Fundamenta Informaticae XXI? (2019) 1–45
DOI 10.3233/FI-2016-0000
IOS Press

Multi-Valued Verification of Strategic Ability

Wojciech Jamroga
Institute of Computer Science
Polish Academy of Sciences
Jana Kazimierza 5, 01-248 Warsaw, Poland
jamroga@ipipan.waw.pl

Beata Konikowska
Institute of Computer Science
Polish Academy of Sciences
Jana Kazimierza 5, 01-248 Warsaw, Poland
konikowska@ipipan.waw.pl

Damian Kurpiewski
Institute of Computer Science
Polish Academy of Sciences
Jana Kazimierza 5, 01-248 Warsaw, Poland
kurpiewski@ipipan.waw.pl,

Wojciech Penczek
Institute of Computer Science
Polish Academy of Sciences
Jana Kazimierza 5, 01-248 Warsaw, Poland
penczek@ipipan.waw.pl

Abstract. Some multi-agent scenarios call for the possibility of evaluating specifications in a richer domain of truth values. Examples include runtime monitoring of a temporal property over a growing prefix of an infinite path, inconsistency analysis in distributed databases, and verification methods that use incomplete anytime algorithms, such as bounded model checking. In this paper, we present *multi-valued alternating-time temporal logic* (mv-ATL$^*_\rightarrow$), an expressive logic to specify strategic abilities in multi-agent systems. It is well known that, for branching-time logics, a general method for model-independent translation from multi-valued to two-valued model checking exists. We show that the method cannot be directly extended to mv-ATL$^*_\rightarrow$. We also propose two ways of overcoming the problem. Firstly, we identify constraints on formulas for which the model-independent translation can be suitably adapted. Secondly, we present a model-dependent reduction that can be applied to all formulas of mv-ATL$^*_\rightarrow$. We show that, in all cases, the complexity of verification increases only linearly when new truth values are added to the evaluation domain. We also consider several examples that show possible applications of mv-ATL$^*_\rightarrow$ and motivate its use for model checking multi-agent systems.

1. Introduction

Alternating-time temporal logic ATL* and its less expressive variant ATL [1, 2] are probably the most popular logics that allow for reasoning about agents' abilities in strategic encounters. ATL* combines features of temporal logic and basic game theory, encapsulated in the main language construct of the logic, $\langle\langle A\rangle\rangle\gamma$, which can be read as "the group of agents A has a strategy to enforce γ". Property γ can include operators X ("next"), G ("always"), F ("eventually") and/or U ("until"). Much research on ATL* has focused on the way it can be used for verification of multi-agent systems, including theoretical studies on the complexity of model checking, as well as practical verification algorithms.

Multi-valued semantics have been already proposed for various temporal and temporal-epistemic logics, and applied in verification of distributed and multi-agent systems. In this paper, we extend the general approach of [3, 4] to verification of strategic abilities for agents and their coalitions. Many relevant properties of multi-agent systems are intuitively underpinned by the ability (or inability) of particular agents to obtain particular outcomes. For example, most functionality requirements can be specified as the ability of the authorized users to achieve their legitimate goals. At the same time, many security properties can be phrased in terms of the inability of unauthorized users to compromise the system. We show that reasoning about, and verification of such statements can be naturally conducted in richer domains of truth values, regardless of the actual notion of strategic play and constraints on the agents' epistemic capabilities. Moreover, multiple truth values incur no significant increase of either the theoretical or the practical complexity of the model checking problem. In fact, we show that multi-valued verification of strategic abilities can be usually done by means of an efficient translation to classical verification. We also present a case study that demonstrates the modeling value of the approach and its efficiency.

1.1. Multi-Valued Model Checking: Why Bother?

Typically, model checking is a yes/no problem. That is, the output is a classical truth value: $\top$ for "yes", $\bot$ for "no." However, it is sometimes convenient to consider the output of verification in a richer domain of values. This can be due to at least three reasons. First, our reasoning about the world can be based on a non-classical notion of truth (e.g., graded, fuzzy, constructive, defeasible etc.). For example, if an atomic statement pol ("the area is polluted") is evaluated according to measurements conducted in the environment, the possible outputs can include "highly," "much," "little," and "not at all," instead of simply "yes" and "no." Multi-valued semantics take those outputs directly as the possible truth values of pol, and hence also of more complex formulas involving pol. This is particularly useful in case of model checking realistic systems, as it allows to lift logical reasoning to a richer domain of answers in a compact way.

Secondly, we can understand the properties of the world in a classical way (being always completely true or fully false), but our model of the system can be only partially conclusive. A good example is runtime monitoring of temporal properties, where a temporal formula (interpreted with an infinite time horizon in mind) is checked on a finite but constantly growing sequence of events, observed so far. Consider for instance specification $\mathsf{F}\,\mathsf{cool}$ ("the reactor will be cooled down sometime in the future"). If cool has already occurred then the formula is clearly true whatever happens next. What if it has not occurred? Then, the formula may still turn out true (because cool may occur in subsequent steps), but it can also turn out false; effectively, the truth value is unknown in our current model. Likewise, formula $\mathsf{G}\,\mathsf{cool}$ ("cool will always hold") can be only proved false in the course of monitoring, or the model is inconclusive regarding its value. Indeed, well known approaches to runtime monitoring use multi-valued interpretation of temporal formulas in finite runs [5, 6].

Conflicting evidence coming from different sources is another reason why one may need to deal with an imperfect model of the system. This can happen, e.g., in case of a distributed knowledge base, where some components may be outdated and/or contain unreliable information. In classical logic, the deductive explosion would make any attempt at reasoning useless. In multi-valued logics, one can assign a special truth value ("inconsistent") to situations when conflicting evidence about the value of p exists, and conduct the verification in the usual way.

Thirdly, even if the model is complete and faithful, the verification procedure can be only partially conclusive. For instance, in bounded model checking [7, 8], a full transition system is given but the formula is checked on runs of length at most n. Now, again, $\mathsf{F}\,\mathsf{p}$ is clearly true if we find p to occur on every path in up to n steps. Otherwise, the output is inconclusive (because p might or might not occur in subsequent steps). The case of $\mathsf{G}\,\mathsf{p}$ is analogous.

Note that different sources of multiple truth values can easily combine. For example, runtime monitoring of a distributed knowledge base may require handling both the possible inconsistency of data across different locations and the uncertainty about how the system state will evolve. This kind of scenarios are most conveniently modeled by partially, rather than linearly ordered sets of answers. Overall, it should be up to the designer to decide which domain of truth values will be used for verification. In this paper, we provide a general methodology, together with flexible algorithms, that can be used for any distributive lattice of interpretation.

1.2. Technical Contribution and Outline of the Paper

The focus of the article is on general multi-valued verification of strategic abilities in multi-agent systems. The main contributions are:

1. We propose a multi-valued variant of alternating-time temporal logic ATL*, called mv-ATL$^*_{\rightarrow}$, over arbitrary lattices of logical values. We also show that the new logic is a conservative extension of the standard, 2-valued ATL*.

2. We study the multi-valued model checking problem for mv-ATL$^*_{\rightarrow}$. Similarly to the previous work on temporal model checking [3, 4], we do not propose dedicated algorithms for mv-ATL$^*_{\rightarrow}$. Instead, we look for general efficient translations from the multi-valued case to the 2-valued case. In this respect, we:

   (a) prove that no model-independent translation exists for the whole language of mv-ATL$^*_{\rightarrow}$,

   (b) identify a broad subclass of formulas for which such a translation can be obtained,

   (c) propose a recursive model-dependent translation that works for all instances of the problem.

3. We show that all results are insensitive to the actual notion of strategy and strategic play. In particular, they easily extend to verification of strategies under imperfect information.

4. Finally, we report an implementation of the verification algorithm based on the translation to 2-valued model checking, and evaluate its performance experimentally.

Point 2 might require further explanation. Multi-valued model checking often provides a *conceptual* approximation of the classical (two-valued) verification problem, especially in case the precise model is difficult to obtain. On the other hand, we show a *technical* reduction of multi-valued model checking to the two-valued variant. Thus, typically, a designer who wants to verify properties of a complex system, expressed in (classical) ATL*, would first come up with a conceptual approximation of the problem in mv-ATL$^*_{\rightarrow}$, then use the technical translation of the specification back to model checking of ATL*, and finally run the latter by means of an existing tool (such as MCMAS [9]).

The structure of the paper is as follows. We begin by presenting the context of our work: the alternating time logic ATL* in Section 2, and distributive lattices in Section 3. We also introduce our working example of drones monitoring the pollution in a city. In Section 4, we define the syntax and semantics of our multi-valued variant of ATL*, the logic mv-ATL$^*_{\rightarrow}$. Section 5 discusses the model checking problem for mv-ATL$^*_{\rightarrow}$ (in general and under certain restrictions). Until that point, we assume the classical interpretation of transitions in models, i.e., the multi-valued approach is applied only to the labeling of *states*. Section 6 extends our definitions and results to models with many-valued transitions. We also show that the idea of *may/must abstraction* can be seen as a special case of model checking mv-ATL$^*_{\rightarrow}$. Section 7 adapts our results to verification of strategies for agents with imperfect information. In Section 8, we present an experimental evaluation of our algorithms, based on the drones scenario. Finally, we conclude in Section 9.

Previous version of the article. The main concepts and some of the results have appeared in a preliminary form in the conference paper [10]. The present version extends it with proper proofs, a comprehensive discussion on motivation and applicability, and a generalization of the framework to models with multi-valued transitions. We also add a case study (complete with a detailed experimental evaluation) based on a newly proposed benchmark model.

1.3. Related Work

Multi-valued interpretation of modal formulas has been used in multiple approaches to verification. The main idea was proposed by Fitting [11, 12] already more than 25 years ago. Further fundamental work on multi-valued modal logic includes, e.g., [13] where general properties of multi-valued abstractions were studied, and demonstrated on an example 9-valued bilattice.

In the 2000s, a number of works adapted the idea to verification of distributed and multi-agent systems. A variant of CTL* for models over finite quasi-boolean lattices was proposed in [3], together with a general translation scheme that reduced multi-valued model checking of CTL* specifications to the standard 2-valued case. This was later extended to multi-valued modal μ-calculus [14, 15, 16, 17], and to multi-valued modal μ-calculus with knowledge [4].[1] Our paper follows this line of work, and extends the techniques to strategic operators of ATL*. We also enrich the language with the two-valued "implication" (or comparison) and "equivalence" operators $\rightarrow$ and $\cong$, which provide: (i) the notions of material implication and biconditional, useful in specifying general properties of multi-valued models; (ii) a way of model checking "threshold properties" analogous to probabilistic temporal logics behind

[1]Note that, despite having "games" in the title, [16] was not concerned with strategic specifications. "Games" were used there only in the technical sense, to define the semantics of μ-calculus. However, [18] deals with game applications of [16].

PRISM [19]. As it turns out, the new operators require non-trivial treatment, significantly different from the previous works [3, 14, 15, 4, 16, 17].

All the above papers consider *general* multi-valued verification, i.e., the interpretation can be based on an arbitrary finite lattice. Another line of work considers model checking over specific domains of truth values, tailored to a particular class of scenarios. Model checking methods for the special case of 3-valued temporal logics were discussed in [20, 21, 22] and, recently, in [23]. Two different methods for approximating the standard two-valued semantics of ATL* under imperfect information by using three-valued ATL* are presented and analysed in [24, 25]. However, these approaches use only Kleene's three valued logic rather than general distributive lattices. Moreover, a partial algorithm for model checking two-valued perfect recall via its approximation as three-valued bounded recall is constructed in [24].

Related approaches include runtime verification, which often uses 3-valued [5] or 4-valued interpretation [6, 14] of temporal formulas. Moreover, a 4-valued semantics has been used to evaluate database queries [26, 27]. A 3-valued semantics of strategic abilities in models of perfect information was considered in [28] for verification of alternating μ-calculus, and in [29] for abilities expressed in ATL.[2] In [30], another 3-valued semantics of ATL was studied, for both perfect and imperfect information. In all those papers (i.e., [28, 29, 30]), the main aim was to verify may/must abstractions of multi-agent systems. Note that, while the agenda of our paper comes close to that of [29, 30], our semantics differs from [29] even in the 3-valued case. Moreover, our multi-valued semantics of ATL is a conservative extension of standard ATL, whereas the one in [29] is not. In contrast, the ATL variant in [30] *is* a conservative extension of the 2-valued semantics, and can be in fact considered as a very special case of our general semantics.

A quite different but related strand of research concerns real-valued logics over probabilistic models for temporal [31, 32, 33] and strategic specifications [34]. We also mention the research on probabilistic model checking of temporal and strategic logics [35, 36, 19, 37, 38, 39] that evaluates specifications in the 2-valued domain but recognizes different degrees of success and the need to aggregate them over available strategies and possible paths.

In sum, there have been approaches to general multi-valued model checking of temporal and epistemic properties, but no analogous proposals for strategic properties. Furthermore, there were proposals for multi-valued verification of strategic properties over specific (and usually very simple) sets of truth vales, but no framework that studies the same problem for arbitrary lattices. This paper fills the gap, and combines elements of both strands to obtain a general framework for multi-valued verification of ability in systems of interacting agents.

2. How to Specify Strategic Abilities

We begin by recalling the basics of two-valued alternating-time temporal logic. We also introduce our working example that will be used throughout the paper.

[2]More precisely, the semantics in [29] includes agents' indistinguishability relations in the model, but it allows for non-uniform play, thus effectively assuming that the agents have perfect information about the current state of the system while playing.

2.1. Syntax

Alternating-time temporal logic [1, 2] generalizes the branching-time temporal logic CTL* by replacing path quantifiers E, A with *strategic modalities* $\langle\!\langle A \rangle\!\rangle$. Informally, $\langle\!\langle A \rangle\!\rangle\gamma$ says that a group of agents A has a collective strategy to enforce temporal property γ. ATL* formulas can include temporal operators: X ("in the next state"), G ("always from now on"), F ("now or sometime in the future"), and U (strong "until"). Similarly to CTL* and CTL, we consider two syntactic variants of the alternating-time logic, namely ATL* and ATL. Formally, let $\mathbb{A}\mathrm{gt}$ be a finite set of agents, and $Prop$ a countable set of atomic propositions. The language of ATL* is defined as follows:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle A \rangle\!\rangle\gamma, \qquad \gamma ::= \varphi \mid \neg\gamma \mid \gamma \wedge \gamma \mid \mathsf{X}\,\gamma \mid \gamma\,\mathsf{U}\,\gamma.$$

where $A \subseteq \mathbb{A}\mathrm{gt}$ and $\mathsf{p} \in Prop$. Traditionally, only the state formulas φ are called formulas of ATL*. Derived boolean connectives and constants ($\vee, \top, \bot$) are defined as usual. "Sometime", "weak until", and "always from now on" are defined as $\mathsf{F}\,\gamma \equiv \top\,\mathsf{U}\,\gamma$, $\gamma_1\,\mathsf{W}\,\gamma_2 \equiv \neg((\neg\gamma_2)\,\mathsf{U}\,(\neg\gamma_1 \wedge \neg\gamma_2))$, and $\mathsf{G}\,\gamma \equiv \gamma\,\mathsf{W}\,\bot$. Also, we can use $[\![A]\!]\gamma \equiv \neg\langle\!\langle A \rangle\!\rangle\neg\gamma$ to express that, for each strategy of A, property γ holds on some paths.

