, $M, q \models \langle\langle A\rangle\rangle \gamma$ implies $\mathcal{A}^U_A(M), \mathcal{A}_S(q) \models \langle\langle A\rangle\rangle \gamma$.*

References

[1] T. Ågotnes, A note on syntactic characterization of incomplete information in ATEL, in: Proceedings of Workshop on Knowledge and Games, 2004, pp. 34–42.
[2] T. Ågotnes, V. Goranko, W. Jamroga, M. Wooldridge, Knowledge and ability, in: H.P. van Ditmarsch, J.Y. Halpern, W. van der Hoek, B.P. Kooi (Eds.), Handbook of Epistemic Logic, College Publications, 2015, pp. 543–589.
[3] R. Alur, L. de Alfaro, R. Grossu, T.A. Henzinger, M. Kang, C.M. Kirsch, R. Majumdar, F.Y.C. Mang, B.-Y. Wang jMocha, A model-checking tool that exploits design structure, in: Proceedings of International Conference on Software Engineering, ICSE, IEEE Computer Society Press, 2001, pp. 835–836.
[4] R. Alur, L. de Alfaro, T.A. Henzinger, S.C. Krishnan, F.Y.C. Mang, S. Qadeer, S.K. Rajamani, S. Tasiran, MOCHA: Modularity in Model Checking, Technical report, University of Berkeley, 2000.
[5] R. Alur, T.A. Henzinger, O. Kupferman, Alternating-time temporal logic, in: Proceedings of the 38th Annual Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society Press, 1997, pp. 100–109.
[6] R. Alur, T.A. Henzinger, O. Kupferman, Alternating-time temporal logic, J. ACM 49 (2002) 672–713.
[7] B. Francesco, A. Lomuscio, Agent-based abstractions for verifying alternating-time temporal logic with imperfect information, in: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems AAMAS, IFAAMAS, 2017, pp. 1259–1267.
[8] T. Ball, O. Kupferman, An abstraction-refinement framework for multi-agent systems, in: Proceedings of Logic in Computer Science, LICS, 2006, pp. 379–388.
[9] F. Belardinelli, R. Condurache, C. Dima, W. Jamroga, A.V. Jones, Bisimulations for verification of strategic abilities with application to ThreeBallot voting protocol, in: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems, AAMAS, IFAAMAS, 2017, pp. 1286–1295.
[10] F. Belardinelli, A. Lomuscio, A. Murano, S. Rubin, Verification of multi-agent systems with imperfect information and public actions, in: Proceedings of AAMAS, 2017, pp. 1268–1276.
[11] N. Belnap, M. Perloff, Seeing to it that: a canonical form for agentives, Theoria 54 (1988) 175–199.
[12] D. Berwanger, L. Kaiser, Information tracking in games on graphs, J. Log. Lang. Inf. 19 (4) (2010) 395–412.
[13] D. Berwanger, A.B. Mathew, M. van den Bogaard, Hierarchical information patterns and distributed strategy synthesis, in: Proceedings of ATVA, in: Lecture Notes in Computer Science, vol. 9364, Springer, 2015, pp. 378–393.
[14] R. Bordini, M. Fisher, C. Pardavila, M. Wooldridge, Model checking AgentSpeak, in: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, ACM, 2003, pp. 409–416.
[15] G. Bruns, P. Godefroid, Model checking partial state spaces with 3-valued temporal logics, in: Proceedings of Computer Aided Verification, CAV, 1999, pp. 274–287.
[16] R.E. Bryant, Binary decision diagrams, in: Handbook of Model Checking, Springer, 2018, pp. 191–217.
[17] N. Bulling, J. Dix, W. Jamroga, Model checking logics of strategic ability: complexity, in: M. Dastani, K. Hindriks, J.-J. Meyer (Eds.), Specification and Verification of Multi-Agent Systems, Springer, 2010, pp. 125–159.
[18] N. Bulling, W. Jamroga, Alternating epistemic mu-calculus, in: Proceedings of IJCAI-11, 2011, pp. 109–114.
[19] N. Bulling, W. Jamroga, Comparing variants of strategic ability: how uncertainty and memory influence general properties of games, J. Auton. Agents and Multi-Agent Syst. 28 (3) (2014) 474–518.
[20] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, L.J. Hwang, Symbolic model checking: 10^{20} states and beyond, Inf. Comput. 98 (2) (1992) 142–170.
[21] S. Busard, Symbolic Model Checking of Multi-Modal Logics: Uniform Strategies and Rich Explanations, PhD thesis, Universite Catholique de Louvain, 2017.
[22] S. Busard, C. Pecheur, H. Qu, F. Raimondi, Improving the model checking of strategies under partial observability and fairness constraints, in: Formal Methods and Software Engineering, in: Lecture Notes in Computer Science, vol. 8829, Springer, 2014, pp. 27–42.