ATL (without "star") is the syntactic variant in which strategic and temporal operators are combined into compound modalities: $\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\!\langle A \rangle\!\rangle\mathsf{X}\,\varphi \mid \langle\!\langle A \rangle\!\rangle\varphi\,\mathsf{U}\,\varphi \mid \langle\!\langle A \rangle\!\rangle\varphi\,\mathsf{W}\,\varphi$.

2.2. Models

The semantics of ATL* is typically defined over synchronous multi-agent transition systems, i.e., models where all the agents simultaneously decide on their next actions, and the combination of their choices determines the next state, see the following definition.

Definition 2.1. (CGS)
A *concurrent game structure (CGS)* is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, V \rangle$, which includes nonempty finite sets of: agents $\mathbb{A}\mathrm{gt}$, states St, actions Act, atomic propositions $Prop$, and a propositional valuation $V : St \to 2^{Prop}$. The function $d : \mathbb{A}\mathrm{gt} \times St \to 2^{Act} \setminus \{\emptyset\}$ defines the availability of actions. The (deterministic) transition function t assigns a successor state $q' = t(q, \alpha_1, \dots, \alpha_{|\mathbb{A}\mathrm{gt}|})$ to each state $q \in St$ and any tuple of actions $\alpha_i \in d(i, q)$, one per agent $i \in \mathbb{A}\mathrm{gt}$, that can be executed in q. A *pointed CGS* is a pair (M, q_0), where M is a CGS and $q_0 \in St$ is the initial state of M.

A *path* $\lambda = q_0 q_1 q_2 \dots$ in a CGS is an infinite sequence of states such that there is a transition between each q_i, q_{i+1} for each $i \geq 0$.

$\lambda[i]$ denotes the ith position on λ (starting from $i = 0$) and $\lambda[i, \infty]$ the suffix of λ starting with i.

2.3. Semantics

Given a CGS, we define the strategies and their outcomes as follows. A *perfect recall strategy* (or *IR-strategy*) for agent a is a function $s_a : St^+ \to Act$ such that $s_a(q_0 q_1 \dots q_n) \in d(a, q_n)$. A *memoryless strategy* (or *Ir-strategy*) for a is a function $s_a : St \to Act$ such that $s_a(q) \in d(a, q)$. A *collective strategy* for a group of agents $A = \{a_1, \dots, a_r\}$ is a tuple of individual strategies $s_A = \langle s_{a_1}, \dots, s_{a_r} \rangle$. Note that s_A only binds the agents in A, while agents outside A can act as they wish. The set of such

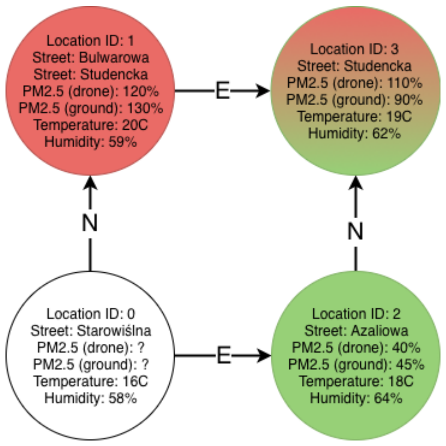


Figure 1. Map: drone navigation and measurements in an area of Cracow. Location colors indicate whether the PM2.5 readings are within or beyond the norm

strategies is denoted by Σ_A^{IR} (resp. Σ_A^{Ir}). The "outcome" function $out(q, s_A)$ returns the set of all paths that can occur when agents A execute strategy s_A from state q onward. The semantics of perfect recall ATL* is defined as follows:

$M, q \models \mathsf{p}$ iff $\mathsf{p} \in V(q)$, for $\mathsf{p} \in Prop$;

$M, q \models \neg\neg\varphi$ iff $M, q \models \varphi$, and $M, q \models \neg\varphi$ iff $M, q \not\models \varphi$ for $\varphi \neq \neg\psi$ for any ψ;

$M, q \models \varphi_1 \wedge \varphi_2$ iff $M, q \models \varphi_1$ and $M, q \models \varphi_2$;

$M, q \models \langle\!\langle A \rangle\!\rangle\gamma$ iff there is a strategy $s_A \in \Sigma_A^{IR}$ such that, for each path $\lambda \in out(q, s_A)$, we have $M, \lambda \models \gamma$.

$M, \lambda \models \varphi$ iff $M, \lambda[0] \models \varphi$;

$M, \lambda \models \neg\neg\gamma$ iff $M, \lambda \models \gamma$, and $M, \lambda \models \neg\gamma$ iff $M, \lambda \not\models \gamma$ for $\lambda \neq \neg\gamma$ for any γ;

$M, \lambda \models \gamma_1 \wedge \gamma_2$ iff $M, \lambda \models \gamma_1$ and $M, \lambda \models \gamma_2$;

$M, \lambda \models \mathsf{X}\,\gamma$ iff $M, \lambda[1, \infty] \models \gamma$; and

$M, \lambda \models \gamma_1 \,\mathsf{U}\, \gamma_2$ iff there is an $i \in \mathbb{N}_0$ such that $M, \lambda[i, \infty] \models \gamma_2$ and $M, \lambda[j, \infty] \models \gamma_1$ for all $0 \leq j < i$,

where $\mathbb{N}_0$ is the set of non-negative integers. The memoryless semantics of ATL* uses strategies of Σ_A^{Ir} rather than Σ_A^{IR}.

Example 2.2. (Drones patrolling for pollution)
Consider a team of k drones monitoring air pollution in the city of Cracow, Poland. For this scenario, we use the map shown in Figure 1. To keep the resulting CGS small, the map is a grid that only

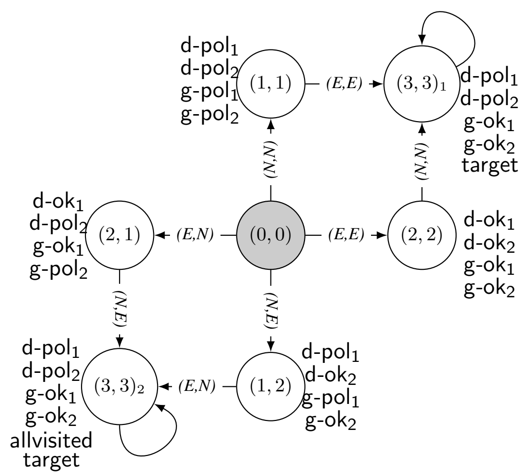


Figure 2. Model M_1: autonomous drones monitoring pollution

includes four locations, and the drones can move between the locations in one direction only (either North or East).[3] The initial location is 0, and the drones are supposed to reach location 3 (called their "target" location). Each drone has a sensor to measure the level of PM2.5 in the air, that is the rate of particles with a diameter less than 2.5 microns. The drone can also communicate with the nearest sensor on the ground that reports the PM2.5 rate on the ground level, as well as the current measurements of temperature, air pressure, and humidity. Note that some measurements may be unobtainable at some locations. In particular, no measurements are available at the start location. We also assume, for the sake of simplicity, that the environment can be viewed as stationary with respect to the movement of drones, i.e., the measurements at a location do not change throughout the patrolling mission.

Figure 2 presents a pointed concurrent game structure M_1 that models the above scenario for $k = 2$ drones. Every drone is a separate agent, with two actions available: N (fly North) and E (fly East). Due to a limited battery capacity, a drone can only visit $l = 2$ locations before the battery dies. Each state of the model includes information about the current locations of the drones, possibly distinguishing the locations that have been already visited by the team. Note that in our simple scenario the only pair of locations that can be reached by the drones via two different routes is $(3, 3)$. The corresponding two states are labeled accordingly $(3, 3)_1$ and $(3, 3)_2$.

Most of the atomic propositions refer to the pollution measurements available to a drone at its current location. That is, $\mathsf{d\text{-}pol_i}$ indicates that the ith drone registers pollution; more precisely: the sensor of drone i reports that the level of PM2.5 exceeds the norm. Similarly, $\mathsf{d\text{-}ok_i}$ indicates that the sensor of drone i reports a level of PM2.5 within the norm. Moreover, $\mathsf{g\text{-}pol_i}$ (resp. $\mathsf{g\text{-}ok_i}$) says that the ground sensor nearest to drone i reports a level of PM2.5 exceeding the norm (resp. within the norm). Proposition target indicates states where all drones have reached the target location. Finally,

[3]A more complex map will be used for the experiments in Section 8.

proposition $\mathsf{allvisited}$ labels the states where the team has already visited all locations in the map: in case of M_1, the only such state is $(3,3)_2$. □

Example 2.3. (Drones, ctd.)
For the model in Example 2.2, we have, for instance, $M_1,(0,0) \models \langle\langle 1\rangle\rangle \mathsf{F}\, \mathsf{d\text{-}pol}_1$: drone 1 has a strategy ensuring that its sensor will eventually register pollution. The stratety itself is simple: when in the state $(0,0)$ fly North. Moreover, $M_1,(0,0) \models \langle\langle 1\rangle\rangle \mathsf{F}\, \mathsf{d\text{-}ok}_1$: it also has a strategy for reaching a location where it registers no pollution. This time the simplest strategy is to fly East. In fact, $M_1,(0,0) \models \langle\langle 1\rangle\rangle(\mathsf{F}\, \mathsf{d\text{-}pol}_1 \wedge \mathsf{F}\, \mathsf{d\text{-}ok}_1)$: there is a single strategy for achieving both goals. The drone needs to fly East first, and then fly North.

Furthermore, no drone can assure on its own that all of the locations will eventually be visited: $M_1,(0,0) \models \neg\langle\langle 1\rangle\rangle \mathsf{F}\, \mathsf{allvisited} \wedge \neg\langle\langle 2\rangle\rangle \mathsf{F}\, \mathsf{allvisited}$. This can only be ensured if both drones cooperate: $M_1,(0,0) \models \langle\langle 1,2\rangle\rangle \mathsf{F}\, \mathsf{allvisited}$. On the other hand, the drones are bound to end up at the target location, no matter what they decide to do: $M_1,(0,0) \models \langle\langle \emptyset\rangle\rangle \mathsf{F}\, \mathsf{target}$. □

3. Multi-Valued Domains of Interpretation

As the formulas of our multi-valued version of ATL* will be interpreted in distributive lattices of truth values [40], we recall the relevant notions and results in this section.

3.1. Lattices of Truth Values

Definition 3.1. A *lattice* is a partially ordered set $\mathcal{L} = (\mathrm{L}, \leq)$, where every pair of elements $x, y \in \mathrm{L}$ has the greatest lower bound (called *meet* and denoted by $x \sqcap y$) and the least upper bound (called *join* and denoted by $x \sqcup y$).

Note that the meet and join of any $x, y \in \mathrm{L}$ are uniquely determined due to the antisymmetry of $\leq$.

In what follows, we only consider finite lattices. We denote the least and the greatest elements of $\mathcal{L}$ by $\bot, \top$, respectively. Also, we write (i) $x_1 < x_2$ iff $x_1 \leq x_2$ and $x_1 \neq x_2$, and (ii) $x_1 \bowtie x_2$ iff neither $x_1 \leq x_2$ nor $x_2 \leq x_1$. Moreover, let:

- $\uparrow x = \{y \in L \mid x \leq y\}$ denote the upward closure of x, and
- $\downarrow x = \{y \in L \mid y \leq x\}$ denote the downward closure of x.

A lattice $\mathcal{L}' = (\mathrm{L}', \leq')$ is a sublattice of a lattice $\mathcal{L} = (\mathrm{L}, \leq)$ if $\mathrm{L}' \subseteq \mathrm{L}$ and $\leq' = \leq \cap (\mathrm{L}' \times \mathrm{L}')$.

Definition 3.2. A lattice $\mathcal{L} = (\mathrm{L}, \leq)$ is *distributive* if, for any $x, y, z \in \mathrm{L}$, the following two conditions hold: (i) $z \sqcup (x \sqcap y) = (z \sqcup x) \sqcap (z \sqcup y)$, (ii) $z \sqcap (x \sqcup y) = (z \sqcap x) \sqcup (z \sqcap y)$.

Note that, for any lattice $\mathcal{L}$, the operations $\sqcup$ and $\sqcap$ are associative.

Example 3.3. (Some useful lattices)
Figure 3 presents five distributive lattices with applicability motivated by clear practical intuitions. The total order **3** is the most popular lattice in multi-valued verification. The intuition is simple: $\top$

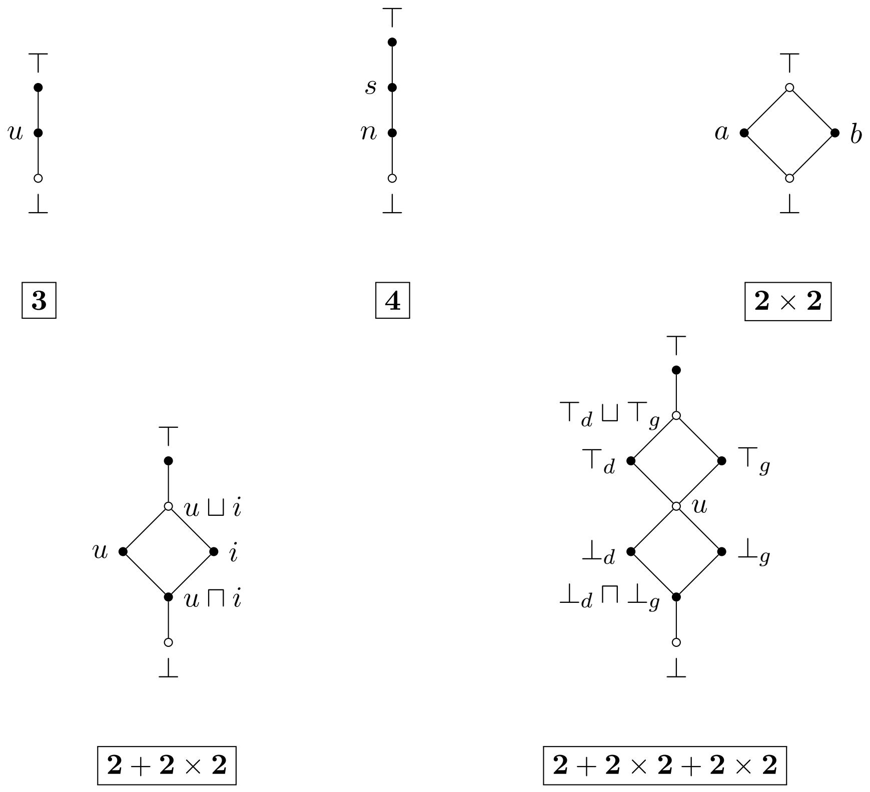


Figure 3. Distributive lattices and their join-irreducible elements

stands for absolute truth, $\bot$ for absolute falsity, and u can be read as "unknown" or "undefined." That is, when a formula φ is assigned the value u, this indicates that the statement represented by φ cannot be conclusively evaluated (its truth value – in the classical sense – cannot be determined, or even does not exist at the moment). The lattice is very often used in model checking approaches based on abstraction, with u assigned to formulas for which the verification has proved inconclusive.

The total order $\mathbf{4}$ allows for representing graded uncertainty. For instance, in 4-valued approaches to runtime monitoring, s is interpreted as "still possibly true," and n stands for "not proved false yet." In evidence-based reasoning, the values can correspond to situations when there is much (respectively, little) evidence supporting φ. The lattice can be generalized to the k-valued linear order $\mathbf{k}$, useful in scenarios where the amount of positive/negative evidence is weakly indicative for the truth of a statement. Consider a corpus of data coming from event logs that support or reject proposition p. Then, the logical value of p can be, e.g., defined as the difference between the amounts of positive and negative evidence. A more sophisticated, partially ordered lattice could also involve the number of conflicts and the support set size.

The partial order $\mathbf{2} \times \mathbf{2}$ can be used to interpret statements with evidence coming from two different, possibly disagreeing sources A and B. Then, the value a can be read as "true according to source A, but not necessarily according to B," and analogously for b. The dual interpretation is also possible,

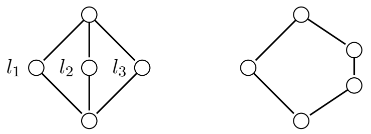


Figure 4. Non-distributive lattices M5 and N5

i.e., we can use a to represent "false according to source A, but not necessarily according to B", and likewise for b. Actually, these two interpretations correspond to two different choices of the set $\mathcal{D}$ of so-called *designated values*, i.e., values corresponding to truth in classical logic and representing satisfaction of a formula. Namely, the first interpretation corresponds to $\mathcal{D} = \{a, \top\}$, and the second — to $\mathcal{D} = \{b, \top\}$. Another useful lattice, $\mathbf{2} + \mathbf{2} \times \mathbf{2}$, allows for a natural representation of both uncertainty and disagreement. It provides truth values for statements with inconsistent evidence (i) and inconclusive evidence (u). Combinations of inconsistency and uncertainty can be easily obtained by join and meet ($u \sqcup i$, $u \sqcap i$)

For a multi-valued interpretation of the formulas in the drone model, we propose another lattice denoted by $\mathbf{2} + \mathbf{2} \times \mathbf{2} + \mathbf{2} \times \mathbf{2}$. The lattice combines two instances of $\mathbf{2} \times \mathbf{2}$: one representing incomplete evidence of truth, and the other incomplete evidence of falsity. The idea is that a drone i will evaluate the truth of the proposition pol_i ("according to i, its current location is polluted"), based on the readings from its own sensor and the nearest ground sensor. Besides that, the lattice includes explicit nodes for the maximal and minimal elements, similarly to the previous lattice. This gives us the following basic truth values and their interpretation:

- $\top$: both readings indicate presence of pollution at the location,
- $\top_d$: reading from the drone sensor indicates pollution, while the ground sensor indicates no pollution or provides no reading,
- $\top_g$: reading from the ground sensor shows pollution; absent or negative reading from the drone,
- u: there are no readings, neither from the ground nor from the drone,
- $\bot_d$: drone sensor indicates no pollution; there is no reading from the ground,
- $\bot_g$: ground sensor indicates no pollution; no reading from the drone,
- $\bot$: no pollution (both readings are within the norm). $\square$

Not every lattice is distributive, as shown in Figure 4. However, distributive lattices have a very simple characterization: a lattice is distributive iff it contains neither M5 nor N5 as a sublattice [40].

Remark 3.4. (Quasi-boolean lattices and De Morgan algebras)
The operations of join and meet are natural semantic counterparts of disjunction and conjunction. Some multi-valued approaches also add the *complement* operation $\sim$ to the lattice, as the semantic counterpart of multi-valued negation. A lattice with complement is usually called *quasi-Boolean*, and when distributive it is referred to as a *De Morgan algebra*. The most popular case is the lattice

underlying 3-valued Kleene logic, used e.g. in [30, 25]. However, the choice of a generic complement to suit any lattice is problematic from the conceptual point of view. For instance, what should be the "opposite" value to i in the lattice $\mathbf{2}+\mathbf{2}\times\mathbf{2}$? In other words, if statement φ is assessed as "inconsistent", what should be the evaluation of "not φ?". There is no uniform answer to that question, so in a general approach paper, like ours, a good strategy is to avoid the negation as much as possible. This is why instead of negation we have chosen to use here another non-monotonic connective: two-valued implication representing the lattice order, which is useful in all practical cases when we want to compare the truth values of two formulas.

3.2. Join-Irreducible Elements

Definition 3.5. Let $\mathcal{L} = (\mathrm{L}, \leq)$ be a lattice. An element $\ell \in \mathrm{L}$ is called *join-irreducible* iff $\ell \neq \bot$ and, for any $x, y \in \mathrm{L}$, if $\ell = x \sqcup y$, then either $\ell = x$ or $\ell = y$. The set of all join-irreducible elements of $\mathcal{L}$ is denoted by $\mathcal{JI}(\mathcal{L})$.

It is well known [41] that every element $x \neq \bot$ of a finite distributive lattice can be uniquely decomposed into the join of all join-irreducible elements in its downward closure, i.e.

$$x = \bigsqcup(\mathcal{JI}(\mathcal{L}) \cap \downarrow x) \qquad (1)$$

Example 3.6. The join-irreducible elements of the distributive lattices in Figure 3 are marked with black dots. All other ones can be decomposed into the join of some join-irreducible elements. □

We will use the characterization (1) to define translations from multi-valued to standard model checking through the following theorem.

Theorem 3.7. ([4])
Let $\mathcal{L}$ be a finite distributive lattice, and let $\ell \in \mathcal{JI}(\mathcal{L})$. Then the *threshold function* $f_\ell : \mathrm{L} \longrightarrow \{\bot, \top\}$, defined by

$$f_\ell(x) = \begin{cases} \top & \text{if } x \geq \ell \\ \bot & \text{otherwise} \end{cases}$$

preserves arbitrary bounds, i.e., for an arbitrary set of indices I, we have:

$$f_\ell(\bigsqcap_{i\in I} x_i) = \bigsqcap_{i\in I} f_\ell(x_i) \qquad \textit{and} \qquad f_\ell(\bigsqcup_{i\in I} x_i) = \bigsqcup_{i\in I} f_\ell(x_i). \qquad (2)$$

Remark 3.8. The above does not hold for lattices which are not distributive. To see this, consider the element ℓ_1 of lattice **M5**, which is join-irreducible. However, f_{ℓ_1} does not preserve the upper bounds, as $f_{\ell_1}(\ell_2 \sqcup \ell_3) = f_{\ell_1}(\top) = \top$ whereas $f_{\ell_1}(\ell_2) \sqcup f_{\ell_1}(\ell_3) = \bot \sqcup \bot = \bot$.

4. Multi-Valued Strategic Logic mv-ATL$^*_\rightarrow$

In this section we extend the syntax and semantics of ATL* to allow for multi-valued reasoning. That is, we propose a variant of ATL* where formulas are interpreted in an arbitrary lattice $\mathcal{L} = (\mathrm{L}, \leq)$.

It is sometimes useful to refer to logical values in the object language. To enable this, we assume that a natural interpretation of suitable constants is given, in the following way.

Definition 4.1. (Interpreted lattice)
Let $\mathcal{C}$ be a countable set of symbols. An interpreted lattice over $\mathcal{C}$ is a triple $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$, where $(\mathrm{L}, \leq)$ is a lattice, and $\sigma : \mathcal{C} \to \mathrm{L}$ is an interpretation of the symbols in $\mathcal{C}$ as truth values in L.

To explicitly show the connection between the named truth values and their names in $\mathcal{C}$, for any interpreted lattice $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$ and any truth value $x \in \sigma(\mathcal{C})$, we will use the notation $\boxed{x}$ to denote an arbitrarily selected symbol $c \in \mathcal{C}$ such that $\sigma(c) = x$. We do not make any specific assumptions about σ. In particular, we do not assume that σ must be surjective, as in many situations only some truth values in L need to be referred to in formulas. However, in all the examples that follow in this paper, σ actually *is* a bijection, since every truth value used there has a specific purpose. Thus, for all our examples, it holds that $\mathcal{C} = \{\boxed{l} \mid l \in \mathrm{L}\}$, and $\sigma(\boxed{x}) = x$ for all $x \in \sigma(\mathcal{C})$.

4.1. Syntax

Since, as explained in Remark 3.4, multi-valued negation can be problematic from the conceptual viewpoint, we will use instead the binary implication operator $\to$ corresponding to the lattice order. Our implication operator is similar to the well-known implication of many-valued Goedel-Dummet logic, and in general to relevant implication or residuum of lattice meet, of which the former is just a special case. However, our implication is a two-valued operator, which makes it better suited for proof system purposes. As it can be used for comparing truth values of formulas, it is of obvious practical importance in many applications, where we are mainly interested in ascertaining whether the truth value of a given formula φ is greater than the logical value of some other formula ψ (see Example 4.3 below for more explanations and illustration). Moreover, in case of two-valued logic, classical negation can be expressed using $\to$ and the constant representing $\bot$.

To increase the expressive power of the language, we also allow for the use of symbols in $\mathcal{C}$.

The resulting logic is called mv-ATL$^*_\to$ and has the following syntax:

$$\varphi ::= c \mid \mathsf{p} \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \to \varphi \mid \langle\!\langle A \rangle\!\rangle \gamma \mid [\![A]\!] \gamma,$$
$$\gamma ::= \varphi \mid \gamma \wedge \gamma \mid \gamma \vee \gamma \mid \mathsf{X}\, \gamma \mid \gamma\, \mathsf{U}\, \gamma \mid \gamma\, \mathsf{W}\, \gamma.$$

where $\mathsf{p} \in \mathit{Prop}$, $A \subseteq \mathbb{A}\mathrm{gt}$, and $c \in \mathcal{C}$, with Prop being a countable set of atomic propositions, and $\mathcal{C}$ a countable set of constants.

In what follows, by an *implication formula* we mean any formula of the form $\varphi_1 \to \varphi_2$. Additionally, we define an equivalence formula as $\varphi_1 \cong \varphi_2 = (\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$.

The sublogic of mv-ATL$^*_\to$ without the implication operator will be denoted by mv-ATL*.

4.2. Semantics

The semantics of mv-ATL$^*_\to$ is defined over concurrent game structures with multi-valued interpretation of atomic propositions.

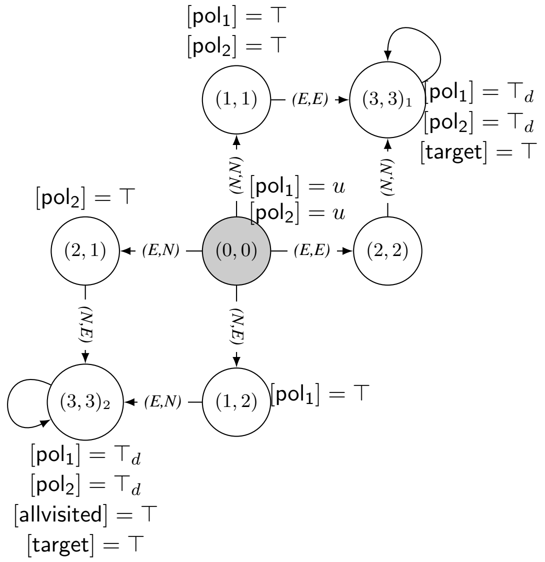


Figure 5. Multi-valued model M_2 for the drone scenario.

Definition 4.2. (Multi-valued CGS)
Let $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$ be an interpreted lattice. A *multi-valued concurrent game structure (mv-CGS)* over $\mathcal{L}^+$ is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, V, \mathcal{L}^+\rangle$, where $\mathbb{A}\mathrm{gt}$, St, Act, d, t, $Prop$ are as before, and $V : Prop \times St \to \mathrm{L}$ assigns at any state all atomic propositions with truth values from the logical domain L.

Example 4.3. (Drones ctd.)
A multi-valued model of the drone scenario is presented in Figure 5. To evaluate atomic propositions and their negations, we use the lattice $\mathbf{2} + \mathbf{2} \times \mathbf{2} + \mathbf{2} \times \mathbf{2}$ introduced in Section 3.1. Each proposition $\mathsf{pol_i}$, $i = 1, \ldots, k$, refers to the level of pollution from the viewpoint of drone i, that is, given by the available measurements at the current location of the drone. Whenever a proposition evaluates to $\perp$, we omit that valuation from the picture.