[23] S. Busard, C. Pecheur, H. Qu, F. Raimondi, Reasoning about memoryless strategies under partial observability and unconditional fairness constraints, Inf. Comput. 242 (2015) 128–156.
[24] K. Chatterjee, L. Doyen, T.A. Henzinger, J.-F. Raskin, Algorithms for omega-regular games of incomplete information, Log. Methods Comput. Sci. 3 (3) (2007).
[25] E.M. Clarke, O. Grumberg, D.E. Long, Model checking and abstraction, ACM Trans. Program. Lang. Syst. 16 (5) (1994) 1512–1542.
[26] M.K. Colby, L.M. Yliniemi, K. Tumer, Autonomous multiagent space exploration with high-level human feedback, J. Aerosp. Inform. Syst. 13 (8) (2016) 301–315.

[27] P. Cousot, R. Cousot, Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints, in: Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, 1977, pp. 238–252.
[28] C. Dima, C. Enea, D.P. Guelev, Model-checking an alternating-time temporal logic with knowledge, imperfect information, perfect recall and communicating coalitions, in: Proceedings of Games, Automata, Logics and Formal Verification, GandALF, 2010, pp. 103–117.
[29] C. Dima, B. Maubert, S. Pinchinat, The expressive power of epistemic μ-calculus, CoRR, arXiv:1407.5166 [abs], 2014.
[30] C. Dima, B. Maubert, S. Pinchinat, Relating paths in transition systems: the fall of the modal mu-calculus, in: Proceedings of Mathematical Foundations of Computer Science, MFCS, in: Lecture Notes in Computer Science, vol. 9234, Springer, 2015, pp. 179–191.
[31] C. Dima, F.L. Tiplea, Model-checking ATL under imperfect information and perfect recall semantics is undecidable, CoRR, arXiv:1102.4225 [abs], 2011.
[32] L. Doyen, J.-F. Raskin, Games with imperfect information: theory and algorithms, in: In Lecture Notes in Game Theory for Computer Scientists, Cambridge University Press, 2011, pp. 185–212.
[33] B.A. Galler, M.J. Fisher, An improved equivalence algorithm, Commun. ACM 7 (5) (1964) 301–303.
[34] P. Gammie, R. van der Meyden MCK, Model checking the logic of knowledge, in: Proc. of the 16th Int. Conf. on Computer Aided Verification, CAV'04, in: LNCS, vol. 3114, Springer-Verlag, 2004, pp. 479–483.
[35] J.M. Gascueña, A. Fernández-Caballero, Review: on the use of agent technology in intelligent, multisensory and distributed surveillance, Knowl. Eng. Rev. 26 (2) (2011) 191–208.
[36] P. Godefroid, R. Jagadeesan, Automatic abstraction using generalized model checking, in: Proceedings of Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 2404, Springer, 2002, pp. 137–150.
[37] S.A. Gómez, A. Goron, A. Groza, I.A. Letia, Assuring safety in air traffic control systems with argumentation and model checking, Expert Syst. Appl. 44 (2016) 367–385.
[38] C. Greulich, S. Edelkamp, N. Eicke, Cyber-physical multiagent-simulation in production logistics, in: Proceedings of Multiagent System Technologies, MATES, 2015, pp. 119–136.
[39] D.P. Guelev, C. Dima, Epistemic ATL with perfect recall, past and strategy contexts, in: Proceedings of Computational Logic in Multi-Agent Systems, CLIMA, in: Lecture Notes in Computer Science, vol. 7486, Springer, 2012, pp. 77–93.
[40] D.P. Guelev, C. Dima, C. Enea, An alternating-time temporal logic with knowledge, perfect recall and past: axiomatisation and model-checking, J. Appl. Non-Class. Log. 21 (1) (2011) 93–131.
[41] N. He, D.Z. Zhang, Q. Li, Agent-based hierarchical production planning and scheduling in make-to-order manufacturing system, in: The Economics of Industrial Production, Int. J. Prod. Econ. 149 (Supplement C) (2014) C:117–130.
[42] X. Huang, R. van der Meyden, Symbolic model checking epistemic strategy logic, in: Proceedings of AAAI Conference on Artificial Intelligence, 2014, pp. 1426–1432.
[43] W. Jamroga, Some remarks on alternating temporal epistemic logic, in: B. Dunin-Keplicz, R. Verbrugge (Eds.), Proceedings of Formal Approaches to Multi-Agent Systems, FAMAS 2003, 2003, pp. 133–140.
[44] W. Jamroga, Logical Methods for Specification and Verification of Multi-Agent Systems, ICS PAS Publishing House, 2015.
[45] W. Jamroga, T. Ågotnes, Modular Interpreted Systems: a Preliminary Report, Technical Report Ifl-06-15, Clausthal University of Technology, 2006.
[46] W. Jamroga, J. Dix, Model checking ATL$_{ir}$ is indeed Δ_2^P-complete, in: Proceedings of EUMAS, in: CEUR Workshop Proceedings, vol. 223, 2006.