□

Logical operators can often be naturally interpreted as either maximizers or minimizers of the truth values. For example, disjunction ($\varphi \vee \psi$) can be understood as a maximizer ("the most that we can hope to make of either φ or ψ"), and conjunction as a minimizer ("the least that we can guarantee for both φ and ψ"). This extends to existential quantification (maximizing) and universal quantification (minimizing) over paths, strategies, moments in time, etc. Formally, let $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, V, \mathcal{L}^+\rangle$ be an mv-CGS over $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$. The valuation function $[\cdot]$ is given as below. We sometimes use $\bigsqcap_X\{Y\}$ as a shorthand for $\bigsqcap\{Y \mid X\}$, and similarly for the supremum. For any $q \in St$ and any path λ in M, we define:

$[c]_{M,q} = \sigma(c) \quad$ for $c \in \mathcal{C}$;

$[\mathsf{p}]_{M,q} = V(\mathsf{p}, q) \quad$ for $\mathsf{p} \in \mathit{Prop}$;

$[\varphi_1 \wedge \varphi_2]_{M,q} = [\varphi_1]_{M,q} \sqcap [\varphi_2]_{M,q}$;

$[\varphi_1 \vee \varphi_2]_{M,q} = [\varphi_1]_{M,q} \sqcup [\varphi_2]_{M,q}$;

$[\gamma_1 \wedge \gamma_2]_{M,\lambda} = [\gamma_1]_{M,\lambda} \sqcap [\gamma_2]_{M,\lambda}$ and $[\gamma_1 \vee \gamma_2]_{M,\lambda} = [\gamma_1]_{M,\lambda} \sqcup [\gamma_2]_{M,\lambda}$;

$[\varphi]_{M,\lambda} = [\varphi]_{M,\lambda[0]}$;

$[\mathsf{X}\,\gamma]_{M,\lambda} = [\gamma]_{M,\lambda[1..\infty]}$;

$[\gamma_1 \,\mathsf{U}\, \gamma_2]_{M,\lambda} = \bigsqcup_{i \in \mathbb{N}_0} \bigsqcap_{0 \le j < i} \{[\gamma_2]_{M,\lambda[i..\infty]} \sqcap [\gamma_1]_{M,\lambda[j..\infty]}\}$;

$[\gamma_1 \,\mathsf{W}\, \gamma_2]_{M,\lambda} = \bigsqcap_{i \in \mathbb{N}_0} \{[\gamma_1]_{M,\lambda[i..\infty]}\} \sqcup \bigsqcup_{i \in \mathbb{N}_0} \bigsqcap_{0 \le j < i} \{[\gamma_2]_{M,\lambda[i..\infty]} \sqcap [\gamma_1]_{M,\lambda[j..\infty]}\}$;

$[\langle\!\langle A \rangle\!\rangle \gamma]_{M,q} = \bigsqcup_{s_A \in \Sigma_A} \bigsqcap_{\lambda \in out(q,s_A)} \{[\gamma]_{M,\lambda}\}$;

$[[\![A]\!]\gamma]_{M,q} = \bigsqcap_{s_A \in \Sigma_A} \bigsqcup_{\lambda \in out(q,s_A)} \{[\gamma]_{M,\lambda}\}$;

$[\varphi_1 \to \varphi_2]_{M,q} = \top$ if $[\varphi_1]_{M,q} \le [\varphi_2]_{M,q}$ and $\bot$ otherwise.

It is worth noting that our implication operator differs from the well-known residue of lattice meet in being two-valued — which makes it better suited for use in any proof system, and more intuitive in specification of many requirements.

The semantics of the two "until" operators demands a more detailed explanation. The computation of $[\gamma_1 \,\mathsf{U}\, \gamma_2]_{M,\lambda}$ seeks to achieve a position i on path λ, for which the value of γ_2 at $\lambda[i]$, and the values of γ_1 at all the points preceding $\lambda[i]$, are guaranteed maximal. The semantics of $\gamma_1 \,\mathsf{W}\, \gamma_2$ is based on the well-known unfolding $\gamma_1 \,\mathsf{W}\, \gamma_2 \equiv (\mathsf{G}\,\gamma_1) \vee (\gamma_1 \,\mathsf{U}\, \gamma_2)$, transformed here to a multi-valued interpretation. Note also that in case of the derived temporal operators "sometime" and "always" the semantic rules reduce to:

$[\mathsf{F}\,\gamma]_{M,\lambda} = \bigsqcup_{i \in \mathbb{N}} [\gamma]_{M,\lambda[i..\infty]}$;

$[\mathsf{G}\,\gamma]_{M,\lambda} = \bigsqcap_{i \in \mathbb{N}} [\gamma]_{M,\lambda[i..\infty]}$.

Thus, for instance, the formula $\langle\!\langle A \rangle\!\rangle \mathsf{F}\,\mathsf{pol}$ can be read as: "the maximal level of pollution readings that A can guarantee to reach." Clearly, such statements do not always submit to intuitive understanding, in particular when nested strategic operators are used. Because of that, we will stick to simple formulas in our working examples, that is, ones that are relatively easy to read.

Example 4.4. (Drones ctd.)
For the model in Figure 5, we have $[\langle\!\langle 1 \rangle\!\rangle \mathsf{F}\,\mathsf{pol}_1]_{M_2,(0,0)} = \top$: there is a strategy for drone 1 to surely detect pollution (the strategy being to fly North in state $(0,0)$, and then East in $(1,1)$ or $(1,2)$). Similarly for the other drone we have $[\langle\!\langle 2 \rangle\!\rangle \mathsf{F}\,\mathsf{pol}_2]_{M_2,(0,0)} = \top$ (the same strategy, but now executed

by drone 2). On the other hand, $[\langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1]_{M_2,(0,0)} = u$: the maximal *guaranteed* level of detection throughout the mission is u (obtained by the same strategy again). This means that if drone 1 wants to maximize its detection level, the best it can achieve is to keep it consistently at the level of "uncertain" or higher. Finally,

$$[\langle\!\langle 1,2\rangle\!\rangle \mathsf{F}\, (\mathsf{target} \wedge \mathsf{allvisited} \wedge (\mathsf{pol}_1 \vee \mathsf{pol}_2))]_{M_2,(0,0)} = \top_d.$$

That is, if the drones cooperate, and their goal is to reach the target, visit all the locations on the way, and at the end get the pollution detected by at least one of them, then their degree of success is $\top_d$ (pollution indicated by the drone sensor but not by the ground sensor). □

The logical constants we have introduced are especially useful in implication formulas, as the subsequent example demonstrates.

Example 4.5. (Implication formulas)
The "implication" operator provides several interesting specification patterns. For instance, it allows for specifications that are accepted when the "strength" of a property reaches a given threshold, similarly to the probabilistic approaches of [19, 39]. As an example, the formula $\boxed{u} \rightarrow \langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1$ can be used to specify that the truth value of $\langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1$ is at least u (intuitively: there is no evidence that the formula is false). It is easy to see that the formula is true in the model of Figure 5; formally: $[\boxed{u} \rightarrow \langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1]_{M_2,(0,0)} = \top$. Naturally, any stronger requirement on the value of $\langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1$ evaluates to "false," e.g., $[\boxed{\top} \rightarrow \langle\!\langle 1\rangle\!\rangle \mathsf{G}\, \mathsf{pol}_1]_{M_2,(0,0)} = \bot$.

Moreover, the formula $\langle\!\langle 1\rangle\!\rangle \mathsf{F}\, \mathsf{pol}_1 \rightarrow \langle\!\langle 2\rangle\!\rangle \mathsf{F}\, \mathsf{pol}_2$ says that the ability of drone 2 to spot pollution is at least as good as that of drone 2 (the formula evaluates to $\top$ in $M_2,(0,0)$). Finally, $\langle\!\langle 1\rangle\!\rangle \mathsf{F}\, (\mathsf{pol}_1 \cong \boxed{\top_g})$ says that the first drone has a strategy to ensure that it will reach a location where only the ground sensor indicates pollution. Clearly, the last formula evaluates to $\bot$ in $M_2,(0,0)$. □

We note that most approaches to general multi-valued model checking of temporal specifications [3, 14, 4, 16] allow also for *multi-valued transitions* in the models, analogous to probabilistic transitions in Markov chains and Markov Decision Processes. That is, transitions can be assigned "weights" drawn from the same algebra $\mathcal{L}$. Similarly, most 3-valued approaches to temporal abstraction and model checking implicitly assume 3-valued transitions by distinguishing between *may* and *must* transitions [42, 20, 21, 22]. However, the two approaches differ in how such transitions affect the semantics of formulas with universal quantification (such as "for all paths γ"). In the general multi-valued approach, the "weaker" the path is, the more it decreases the value of the formula. In the 3-valued approach, "weaker" paths have less influence on the overall value. We do not engage in this discussion here, and leave a proper treatment of multi-valued transitions until Section 6.

4.3. Truth Levels

We assume that $\top$ is a single designated value, standing for full logical truth. In consequence, the truth and validity of formulas can be defined in a straightforward way as follows:

Definition 4.6. (Validity levels)
Let M be mv-CGS, q a state in M, and φ a state formula of mv-ATL$^*_{\rightarrow}$. Then:

- φ is *true in* M, q (written $M, q \models \varphi$) iff $[\varphi]_{M,q} = \top$.
- φ is *valid in* M (written $M \models \varphi$) iff φ is true in every state of M.
- φ is *valid* (written $\models \varphi$) iff φ is valid in every mv-CGS M.
- Additionally, for a path formula γ, we can say that γ holds on run λ in a mv-CGS M (written $M, \lambda \models \gamma$) iff $[\gamma]_{M,\lambda} = \top$.

We now show that mv-ATL$^*_\rightarrow$ agrees with standard ATL* on 2-valued models, unlike the 3-valued version of ATL* from [29].

Theorem 4.7. The logic mv-ATL$^*_\rightarrow$ is a conservative extension of ATL*, i.e., every CGS M for ATL* can be identified with an mv-CGS M' for mv-ATL$^*_\rightarrow$ over the lattice $\mathbf{2}$ such that, for any ATL* formula φ and any state (path) ξ, we have $M', \xi \models \varphi$ iff $M, \xi \models \varphi$.

Proof:
For any CGS $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, V\rangle$ for ATL*, let $M' = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, V', \mathbf{2}\rangle$, where: (i) $\mathbf{2} = (\{\bot, \top\}, \leq, \sigma)$ is an interpreted classical lattice of two truth values over $\mathcal{C} = \{\boxed{\bot}, \boxed{\top}\}$ and $\sigma(\boxed{l}) = l$ for any $l \in \{\bot, \top\}$; (ii) $V'(p, q) = \top$ if $q \in V(p)$ and $\bot$ otherwise. Then M' is an mv-CGS for mv-ATL$^*_\rightarrow$, and an easy check shows that, for any ATL* formula φ and any state (path) ξ, it indeed obtains $M', \xi \models \varphi$ iff $M, \xi \models \varphi$. □

5. Model Checking mv-ATL$^*_\rightarrow$

Given an mv-CGS M, a state q in M, and an mv-ATL$^*_\rightarrow$ formula φ, the model checking problem consists in computing the value of $[\varphi]_{M,q}$. This can be done in two ways: either by using a dedicated algorithm, or through an efficient reduction to the ”classical”, 2-valued version of model checking. The latter option has many advantages. First and foremost, it allows us to benefit from the ongoing developments in 2-valued model checking, including symbolic model checking techniques, heuristics, model reduction techniques, etc. In this section, we show how model checking of mv-ATL$^*_\rightarrow$ can be reduced to the 2-valued variant of this problem. Since a basic result underlying such reduction holds for distributive lattices only, throughout the section we assume that all lattices under consideration are distributive, unless stated to the contrary.

We emphasize again that, while multi-valued model checking typically provides a *conceptual* approximation of classical verification, the results in this section are about something else. Here, we look for a *technical* reduction from multi-valued to two-valued model checking, with the sole purpose of facilitating the verification process.

5.1. From Multi-Valued Model Checking to Classical Model Checking

It is well known that model checking multi-valued temporal logics can be reduced to classical, 2-valued model checking [3, 14, 15, 4]. The reduction is of one-to-many type, i.e., a single instance of multi-valued model checking translates to linearly many instances of classical model checking. The key result in this respect is [3, Theorem 1]. It proposes a method for "clustering" the truth values from lattice $\mathcal{L}$ into a smaller lattice $\mathcal{L}'$ in such a way that the outcome of model checking is preserved. We will now show that the analogue of that theorem holds for mv-ATL*, i.e., the sublanguage of mv-ATL$^*_\rightarrow$ without the $\rightarrow$ operator.

Definition 5.1. 1. By a lattice reduction triple (LRT) we mean a triple $(\mathcal{L}, \mathcal{L}_f, f)$, where $\mathcal{L} = (\mathrm{L}, \leq)$ is an arbitrary finite lattice, $\mathcal{L}_f = (\mathrm{L}_f, \leq_f)$ its sublattice, and $f : \mathrm{L} \rightarrow \mathrm{L}_f$ a homomorphism — a mapping which preserves arbitrary bounds in $\mathcal{L}$, i.e. such that

$$f(\bigsqcap_{i\in I} x_i) = \bigsqcap_{i\in I} f(x_i) \qquad \textit{and} \qquad f(\bigsqcup_{i\in I} x_i) = \bigsqcup_{i\in I} f(x_i) \tag{3}$$

for an arbitrary set of indices I.

2. Given an LRT $(\mathcal{L}, \mathcal{L}_f, f)$ and an mv-CGS $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, \mathcal{V}, \mathcal{L}^+\rangle$ over an interpreted lattice $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$, by the *reduction of M to $\mathcal{L}_f$ via f* we mean the mv-CGS $f(M) = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, \mathcal{V}_f, (\mathrm{L}_f, \leq_f, \sigma_f)\rangle$, where
   (a) $\sigma_f(c) = f(\sigma(c))$ for any $c \in \mathcal{C}$, and
   (b) $\mathcal{V}_f(p, q) = f(\mathcal{V}(p, q))$ for any $q \in St$ and $p \in Prop$.

Definition 5.2. For any LRT $(\mathcal{L}, \mathcal{L}_f, f)$ and any model M over L, by the *translation condition for LRT and formula φ we mean the relationship*

$$[\varphi]_{M,\xi} \in f^{-1}(x) \qquad \textit{iff} \qquad [\varphi]_{f(M),\xi} = x \tag{4}$$

holding for any state (respectively, path) ξ.

The proof of the theorem follows easily from the key result given below:

Lemma 5.3. Let a state or path formula φ be such that

$$[\varphi]_{M,\xi} = \bigsqcup_{i\in I} \bigsqcap_{j_i\in J_i} [\varphi_{j_i}]_{M,\xi_{j_i}} \quad \text{or} \quad [\varphi]_{M,\xi} = \bigsqcap_{i\in I} \bigsqcup_{j_i\in J_i} [\varphi_{j_i}]_{M,\xi_{j_i}}$$

for any mv-CGS M, any states and/or paths ξ, ξ_{j_i} of M, any countable sets I, J_i, and state (resp. path) formulas of mv-ATL* φ_{j_i} for $j_i \in J_i, i \in I$, such that all φ_{j_i}'s satisfy translation condition (4). Then φ satisfies the translation condition too.

Proof:
We consider the case $[\varphi]_{M,\iota} = \bigsqcup_{i\in I} \bigsqcap_{j_i\in J_i} [\varphi_{j_i}]_{M,\iota_{j_i}}$; the other case follows by symmetry. As f preserves the bounds, by the assumption on φ we have $f([\varphi]_{M,\iota}) = \bigsqcup_{i\in I} \bigsqcap_{j_i\in J_i} f([\varphi_{j_i}]_{M,\iota_{j_i}})$. Each φ_{j_i} satisfies (4), so $f([\varphi]_{M,\iota}) = \bigsqcup_{i\in I} \bigsqcap_{j_i\in J_i} [\varphi_{j_i}]_{M_f,\iota_{j_i}} = [\varphi]_{M_f,\iota}$, whence $[\varphi]_{M_f,\iota} = x$ iff $[\varphi]_{M,\iota} \in f^{-1}(x)$, and (4) holds for φ. □

Now, we can formulate the reduction theorem.

Theorem 5.4. (Reduction theorem)
Let $\mathcal{L} = (\mathrm{L}, \leq)$ be an arbitrary finite lattice, and $(\mathcal{L}, \mathcal{L}_f, f)$ an LRT. Further, let M be an mv-CGS over an interpreted lattice $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$ with $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, \mathcal{V}, \mathcal{L}^+ \rangle$, and let $f(M) = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, Prop, \mathcal{V}_f, (\mathrm{L}_f, \leq_f, \sigma_f) \rangle$ be the image of M under f. Then, for any state (respectively, path) formula φ of mv-ATL* over $\mathcal{L}$ and any state (respectively, path) ξ, the following condition is satisfied:

$$[\varphi]_{M,\xi} \in f^{-1}(x) \qquad \text{iff} \qquad [\varphi]_{f(M),\xi} = x \tag{5}$$

Proof:
We use induction on the length of a formula. Equation (5) clearly holds for propositional variables and their negations. Assume it holds for formulas of length at most k, and consider formula φ of length $k + 1$. Then we have the following cases:

(a) $\varphi = \varphi_1 \wedge \varphi_2$ or $\varphi = \varphi_1 \vee \varphi_2$, where each φ_i is a formula of length at most k. Then $[\varphi_1]_{M,q} = [\varphi_1]_{M,q} \sqcap [\varphi_2]_{M,q}$ or $[\varphi_1]_{M,q} = [\varphi_1]_{M,q} \sqcup [\varphi_2]_{M,q}$, respectively, where φ_1, φ_2 satisfy (5). As $[\varphi_1]_{M,q}$ is in one of the two dual forms prescribed by Lemma 5.3 for $I = \{1\}$ and $J_1 = \{1, 2\}$, by that lemma, φ must satisfy (5), too.

(b) $\gamma = \gamma_1 \wedge \gamma_2$ or $\gamma = \gamma_1 \vee \gamma_2$ — analogously to (a).

(c) $\varphi = \mathrm{X}\psi$, where ψ is of length at most k. Then $[\mathsf{X}\,\gamma]_{M,\lambda} = [\gamma]_{M,\lambda[1..\infty]}$, and as ψ satisfies (5) by inductive hypothesis, so obviously does φ. The reasoning is similar to (a).

(d) $\varphi = \mathrm{U}\psi$, where $[\gamma_1 \,\mathsf{U}\, \gamma_2]_{M,\lambda} = \bigsqcup_{i \in \mathbb{N}_0} \bigsqcap_{0 \leq j < i} \{[\gamma_2]_{M,\lambda[i..\infty]} \sqcap [\gamma_1]_{M,\lambda[j..\infty]}\}$;
Since the operator U corresponds to a combination of finite and infinite lower and upper bounds applied to values of formulas of length at most k for which (5) holds, then by Lemma 5.3 Equation (5) must hold for φ too.

(e) $\varphi = \mathrm{W}\psi$, where
$$[\gamma_1 \,\mathsf{W}\, \gamma_2]_{M,\lambda} = \bigsqcap_{i \in \mathbb{N}_0} \{[\gamma_1]_{M,\lambda[i..\infty]}\} \sqcup \bigsqcup_{i \in \mathbb{N}_0} \bigsqcap_{0 \leq j < i} \{[\gamma_2]_{M,\lambda[i..\infty]} \sqcap [\gamma_1]_{M,\lambda[j..\infty]}\};$$
Since the operator W corresponds to a combination of finite and infinite lower and upper bounds applied to values of formulas of length at most k for which (5) holds, then by Lemma 5.3 Equation (5) must hold for φ too.

(f) $\varphi = \langle\!\langle A \rangle\!\rangle \gamma$, where γ is of length at most k. Then γ satisfies (5) by inductive hypothesis, and as $[\langle\!\langle A \rangle\!\rangle \gamma]_{M,q} = \bigsqcup_{s_A \in \Sigma_A} \bigsqcap_{\lambda \in out(q,s_A)} [\gamma]_{M,\lambda}$, then φ satisfies (5) by Lemma 5.3.

(g) $\varphi = [\![A]\!]\gamma$ — analogously to (e).

□

Note that the mapping f can be seen as an abstraction of truth values similar to the well-known technique of *state abstraction* [43, 44]. That is, we can view each value $x \in \mathrm{L}_f$ as an *abstract truth value* corresponding to the subset $f^{-1}(x)$ of the original truth values in L. Clearly, those subsets partition L into equivalence classes. Theorem 5.4 says that if f satisfies conditions (3), then model checking in the abstract model M_f yields the equivalence class corresponding to the output of the original model checking problem in the concrete model M.

How can we use Theorem 5.4 to reduce multi-valued model checking to the 2-valued case? Recall the threshold functions $f_\ell : \mathrm{L} \longrightarrow \{\bot, \top\}$, defined by

$$f_\ell(x) = \begin{cases} \top & \text{if } x \geq \ell \\ \bot & \text{otherwise} \end{cases}$$

We already stated in Theorem 3.7 that those functions preserve bounds. The following is an immediate corollary of the above:

Corollary 5.5. For any state (respectively, path) formula φ of mv-ATL* and any state (respectively, path) ξ, we have

$$[\varphi]_{M,\xi} \geq \ell \qquad \text{iff} \qquad M_{f_\ell}, \xi \models \varphi. \tag{6}$$

Note that each M_{f_ℓ} is a classical, 2-valued model. Together with Equation (1), we have $[\varphi]_{M,\xi} = \bigsqcup\{\ell \in \mathcal{JI}(\mathcal{L}) \mid [\varphi]_{M^\ell,\xi} = \top\}$. This gives us a simple algorithm for computing $[\varphi]_{M,\xi}$, presented in Figure 6. The following is straightforward.

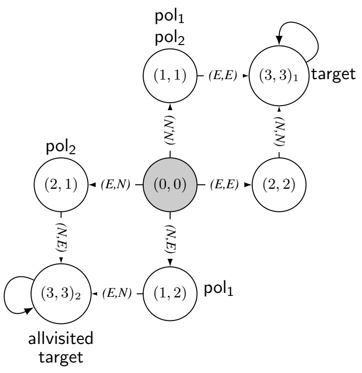
```
Algorithm   $mcheck_{tr}(M, \xi, \varphi)$;

1. For every join-irreducible logical value $\ell \in \mathcal{JI}(\mathcal{L})$, model-check (classically) $\varphi$ in $M_{f_\ell}, \xi$;
2. Collect the values of $\ell$ for which the answer was "yes" in a set $\mathcal{X}$;
3. Return the join of the values in $\mathcal{X}$, i.e., $\bigsqcup \mathcal{X}$.
```

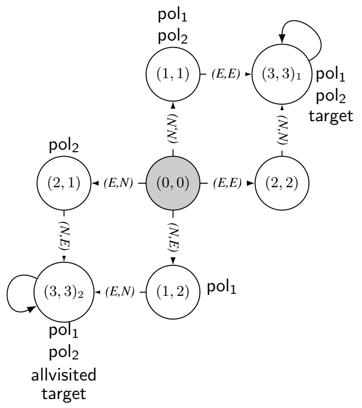
Figure 6. Translation-based model checking for mv-ATL*

Theorem 5.6. The one-to-many reduction from multi-valued model checking of mv-ATL* to 2-valued model checking of ATL* runs in linear time with respect to the size of the model and the number of truth values.

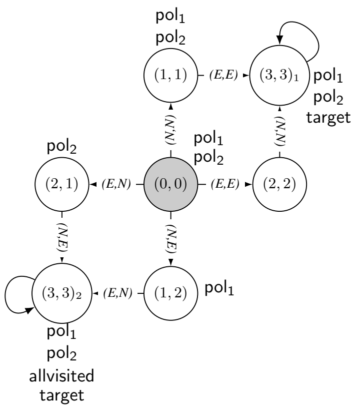
Example 5.7. (Testing of the drones)
Consider the pollution monitoring scenario from the previous examples. Suppose that we want to test the design of a drone patrol before its deployment in the physical environment. One way to carry out offline testing is to model-check the relevant properties of the design against a randomly generated sample of area maps. For the clarity of the examples we have used a crafted by hand map. For the map in Figure 1 and the mv-CGS M_2 in Figure 5, we obtain the collection of classical models presented in Figure 7. Note that projections $(M_2)_{f_\top}$ and $(M_2)_{f_{\top_g}}$ are in fact identical, and similarly for $(M_2)_{f_{\bot_d}}$, $(M_2)_{f_{\bot_g}}$, and $(M_2)_{f_{\bot_d \sqcap \bot_g}}$.



$(M_2)_{f_\top}$ and $(M_2)_{f_{\top_g}}$



$(M_2)_{f_{\top_d}}$



$(M_2)_{f_{\bot_d}}$, $(M_2)_{f_{\bot_g}}$, and $(M_2)_{f_{\bot_d \sqcap \bot_g}}$

Figure 7. Model translations for the drone scenario

Suppose now that we want to compute the value of $\langle\langle 1,2\rangle\rangle \mathsf{F}\, (\mathsf{target} \wedge \mathsf{allvisited} \wedge (\mathsf{pol}_1 \vee \mathsf{pol}_2))$ in $M_2, (0,0)$. The formula holds in state $(0,0)$ of models $(M_2)_{f_{\top_d}}$, $(M_2)_{f_{\bot_d}}$, $(M_2)_{f_{\bot_g}}$, and $(M_2)_{f_{\bot_d \sqcap \bot_g}}$, but not in $(M_2)_{f_\top}$ and $(M_2)_{f_{\top_g}}$. Thus, the output of model checking is $\top_d \sqcup \bot_d \sqcup \bot_g \sqcup (\bot_d \sqcap \bot_g) = \top_d$. Moreover, to model-check $\langle\langle 1\rangle\rangle \mathsf{F}\, \mathsf{pol}_1$, we observe that the formula holds in all the projection models in Figure 7. Thus, its value in $M_2, (0,0)$ is $\top \sqcup \top_d \sqcup \top_g \sqcup \bot_d \sqcup \bot_g \sqcup (\bot_d \sqcap \bot_g) = \top$. □

The algorithm in Figure 6 is an example of *local* model checking. That is, given a state (respectively, path) and a formula, it returns the truth value of the formula in that state (respectively, on that path). In two-valued modal logics, verification of state formulas is often done by means of *global model checking* that returns the exact set of states where the input formula holds. For many logics – including ATL and ATL*– this provides strictly more information with no extra computational cost. The analogous problem for multi-valued modal logics would ask for a *valuation* of the input formula, i.e., a mapping from the states of the model to the truth values of φ. A global model checking algo-

```
Algorithm   $gmcheck_{tr}(M, \varphi)$;

1. Set the initial valuation of $\varphi$ to $\mathcal{V}_\varphi$ such that $\mathcal{V}_\varphi(q) = \bot$ for every $q \in St$;
2. For every join-irreducible logical value $\ell \in \mathcal{JI}(\mathcal{L})$:
    • Compute the set of states $Q_\ell$ that (classically) satisfy $\varphi$ in $M_{f_\ell}$;
    • For each $q \in Q_\ell$, do $\mathcal{V}_\varphi(q) := \mathcal{V}_\varphi(q) \sqcup \ell$;
3. Return $\mathcal{V}_\varphi$.
```

Figure 8. Translation-based global model checking for mv-ATL*

rithm for mv-ATL*, based on the translation to two-valued model checking, is presented in the next section, in Figure 8.

5.2. Translating Implication Formulas: Impossibility Result

Unfortunately, Theorem 5.4 cannot be extended to mv-ATL$^*_\to$, i.e., to the full language containing implication formulas of the form $\varphi_1 \to \varphi_2$, where $\to$ represents the lattice order.

Proposition 5.8. There are lattice reduction triples and formulas of mv-ATL$^*_\to$ that do not satisfy translation condition (5).

Proof:
Consider the lattice $\mathcal{L}_o = (\mathrm{L}_o, \le)$, where $\mathrm{L}_o = \{0, ..., k-1, k, ..., 2k-1\}$, and $\le$ is the usual total order on L_o. Clearly, in $\mathcal{L}_o$ we have $0 = \bot$ and $2k-1 = \top$ according to our lattice notation. Then $(\{0, 2k-1\}, \le)$ is a sublattice of $\mathcal{L}_o$ and the reduction $f : \mathrm{L}_o \to \{0, 2k-1\}$ given by $f(x) = 2k-1$ if $x \ge k$, and $f(x) = 0$ if $x < k$ preserves the bounds in $\mathcal{L}_o$. Thus $(\mathcal{L}_o, \{0, 2k-1\}, f)$ is a lattice reduction triple.

Now take arbitrary k_1, k_2 such that $0 < k_1 < k_2 < k$, and an mv-CGS M over $\mathcal{L}_o^+ = (\mathcal{L}_o, \sigma)$ for an arbitrary $\sigma : \mathcal{C} \to \mathrm{L}_o$ such that, for some state $q \in St$ of M and atomic propositions $p_1, p_2 \in Prop$, we have $V(p_i, q) = k_i$ for $i = 1, 2$.

Next, let $\varphi = p_2 \to p_1$. Since $[p_i]_{M,q} = k_i$ for $i = 1, 2$ and $k_2 > k_1$, we have $\neg([p_2]_{M,q} \le [p_1]_{M,q})$, whence $[\varphi]_{M,q} = 0$.

However, for the model M_1 obtained from M with the reduction f we get $[p_i]_{M_1,q} = 0$ for $i = 1, 2$ (as $k_i < k$, $f(k_i) = 0$ for $i = 1, 2$), where $[p_2]_{M_1,q} \le [p_1]_{M_1,q}$, which implies $[\varphi]_{M_1,q} = 2k-1$. Yet, as $f^{-1}(2k-1) = \{k, k+1, ..., 2k-1\}$ we have $[\varphi]_{M,q} = 0 \notin f^{-1}(2k-1)$, which contradicts Equation (5). □

The above result can be generalized as follows:

Theorem 5.9. If $\mathcal{L} = (\mathrm{L}, \leq)$ of Theorem 5.4 contains a chain or anti-chain of cardinality n, and $\mathcal{L}' = (\mathrm{L}', \leq')$ is a sublattice of $\mathcal{L}$ of cardinality $n' < n$, then there is no function $f : \mathrm{L} \to \mathrm{L}'$ satisfying translation condition (5) if the language under consideration contains implication formulas.