[47] W. Jamroga, M. Knapik, D. Kurpiewski, Fixpoint approximation of strategic abilities under imperfect information, in: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems, AAMAS, IFAAMAS, 2017, pp. 1241–1249.
[48] W. Jamroga, M. Knapik, D. Kurpiewski, Model checking the SELENE e-voting protocol in multi-agent logics, in: Proceedings of the 3rd International Joint Conference on Electronic Voting, E-VOTE-ID, in: Lecture Notes in Computer Science, vol. 11143, Springer, 2018, pp. 100–116.
[49] W. Jamroga, W. Penczek, P. Dembiński, A. Mazurkiewicz, Towards partial order reductions for strategic ability, in: Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems, AAMAS, IFAAMAS, 2018, pp. 156–165.
[50] W. Jamroga, W. van der Hoek, Agents that know how to play, Fundam. Inform. 63 (2–3) (2004) 185–219.
[51] A. Lomuscio, J. Michaliszyn, Verification of multi-agent systems via predicate abstraction against ATLK specifications, in: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, 2016, pp. 662–670.
[52] A. Lomuscio, H. Qu, F. Raimondi MCMAS, A model checker for the verification multi-agent systems, in: Proceedings of Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 5643, Springer, 2009, pp. 682–688.
[53] A. Lomuscio, H. Qu, F. Raimondi MCMAS, An open-source model checker for the verification of multi-agent systems, Int. J. Softw. Tools Technol. Transf. 19 (1) (2017) 9–30.
[54] A. Lomuscio, F. Raimondi MCMAS, A model checker for multi-agent systems, in: Proceedings of Tools and Algorithms for Construction and Analysis of Systems, TACAS, in: Lecture Notes in Computer Science, vol. 4314, Springer, 2006, pp. 450–454.
[55] A. Lomuscio, F. Raimondi, Model checking knowledge, strategies, and games in multi-agent systems, in: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, 2006, pp. 161–168.
[56] J. McCarthy, P.J. Hayes, Some philosophical problems from the standpoint of artificial intelligence, in: B. Meltzer, D. Michie (Eds.), Machine Intelligence, vol. 4, Edinburgh University Press, 1969, pp. 463–502.
[57] R.C. Moore, Reasoning about knowledge and action, in: Proceedings of International Joint Conference on Artificial Intelligence, IJCAI, William Kaufmann, 1977, pp. 223–227.
[58] R.C. Moore, A formal theory of knowledge and action, in: J. Hobbs, R.C. Moore (Eds.), Formal Theories of the Commonsense World, Ablex Publishing Corp., 1985.
[59] J.P. Müller, K. Fischer, Application impact of multi-agent systems and technologies: a survey, in: Agent-Oriented Software Engineering – Reflections on Architectures, Methodologies, Languages, and Frameworks, 2014, pp. 27–53.
[60] G. Peterson, J. Reif, Multiple-person alternation, in: Proceedings of the 20th Annual Symposium on Foundations of Computer Science, FOCS, IEEE Computer Society Press, 1979, pp. 348–363.
[61] J. Pilecki, M.A. Bednarczyk, W. Jamroga, Synthesis and verification of uniform strategies for multi-agent systems, in: Proceedings of CLIMA XV, in: Lecture Notes in Computer Science, vol. 8624, Springer, 2014, pp. 166–182.
[62] J. Pilecki, M.A. Bednarczyk, W. Jamroga, SMC: synthesis of uniform strategies and verification of strategic ability for multi-agent systems, J. Log. Comput. 27 (7) (2017) 1871–1895.
[63] F. Raimondi, Model Checking Multi-Agent Systems, PhD thesis, University College London, 2006.
[64] P.Y.A. Ryan, P.B. Rønne, V. Iovino Selene, Voting with transparent verifiability and coercion-mitigation, in: Financial Cryptography and Data Security: Proceedings of FC 2016. Revised Selected Papers, in: Lecture Notes in Computer Science, vol. 9604, Springer, 2016, pp. 176–192.
[65] G. Ryle, The Concept of Mind, Chicago University Press, 1949.
[66] P.Y. Schobbens, Alternating-time logic with imperfect recall, Electron. Notes Theor. Comput. Sci. 85 (2) (2004) 82–93.
[67] M.P. Singh, Cybersecurity as an application domain for multiagent systems, in: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, 2015, pp. 1207–1212.

[68] W. van der Hoek, A. Lomuscio, M. Wooldridge, On the complexity of practical ATL model checking, in: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, ACM, 2006, pp. 201–208.
[69] W. van der Hoek, M. Wooldridge, Tractable multiagent planning for epistemic goals, in: C. Castelfranchi, W.L. Johnson (Eds.), Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems, AAMAS-02, ACM Press, New York, 2002, pp. 1167–1174.
[70] R. van der Meyden, Optimizing epistemic model checking using conditional independence, in: Proceedings of Theoretical Aspects of Rationality and Knowledge, TARK, 2017, pp. 398–414.