Proof:
Assume $X = \{x_1, x_2, \dots, x_n\}$ is a chain or anti-chain in L. Let M be a model such that, for some state $s \in Q$ of M and propositional variables $p_1, p_2, \dots, p_n \in Prop$, we have $V(s, p_i) = k_i$ for $i = 1, 2, \dots, n$. Consider any $f : L \to L'$. As $card(L') = n' < n$, than there must be $k, l \in \{1, \dots, n\}$ such that $k \neq l$ and $f(x_k) = f(x_l)$. Let

$$\varphi = \begin{cases} p_l \to p_k & \text{if } X \text{ is a chain and } x_k < x_l \\ p_k \to p_l & \text{otherwise} \end{cases}$$

Then, clearly, $x_l \not\leq x_k$ (note that if X is an anti-chain, then $x_r \not\leq x_q$ for any $1 \leq r, s \leq n$). Hence $[p_l]_{M,q} \not\leq [p_k]_{M,q}$, and $[\varphi]_{M,q} = \bot$. However, as $f(x_k) = f(x_l)$, we have $[p_l]_{M_1,q} \leq [p_k]_{M_1,q}$, whence $[\varphi]_{M_1,q} = \top$. Thus f does not satisfy Equation (5). □

In other words, if the size of the target lattice $\mathcal{L}'$ is strictly smaller than the "diameter" of the source lattice $\mathcal{L}$ (that is, the cardinality of the longest chain or antichain in $\mathcal{L}$), then a "clustering" of truth values from $\mathcal{L}$ into $\mathcal{L}'$ that preserves Equation (5) is impossible. Note that the diameter of *any* lattice with more than 2 values must be at least 3, and hence it exceeds the size of the classical 2-valued lattice **2**. The following is an immediate consequence of the above:

Corollary 5.10. For any multi-valued lattice $\mathcal{L}$ there is no reduction of $\mathcal{L}$ to the 2-valued lattice of classical truth values that satisfies the translation condition (5) for the whole language of mv-ATL$^*_\to$.

5.3. Translating Implication Formulas Even More Impossible

We already know that there is no general translation from multi-valued to two-valued model checking for implication formulas. Now we will show that the impossibility result can be extended to any "clustering" of truth values from the original lattice $\mathcal{L}$. To this end, we give a necessary and sufficient condition for the existence of a function $f : L \to L_f$ that preserves bounds in $\mathcal{L}$ and satisfies the translation condition (5) also for implication formulas.

The following lemma states that, in order to obtain the analogue of Theorem 5.4 for implication formulas, the mapping f would have to preserve both the ordering and incomparability of the elements in $\mathcal{L}$.

Lemma 5.11. Let $(\mathcal{L}, \mathcal{L}_f, f)$ be a lattice reduction triple, let M be an mv-CGSs over L, and $f(M)$ its reduction to $\mathcal{L}_f$. Then translation condition (5) of that theorem is satisfied for all implication formulas iff the following conditions hold:

C1: $(\forall x_1, x_2 \in \mathrm{L})\ [x_1 < x_2 \ \Rightarrow\ f(x_1) < f(x_2)]$

C2: $(\forall x_1, x_2 \in \mathrm{L})\ [x_1 \bowtie x_2 \ \Rightarrow\ f(x_1) \bowtie f(x_2)]$

Proof:
Note that an implication formula is a state formula, and that for any such formula ψ we have $[\psi]_{M,q} \in \{\bot, \top\}$ for any mv-CGS M and any $q \in St$. Thus in order to prove (5) for such formulas, it suffices to show that, for any implication formula φ and any state q:

$$[\varphi]_{M_f,q} = \top \text{ iff } [\varphi]_{M,q} \in f^{-1}(\top) \tag{7}$$

"(7) ⇒ (C1 $\wedge$ C2)": We start by proving the necessity of conditions C1 and C2. Assume first that f satisfies (7) for implication formulas. We should prove that f satisfies Conditions C1, C2 for all such formulas. For what follows, denote $\xi = p_1 \to p_2, \psi = p_2 \to p_1$, where $p_1, p_2 \in Prop$, $p_1 \neq p_2$ where $Prop$ is the set of atomic propositions of our logic).

C1: We argue by contradiction. Suppose $x_1, x_2 \in \mathrm{L}, x_1 < x_2$ and $f(x_1) \geq f(x_2)$. Then, as $x_1 < x_2$ implies $x_1 \leq x_2$ and f preserves bounds, we also have $f(x_1) \leq f(x_2)$, whence $f(x_1) = f(x_2)$.

Now let M be an mv-CGS over $\mathcal{L}$ such that, for some state $q \in St$, we have $V(p_i, q) = x_i$ for $i = 1, 2$, and let M_f be the image of M under f. Since $\psi = p_2 \to p_1$ and $[p_2]_{M,q} = x_2 > x_1 = [p_1]_{M,q}$, we have $[\psi]_{M,q} = \bot$. However, $[\psi]_{M_f,q} = \top$, because $[p_2]_{M_f,q} = f(x_2) = f(x_1) = [p_1]_{M_f,q}$. As $\bot \notin f^{-1}(\top)$, this contradicts (7).

C2: We again argue by contradiction. Suppose $x_1, x_2 \in L, x_1 \bowtie x_2$ and $\neg(f(x_1) \bowtie f(x_2))$. Without any loss of generality, we can assume that $f(x_1) \leq f(x_2)$. Let mv-CGSs M, M_f and state q of M be like in the preceding item. Then, as $[p_1]_{M,q} = x_1 \bowtie x_2 = [p_2]_{M,q}$, we have in particular $[p_1]_{M,q} \not\leq [p_2]_{M,q}$. Since $\xi = p_1 \to p_2$, this implies $[\xi]_{M,q} = \bot$. In turn, $[\xi]_{M_f,q} = \top$, because $[p_1]_{M_f,q} = f(x_1) \leq f(x_2) = [p_2]_{M_f,q}$, which again contradicts (7).

"(C1 $\wedge$ C2) ⇒ (7)": It remains to prove the sufficiency of conditions C1 and C2. We assume that C1, C2 hold, and prove that (7) holds for formulas of the form $\varphi = \varphi_1 \to \varphi_2$. We start by proving this result for non-nested implication formulas, i.e., we assume that φ_1, φ_2 do not contain $\to$. Then, by Theorem 5.4, (7) holds for φ_1, φ_2, which implies that

$$[\varphi_i]_{M_f,q} = f([\varphi_i]_{M,q}), \;\; i = 1, 2. \tag{8}$$

"(7L) ⇒ (7R)": We begin with the forward implication in (7). Assume that $[\varphi]_{M_f,q} = \top$. Then $[\varphi_1]_{M_f,q} \leq [\varphi_2]_{M_f,q}$. By (8), this implies $f([\varphi_1]_{M,q}) \leq f([\varphi_2]_{M,q}$. We show by contradiction that

$$[\varphi_1]_{M,q} \leq [\varphi_2]_{M,q}. \tag{9}$$

Suppose that (9) does not hold, then we have two possible cases:

Case 1: $[\varphi_1]_{M,q} > [\varphi_2]_{M,q}$. Then by C1 we have $f([\varphi_1]_{M,q}) > f([\varphi_2]_{M,q})$, whence from (8) we get $[\varphi_1]_{M_f,q} > [\varphi_2]_{M_1,q}$ and $[\varphi]_{M_f,q} = \bot$ — which is a contradiction.

Case 2: $[\varphi_1]_{M,q} \bowtie [\varphi_2]_{M,q}$. Then $f([\varphi_1]_{M,q}) \bowtie f([\varphi_2]_{M,q})$ by C2, whence from (8) we get $\neg([\varphi_1]_{M_f,q} \leq [\varphi_2]_{M_f,q})$. Consequently, $[\varphi]_{M_f,q} = \bot$ — which is again a contradiction.

Thus (9) above holds, whence $[\varphi]_{M,q} = \top \in f_1(\top)$, and the forward implication in (7) holds.

"(7R) $\Rightarrow$ (7L)": The final step is proving the backward implication in (7). Assume that $[\varphi]_{M,q} = f^{-1}(\top)$. As $[\varphi]_{M,q} \in \{\bot, \top\}$ and $f(\bot) \neq \top$ by the preservation of bounds by f and the non-triviality of L, L_f, we obtain $[\varphi]_{M,q} = \top$, whence $[\varphi_1]_{M,q} \leq [\varphi_2]_{M,q}$. Since f preserves bounds, this implies $f([\varphi_1]_{M,q}) \geq f([\varphi_2]_{M,q})$, whence from (8) we obtain $[\varphi_1]_{M_1,q} \leq [\varphi_2]_{M_f,q}$. This yields $[\varphi]_{M_f,q} = \top$, whence the backward implication in (7) holds, too.

Nested formulas: The proof for nested formulas proceeds by induction. Assume that (7) holds for implication formulas with $\rightarrow$ nested at most k times, and assume φ is an implication formula with $\rightarrow$ nested $k+1$ times. Then $\varphi = \varphi_1 \rightarrow \varphi_2$, where $\rightarrow$ is nested at most k times in φ_1, φ_2. Consequently, by the inductive assumption (9) holds for φ_1, φ_2, and repeating the proof given above for implication formulas without nesting of $\rightarrow$ we can show that (7) holds for φ too.

This completes the proof of the sufficiency of C1, C2 for all implication formulas, and the proof of Lemma 5.11. □

From Lemma 5.11 we can easily derive by induction the following general result:

Theorem 5.12. Let the $(\mathcal{L}, \mathcal{L}_f, f)$ be an LRT. Then, the translation condition is satisfied for all formulas of mv-ATL$^*_\rightarrow$ over $\mathcal{L}$ iff conditions C1, C2 of Lemma 5.11 hold.

It can be seen that conditions C1, C2 imply that any translation f meeting them must preserve the exact structure of the lattice $\mathcal{L}$. An important consequence of that fact is:

Corollary 5.13. Given a lattice $\mathcal{L} = (\mathrm{L}, \leq)$ and its sublattice $\mathcal{L}_f = (\mathrm{L}_f, \leq_f)$, any function $f : \mathrm{L} \rightarrow \mathrm{L}_f$ preserving the algebra bounds and satisfying translation condition (5) for all implication formulas must be one-to-one.

Proof:
Suppose that f satisfies the above assumption, $x_1, x_2 \in L$ and $x_1 \neq x_2$. Then we have one of the following cases:

1. $x_1 < x_2$ or $x_2 < x_1$. Then $f(x_1) \neq f(x_2)$ by C1 of Theorem 5.12.

2. $x_1 \bowtie x_2$. Then $f(x_1) \bowtie f(x_2)$ by C2 of Theorem 5.12, which again implies $f(x_1) \neq f(x_2)$.

□

The meaning of Corollary 5.13 is that there is no way of reducing n-valued model checking to k-valued model checking for $k < n$, if we want to handle all implication formulas. Clearly, Corollary 5.10 in Section 5.2 is a special case of the above result.

5.4. Translation of Model Checking for *Some* Implication Formulas

By Corollary 5.5, there is a simple translation from multi-valued to classical model checking for strategic and temporal operators. By Corollary 5.13, we know that it cannot be generally extended to implication formulas. The next question is: can we construct such a translation for *some* implication formulas? If so, for which ones? The impossibility result in Corollary 5.13 is due to the fact that implication formulas can be used to encode the semantics in the language — including in particular its n-valued character. However, one usually wants to model-check one formula at a time. Then, Theorem 5.12 can be in some cases modified to provide the desired reduction:

Theorem 5.14. Let $(\mathcal{L}, \mathrm{L}_f, f)$ be a lattice reduction triple (LRT), let M be a CGS over $\mathcal{L}$ and $f(M)$ its reduction to $\mathcal{L}_f$. Further, let φ be a formula of mv-ATL$^*_\to$ and $Sub(\varphi)$ the set of all its subformulas. Then φ satisfies translation condition (5) whenever, for any implication formula $\phi \in Sub(\varphi)$ such that $\phi = \varphi_1 \to \varphi_2$, any state (resp. path) ξ, and $x_i = [\varphi_i]_{M,\xi}, i = 1, 2$, the following conditions hold:

$$\textbf{C1':}\ x_1 < x_2 \ \Rightarrow\ f(x_1) < f(x_2) \qquad\qquad \textbf{C2':}\ x_1 \bowtie x_2 \ \Rightarrow\ f(x_1) > f(x_2)$$

Proof:
To prove the thesis, we assume that C1', C2' are satisfied, and show by structural induction that translation condition (5)

$$[\psi]_{M_f,\xi} = x \quad \text{iff} \quad [\psi]_{M,\xi} \in f^{-1}(x)$$

holds for any $\psi \in Sub(\varphi)$.

For atomic or constant ψ, the thesis follows from Theorem 5.4. Suppose now that Equation (5) holds for all subformulas of φ having rank k, and assume ψ is of rank $k+1$. If ψ is obtained from subformulas of rank at most k using any operator *Op* other than $\to$, then Equation (5) follows from the fundamental Lemma 5.3.

Thus, it remains to consider the case of $\to$. Assume $\psi = \psi_1 \to \psi_2$, with Equation (5) being satisfied for both ψ_1, ψ_2. Since ψ is an implication formula, according to what we have already noted in the proof of Lemma 5.11, proving (5) for ψ reduces to showing condition (7), i.e.,

$$[\psi]_{M_f,q} = \top \quad \text{iff} \quad [\psi]_{M,q} \in f^{-1}(\top).$$

Note that since ψ_1, ψ_2 are in $Sub(\varphi)$, then C1', C2' hold for $x_i = [\varphi_i]_{M,q}, i = 1, 2$. By the inductive assumption, we also have

$$[\psi_i]_{M_f,q} = f([\psi_i]_{M,q}), \;\; i = 1, 2 \tag{10}$$

"$\Rightarrow$": We begin with the forward implication in (7). Assume that $[\psi]_{M_f,q} = \top$. Then $[\psi_1]_{M_f,q} \leq [\psi_2]_{M_f,q}$. By (10), this implies $f([\psi_1]_{M,q}) \leq f([\psi_2]_{M,q})$. We show by contradiction that it implies

$$[\psi_1]_{M,q} \leq [\psi_2]_{M,q} \tag{11}$$

Suppose that (11) does not hold, then we have two possible cases:

Case 1: $[\psi_1]_{M,q} > [\psi_2]_{M,q}$. Then by condition C1' we have $f([\psi_1]_{M,q}) > f([\psi_2]_{M,q})$, whence from (10) we get $[\psi_1]_{M_f,q} > [\psi_2]_{M_f,q}$ and $[\psi]_{M_f,q} = \bot$, which is a contradiction.

Case 2: $[\psi_1]_{M,q} \bowtie [\psi_2]_{M,q}$. Then $f([\psi_1]_{M,q}) > f([\psi_2]_{M,q})$ by condition C2', which again leads to a contradiction by what we have already proved for Case 1.

Thus (11) above holds, whence $[\psi]_{M,q} = \top \in f^{-1}(\top)$, and the forward implication in (7) holds.

"$\Leftarrow$": The final step consists in proving the backward implication in (7). Assume that $[\psi]_{M,q} = f^{-1}(\top)$. As $[\psi]_{M,q} \in \{\bot, \top\}$ and $f(\bot) \neq \top$ by the preservation of bounds by f and the non-triviality of $\mathcal{L}, \mathcal{L}_f$, we get $[\psi]_{M,q} = \top$, and consequently $[\psi_1]_{M,q} \leq [\psi_2]_{M,q}$. Since f preserves bounds, this implies $f([\psi_1]_{M,q}) \leq f([\psi_2]_{M,q})$, whence from (10) we obtain $[\psi_1]_{M_f,q} \leq [\psi_2]_{M_f,q}$. This yields $[\psi]_{M_f,q} = \top$, whence the backward implication in (7) holds, too. □

Assume that our mv-CGSs are defined over distributive lattices. We now show that the translation method of Section 5.1, based on join irreducible elements $\mathcal{JI}(\mathcal{L})$, can be applied to a formula φ of ATL* and an mv-CGS M, provided that the assumptions of Theorem 5.14 are satisfied. By (1), for each $x \in \mathrm{L}$ we have $x = \bigsqcup(\mathcal{JI}(\mathcal{L}) \cap \downarrow x)$. Let M^ℓ be the model obtained using the translation f_ℓ. Therefore, according to Theorem 5.14: $[\varphi]_{M^\ell,\xi} = x$ iff $[\varphi]_{M,\xi} \in f_\ell^{-1}(x)$ whence $[\varphi]_{M^\ell,\xi} = \top$ iff $[\varphi]_{M,\xi} \in \uparrow\ell$. Thus,

$$[\varphi]_{M,\xi} = \bigsqcup\{\ell \in \mathcal{JI}(\mathcal{L}) \mid [\varphi]_{M^\ell,\xi} = \top\}. \tag{12}$$

Example 5.15. Consider model M_2 in Figure 5 and formula $\phi = \langle\!\langle 1 \rangle\!\rangle \mathsf{G}\,(\mathsf{pol}_1 \to (\mathsf{target} \wedge \mathsf{pol}_2))$. Subformula pol_1 can take the following truth values throughout the model: $\bot, u, \top_d, \top$. Similarly, $\mathsf{target} \wedge \mathsf{pol}_2$ can evaluate to $\bot, \top_d$. Thus by Theorem 5.14 mapping $f_{\top_d}$ meets translation condition (5), and we can use the translation method of Section 5.1 to check if the value of ϕ is at least $\top_d$.

On the other hand, all the other "cutoff" mappings (i.e., $f_{\bot_d \sqcap \bot_g}, f_{\bot_d}, f_{\bot_g}, f_{\top_g}$, and $f_\top$) do not satisfy condition C1', and hence the correctness of the translation is not guaranteed for those truth values.

The following is an immediate consequence of Theorem 5.14.

Corollary 5.16. Let $\mathcal{L}$, its sublattice $\mathcal{L}_f$, $f : \mathrm{L} \to \mathrm{L}_f$, and M, M_f be as in Theorem 5.4. Further, let φ be a formula of mv-ATL$^*_\to$ such that every implication subformula of φ is of the form $\psi_1 \to \psi_2$, where $\psi_i \in \{\boxed{\bot}, \boxed{\top}\}$ for some $i \in \{1, 2\}$. Then φ satisfies the translation condition (5).

Proof:
It suffices to observe that if at least one of the formulas ψ_1, ψ_2 is either $\boxed{\bot}$ or $\boxed{\top}$, then the implication subformula $\psi_1 \to \psi_2$ trivially satisfies conditions C1' and C2' of Theorem 5.14. □

5.5. Recursive Model Checking of ATL*

For many model checking instances, the assumptions of Theorem 5.14 do not hold. In those cases, we cannot translate the multi-valued model checking of mv-ATL$^*_\to$ formulas to the classical model checking for ATL*. Then, the simplest solution is to adapt the standard recursive algorithm that, in order to model-check formula φ, proceeds bottom-up from the simplest subformulas, and replaces them with fresh atomic propositions. In our case, this means that model checking of each implication formula $\varphi_1 \to \varphi_2$ consists in computing the values of φ_1, φ_2 by means of the translation in Section 5.1, and then fixing the valuation of the fresh variable $\mathsf{p}_{\varphi_1 \to \varphi_2}$ according to their comparison. The detailed algorithm is presented in Figure 9.

The main disadvantage of the above method compared to the direct translation method is that it requires computing the values of the new atomic propositions for all states of the model M. In other words, we need to carry out global model checking, whereas for formulas without the implication operator $\to$ both global and local model checking were possible. The method can be possibly improved if we assume that a specific symbolic model checking method for two-valued ATL* is used; we leave a study of this subject for future work.

Nevertheless, the algorithm presented in Figure 9 has two important consequences. First, it provides a general linear-time reduction from model checking mv-$\mathrm{ATL}^*_{\to}$ (resp. mv-$\mathrm{ATL}_{\to}$) to model checking standard 2-valued ATL* (resp. ATL). We state it formally as follows.

Theorem 5.17. The one-to-many reduction from multi-valued model checking of mv-$\mathrm{ATL}^*_{\to}$ to 2-valued model checking of ATL* runs in linear time with respect to the size of the model, the length of the formula, and the number of truth values.

Corollary 5.18. Model checking mv-$\mathrm{ATL}^*_{\to}$ (resp. mv-$\mathrm{ATL}_{\to}$) is $\mathbf{2EXPTIME}$-complete (resp. $\mathbf{P}$-complete) in the size of the model, the length of the formula, and the number of truth values.

Secondly, we note that correctness of the translation does not depend on the type of strategies being used in the semantics of mv-$\mathrm{ATL}^*_{\to}$. As it is, the translation provides a model checking reduction to the *IR* variant of ATL* (perfect information + perfect recall). If we used memoryless strategies of type

```
Algorithm   $gmcheck_{rec}(M, \varphi)$;

1. If $\varphi$ contains no instance of the comparison operator $\to$, then return $gmcheck_{tr}(M, \varphi)$;
2. Else, pick the first implication subformula $\varphi_1 \to \varphi_2$ in $\varphi$, and:
    • Compute $\mathcal{V}_{\varphi_1} := gmcheck_{rec}(M, \varphi_1)$ and $\mathcal{V}_{\varphi_2} := gmcheck_{rec}(M, \varphi_2)$;
    • Create an extension $M'$ of model $M$ by adding a fresh atomic proposition $\mathsf{p}$, and fixing
      its valuation so that $V(\mathsf{p}, q) = \top$ if $\mathcal{V}_{\varphi_1}(q) \le \mathcal{V}_{\varphi_2}(q)$ and $V(\mathsf{p}, q) = \bot$ otherwise;
    • Create formula $\varphi'$ by replacing every occurrence of $\varphi_1 \to \varphi_2$ by $\mathsf{p}$;
    • Return $gmcheck_{rec}(M', \varphi')$.
```

Figure 9. Recursive global model checking for ATL*

$s_a : St \to Act$ instead of perfect recall, the translation would yield reduction to the *Ir* variant of ATL* (perfect information + imperfect recall [45]). Since the *IR* and *Ir* semantics coincide in 2-valued ATL (though not in ATL*!), we get the following.

Theorem 5.19. For mv-ATL$_\to$, memory is irrelevant, i.e., its semantics can be equivalently given by memoryless strategies.

6. Multi-Valued Transitions

In this paper our aim is to propose a framework for a graded interpretation of logical statements referring to the strategic ability of agents and coalitions. Until this point, the “graded” truth values have only originated from non-classical interpretation of atomic propositions and literals. Typically, this happens because when constructing a model we cannot determine the truth of some basic statements in absolute terms (as either true or false). Instead, we assign such basic statements with their ”truth degrees” (which can be also seen as “weights of evidence”) drawn from a suitable lattice, which then propagate to more complex formulas.

Another source of non-classical truth values sometimes considered in the literature is a graded interpretation of transitions. In that case, each transition is labelled according to its “strength.” An extension of mv-ATL$^*_\to$ with weighted transitions is discussed in this section.

6.1. Weighted Transitions: Potential Interpretations

The shift from 2-valued to multi-valued modal logic typically arises when we extend the domain of interpretation for atomic propositions in *states* of the model. The level of truth for $\mathsf{p}_1, \mathsf{p}_2, \dots$, is not crisp anymore, and this propagates to more complex formulae φ via semantic clauses. So far, we have assumed that the transition relation is crisp, i.e., given states q, q' and a vector of actions $\vec{\alpha}$, the transition from q to q' labeled by $\vec{\alpha}$ is either fully included in the model, or is completely absent from it. An alternative would be to consider multi-valued transition relations, with transitions that are possible to a certain degree.

There are at least two sensible interpretations of such weighted transitions. On the one hand, the weight can be interpreted as the strength of evidence supporting the existence of the transition. This approach has been adopted in the previous works on multi-valued temporal logics over arbitrary lattices of truth values [3, 4], with the additional assumption that the weights of transitions are drawn from the same lattice as the values of propositions. A characteristic feature of the semantics in [3, 4] is that, whenever the weights on transitions decrease sufficiently, the value of a temporal formula must also decrease. Formally, consider a multi-valued transition system M, a state q in M, and a formula $\mathsf{AX}\,\varphi$ such that $[\mathsf{AX}\,\varphi]_{M,q} = x$. Moreover, let M' be the same as M except for the weights of all the outgoing transitions from q being strictly lower than x. Then we have $[\mathsf{AX}\,\varphi]_{M',q} < [\mathsf{AX}\,\varphi]_{M,q}$. Analogous characterizations can be shown for all other temporal operators.

On the other hand, the transition weights can be also interpreted as a qualitative distribution, similarly to quantitative transitions in Markov chains and Markov decision processes, used in the semantics of probabilistic temporal logics [46, 35, 19] and their strategic variants [38, 39, 31, 33, 34].

A natural assumption in that case is that the distribution is complete. In the probabilistic case, it amounts to the weights on the outgoing edges from q always summing up to 1. The requirement is essential in models of multi-agent systems, where establishing what *cannot* happen is often as important as reasoning about what can.

In the qualitative case, at a minimum, $[\varphi]_{M,q'} = x$ on all the successors q' of q should imply $[\mathsf{AX}\,\varphi]_{M,q} = x$ (and analogously for other temporal operators and strategic operators). In particular, if the value of φ is bound to be $\top$ at the next moment – no matter how the systems evolves – then $\mathsf{AX}\,\varphi$, $\langle\!\langle A\rangle\!\rangle\mathsf{X}\,\varphi$, etc., should also evaluate to $\top$. It is easy to see that the semantics in [3, 4] do not satisfy this requirement.

In the remainder of the section, we outline how the probabilistic approach can be adapted to arbitrary lattices of transition weights. Our proposal is based on the concept of *designated paths*, i.e., paths that are considered relevant in a given context. We also show that the idea of may/must abstraction can be seen as a special, 3-valued case of this kind of reasoning.

6.2. Weighted Transitions in Concurrent Game Structures

Definition 6.1. (Weighted multi-valued CGS)
Assume two lattices: an interpreted lattice $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$ of truth values, and a lattice $\mathcal{L}_t = (\mathrm{L}_t, \leq_t)$ for weights that will be assigned to transitions. A *weighted multi-valued concurrent game structure (wmv-CGS)* over $\mathcal{L}^+$ and $\mathcal{L}_t$ is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, w, Prop, V, \mathcal{L}^+, \mathcal{L}_t\rangle$, where $\mathbb{A}\mathrm{gt}$, St, Act, t, $Prop$ are as in case of an mv-CGS, and $w : t \to \mathrm{L}_t$ is a weight function which maps each individual transition in t (i.e., each tuple $(q, \alpha_1, \ldots, \alpha_k, t(q, \alpha_1, \ldots, \alpha_k))$ to a value in L_t.

The interpretation of mv-ATL$^*_{\to}$ formulas in a wmv-CGS M as above is parameterized by the set $\mathcal{D}$ of designated values in $\mathcal{L}_t$ — the logical values whose assignment to a formula makes it deemed to be satisfied.

A path in an wmv-CGS is defined analogously as in an mv-CGS. A path $\lambda = q_0q_1q_2\ldots$ is said to be designated if for every i there are actions $\alpha_1, \ldots, \alpha_k$ such that $t(q_i, \alpha_1, \ldots, \alpha_k) = q_{i+1}$ and $w(q_i, \alpha_1, \ldots, \alpha_k, q_{i+1}) \in \mathcal{D}$.

Given $\mathcal{D}$, we reduce a wmv-CGS M to an mv-CGS

$$M_{\mathcal{D}} = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t_{\mathcal{D}}, Prop, V, \mathcal{L}^+\rangle$$

where

$$t_{\mathcal{D}}(q, \alpha_1, \ldots, \alpha_k) = \begin{cases} t(q, \alpha_1, \ldots, \alpha_k) & \text{if } w(q, \alpha_1, \ldots, \alpha_k, t(q, \alpha_1, \ldots, \alpha_k)) \in \mathcal{D} \\ \text{undefined} & \text{otherwise} \end{cases}$$

Then any state of M is a state of $M_{\mathcal{D}}$, and any designated path in M is a path in $M_{\mathcal{D}}$. As the interpretation of mv-ATL$^*_{\to}$ in M we take the interpretation of mv-ATL$^*_{\to}$ in $M_{\mathcal{D}}$:

For any state or designated path ξ in M and any formula φ in mv-ATL$^*_{\to}$, we take:

$$[\varphi]_{M,\xi,\mathcal{D}} = [\varphi]_{M_{\mathcal{D}},\xi} \tag{13}$$

6.3. Embedding May/Must Abstractions

A natural example of many-valued transitions is provided by may-must transitions: the "may" transitions are only possible, but need not happen, while the "must" transitions will necessarily take place. This kind of models were used in [42]. Also, the may/must abstractions presented in [28, 29, 30] produce models with the same or similar structure and interpretation. Adapting the notation, those models take the form $M^{GJ} = \langle St, Prop, \delta_{must}, \delta_{may}, V, \mathcal{L}\rangle$, where $St, Prop, V$ are defined as before, $\mathcal{L} = \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$, and δ_{must}, $\delta_{may} \subseteq St \times St$ are transition relations such that $\delta_{must} \subseteq \delta_{may}$. The language contains negation and conjunction interpreted as in Kleene three-valued calculus over $\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$, and the AX operator interpreted as:

$$[AX\varphi]_{M^{GJ},q} = \begin{cases} \mathbf{t} & \text{if } \forall s'(\delta_{may}(s,s') \Rightarrow [\varphi]_{M^{GJ},s'} = \mathbf{t}) \\ \mathbf{f} & \text{if } \exists s'(\delta_{must}(s,s') \wedge [\varphi]_{M^{GJ},s'} = \mathbf{f}) \\ \mathbf{u} & \text{otherwise} \end{cases}$$

If, following [42], we disregard explicit inclusion of agents and actions in our approach, then if the transition relation δ_{may} is a function, such a model can be represented as an wmv-CGS $M^{JKP} = \langle St, \delta_{may}, Prop, V, \mathcal{L}^+, \mathcal{L}_t\rangle$ with three-valued transitions, where $LT = \{\top, U, \bot\}$, and the weight function $w : St \times St \to LT$ is defined by:

$$w(s,s') = \begin{cases} \top & \text{if } (s,s') \in \delta_{must} \\ U & \text{if } (s,s') \in \delta_{may} \setminus \delta_{must} \\ \bot & \text{otherwise} \end{cases}$$

Denote $\mathcal{D}_\top = \{\top\}, \mathcal{D}_U = \{U, \top\}$. We can show that Godefroid's-Jagadessan's semantics based on M^{GJ} can be expressed using our model M^{JKP} as follows:

Lemma 6.2. If φ does not contain the AX operator, then:

1. $[\varphi]_{M^{GJ},q} = [\varphi]_{M^{JKP},q,\mathcal{D}_\top}$

2. $[AX\varphi]_{M^{GJ},q} = \begin{cases} \mathbf{t} & \text{if } [AX\varphi]_{M^{JKP},q,\mathcal{D}_U} = \mathbf{t} \\ \mathbf{f} & \text{if } [AX\varphi]_{M^{JKP},q,\mathcal{D}_\top} = \mathbf{f} \\ \mathbf{u} & \text{otherwise} \end{cases}$

Proof:
Since both M^{JKP} and M^{GJ} are based on Kleene 3-valued calculus of propositional formulas, Condition 1 obviously holds. Further, as the translation of M^{JKP} to M_{D_U} preserves all transitions in δ_{may}, we have $[AX\varphi]_{M^{JKP},q,\mathcal{D}_U} = \mathbf{t}$ iff $[\varphi]_{M^{JKP},q',\mathcal{D}_U} = \mathbf{t}$ for every $(q,q') \in \delta_{may}$. In view of Condition 1, the latter implies $[\varphi]_{M^{GJ},q'} = \mathbf{t}$ for every $(q,q') \in \delta_{may}$. Consequently, the first clause in Condition 2 holds. For the second clause, note that as $M_{D_\top}$ only contains transitions in δ_{must}, then $[AX\varphi]_{M^{JKP},q,\mathcal{D}_\top} = \mathbf{f}$ iff there is a transition $(q,q') \in \delta_{must}$ such that $[\varphi]_{M^{JKP},q'} = \mathbf{f}$. Then by Condition 1 $[\varphi]_{M^{GJ},q'} = \mathbf{f}$, whence $[AX\varphi]_{M^{GJ},q} = \mathbf{f}$ — and so Condition 2 holds. □

6.4. Model Checking Multi-Valued CGS with Weighted Transitions

Fortunately, the introduction of weighted transition, while enriching our models and making them better suited to some practical applications, does not introduce any essential complications into model-checking compared to mv-CGS's with two-valued transitions. Thus the results obtained in the latter case carry over to mv-CGS, and we have the following generalization of the Reduction Theorem 5.4:

Theorem 6.3. Let $\mathcal{L} = (\mathrm{L}, \leq)$ be an arbitrary finite lattice, $\mathcal{L}_f = (\mathrm{L}_f, \leq_f)$ a sublattice of $\mathcal{L}$, and let $f : \mathrm{L} \to \mathrm{L}_f$ a mapping which preserves arbitrary bounds in $\mathcal{L}$. Furthermore, let $M = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, w, Prop, \mathcal{V}, \mathcal{L}^+ \rangle$ be an wmv-CGS over an interpreted lattice $\mathcal{L}^+ = (\mathrm{L}, \leq, \sigma)$ over $\mathcal{C}$, and let $M_f = \langle \mathbb{A}\mathrm{gt}, St, Act, d, t, w_f, Prop, \mathcal{V}_f, (\mathrm{L}_f, \leq_f, \sigma_f) \rangle$ be the mv-CGS obtained from M by "clustering" the truth values in M according to f, i.e.:

1. $\sigma_f(c) = f(\sigma(c))$ for any $c \in \mathcal{C}$,
2. $w_f(\tau) = f(w(\tau)))$ for any $\tau \in t$, and
3. $V_f(p, q) = f(V(p, q))$ for any $q \in St$ and $p \in Prop$.

Then, for any state (respectively, path) formula φ of mv-ATL$^*_\to$ over $\mathcal{L}$, any state (respectively, path) ξ, and any set of designated truth values $\mathcal{D}$, we have

$$[\varphi]_{M,\xi,\mathcal{D}} \in f^{-1}(x) \qquad \text{iff} \qquad [\varphi]_{M_f,\xi,\mathcal{D}} = x \tag{14}$$

Proof:
Straightforward from Equation (13) and Theorem 5.4.

Note that the conditions of the above theorem (preservation of the bounds plus Conditions 1 and 3) correspond to those of Theorem 5.4 , with an analogous Condition 2 for weights added). □

This theorem can be used, in a way analogous to that employed for mv-ATL$^*_\to$ with two-valued transitions, to reduce mv-model checking for mv-ATL$^*_\to$ with mv-transitions to two-valued model checking. This is because the semantics of mv-ATL$^*_\to$ with many-valued transitions contains an embedded reduction of models with mv-transitions to models with two-valued transitions — and for those models we can again use the reduction based on our threshold functions. Consequently, the local and global model checking algorithm given in Figure 9 and Figure 8 also carry-over to the case of many-valued transitions.

Like previously, the positive results quoted above apply to formulas which do not involve the implication operator. For the formulas involving that operator, the negative result obtained in case of two-valued transitions of course still holds — because a CGS with two-valued transitions is just a special case of a CGS with many-valued transitions.

7. Multi-Valued Verification of Agents with Imperfect Information

ATL and ATL* were originally proposed for reasoning about agents in perfect information scenarios. It can be argued that realistic multi-agent systems always include some degree of limited observability [45, 47, 48, 49, 50, 51, 52]. However, model checking of ATL and ATL* with imperfect information is hard – more precisely, $\mathbf{\Delta_2^P}$- to $\mathbf{PSPACE}$-complete for agents playing memoryless strategies [45, 53, 54] and undecidable for agents with perfect recall [55]. Furthermore, the imperfect information semantics of strategic ability does not admit standard fixpoint equivalences [56], which makes incremental synthesis of strategies cumbersome. Practical attempts at the problem have emerged only recently [57, 58, 59, 60, 61, 62], and the experimental results show that verification is feasible only for very small models.

Such hard problems can be potentially tackled by means of approximation techniques [30, 63]. In particular, abstraction techniques [43, 30, 23] can be used to cluster multiple states and/or transitions in the system into *abstract* states and transitions, thus reducing the model size. However, in order to be effective, the abstraction must be very coarse, which potentially results in loss of information about the truth of (some) atomic propositions and the existence of (some) transitions. This leads to a substantial reduction of the verification cost, possibly at the expense of introducing non-classical truth values of propositions in some abstract states, as well as transitions of various strength. In consequence, multi-valued model checking can be extremely useful when reasoning about strategies under uncertainty.

Clearly, all the previously cited reasons for using multi-valued verification (description of the world based on a non-classical notion of truth, lifting the logical reasoning to a richer domain of answers, inconclusive or inconsistent information about the system, conflicting evidence coming from different sources, inconclusive verification procedure, etc.) are also relevant for agents with uncertainty. In this section, we show that the framework of mv-ATL$^*_\rightarrow$ can be easily extended to the case of imperfect information.

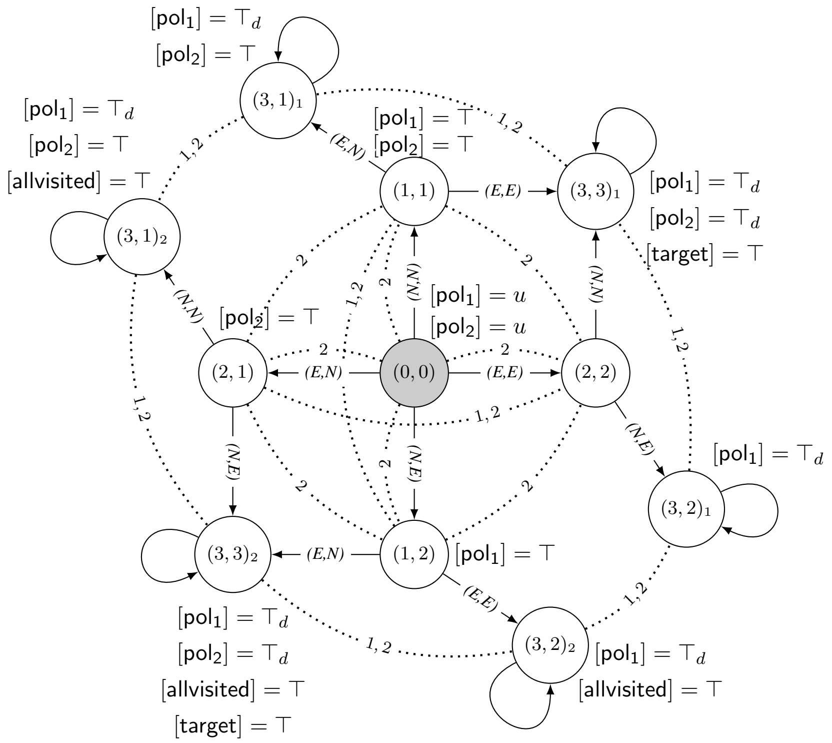
7.1. Logic mv-ATL$^*_\rightarrow$ with Imperfect Information

Let us extend mv-CGS with epistemic indistinguishability relations $\sim_1, \dots, \sim_k \subseteq St \times St$, one per agent in $\mathbb{A}\mathrm{gt}$. The idea is that, whenever $q \sim_a q'$ and the system is in state q, agent a might think that the system is actually in q'. Each $\sim_a$ is assumed to be an equivalence relation. We also assume that the resulting model is *uniform* with respect to the indistinguishability relations, i.e., $q \sim_a q'$ implies $d_a(q) = d_a(q')$. In other words, the choices available to an agent are identical in the states indistinguishable for that agent.

In a similar way, strategies under imperfect information must specify identical choices in indistinguishable situations. That is, memoryless strategies with imperfect information (*ir* strategies, for short) are functions $s_a : St \to Act$ such that $q \sim_a q'$ implies $s_a(q) = s_a(q')$. Moreover, perfect recall strategies with imperfect information (shortly: *iR* strategies) are functions $s_a : St^+ \to Act$ st. $q_0 \sim_a q_0', \dots, q_n \sim_a q_n'$ implies $s_a(q_0 \dots q_n) = s_a(q_0' \dots q_n')$. Again, collective strategies for $A \subseteq \mathbb{A}\mathrm{gt}$ are tuples of individual strategies for $a \in A$. We denote them by Σ_A^{ir} and Σ_A^{iR}, respectively.

The semantics of mv-ATL$^*_{\mathfrak{S}\rightarrow}$, parameterized by the type of strategies $\mathfrak{S} = IR, Ir, ir, iR$, can be defined by replacing the clause for the strategic operators from Section 4 as follows:

$$[\langle\!\langle A\rangle\!\rangle\gamma]^{\mathfrak{S}}_{M,q} = \bigsqcup_{s_A\in\Sigma_A^{\mathfrak{S}}} \bigsqcap_{\lambda\in out(q,s_A)}\{[\gamma]^{\mathfrak{S}}_{M,\lambda}\};$$
$$[[\![A]\!]\gamma]^{\mathfrak{S}}_{M,q} = \bigsqcap_{s_A\in\Sigma_A^{\mathfrak{S}}} \bigsqcup_{\lambda\in out(q,s_A)}\{[\gamma]^{\mathfrak{S}}_{M,\lambda}\}.$$



$[\mathsf{pol}_1] = \top_d$
$[\mathsf{pol}_2] = \top$
$(3,1)_1$
$[\mathsf{pol}_1] = \top_d$
$[\mathsf{pol}_2] = \top$
$[\mathsf{allvisited}] = \top$
$(3,1)_2$
$[\mathsf{pol}_1] = \top$
$[\mathsf{pol}_2] = \top$
$(1,1)$
(E,N)
(E,E)
$(3,3)_1$
$[\mathsf{pol}_1] = \top_d$
$[\mathsf{pol}_2] = \top_d$
$[\mathsf{target}] = \top$
(N,N)
$[\mathsf{pol}_1] = u$
$[\mathsf{pol}_2] = u$
$[\mathsf{pol}_2] = \top$
$(2,1)$
$(0,0)$
$(2,2)$
(N,E)
$[\mathsf{pol}_1] = \top_d$
$(3,2)_1$
$(3,3)_2$
$(1,2)$
$[\mathsf{pol}_1] = \top$
$[\mathsf{pol}_1] = \top_d$
$[\mathsf{pol}_2] = \top_d$
$[\mathsf{allvisited}] = \top$
$[\mathsf{target}] = \top$
$(3,2)_2$
$[\mathsf{pol}_1] = \top_d$
$[\mathsf{allvisited}] = \top$
1,2
2

Figure 10. Multi-valued model M_3 for drones with imperfect information. Epistemic indistinguishability is depicted by dotted lines.

Example 7.1. (Drones with partial information)
Consider again the drone model introduced in Example 4.3 and Figure 5. We assume now that drone 1 sees its own position but not that of drone 2, whereas drone 2 only sees if the other drone is in the same location but does not recognize the location itself. Moreover, each drone can identify the initial state (i.e., $(0,0)$), as well as recognize that it has run out of battery (states $(3,3)_1$ and $(3,3)_2$). Finally, drone 2 – not knowing its exact position – may try to fly in a direction which is not available for a given location (e.g., fly North in location 2). In that case, the attempt fails, and the drone stays in its current location. The updated mv-CGS M_3 is presented in Figure 10.

For the formulas from Example 4.4, we now have :

- $[\langle\langle 1\rangle\rangle \mathsf{F}\ \mathsf{pol}_1]^{ir}_{M_2,(0,0)} = [\langle\langle 1\rangle\rangle \mathsf{F}\ \mathsf{pol}_1]^{iR}_{M_2,(0,0)} = \top$, as the strategy to fly North in state $(0,0)$, and then East in $(1,1)$ or $(1,2)$ is uniform for drone 1;

- $[\langle\langle 2\rangle\rangle \mathsf{F}\ \mathsf{pol}_2]^{ir}_{M_2,(0,0)} = [\langle\langle 2\rangle\rangle \mathsf{F}\ \mathsf{pol}_2]^{iR}_{M_2,(0,0)} = \top$ (the analogous strategy for drone 2 is *not* uniform, but the agent can achieve the goal by playing N in all the states);

- $[\langle\!\langle 1,2\rangle\!\rangle \mathsf{F}\,(\mathsf{target} \wedge \mathsf{allvisited} \wedge (\mathsf{pol}_1 \vee \mathsf{pol}_2))]^{ir}_{M_2,(0,0)} = \bot$ because neither of the uniform memoryless strategies leads to a state where $\mathsf{target} \wedge \mathsf{allvisited}$ holds;

- $[\langle\!\langle 1,2\rangle\!\rangle \mathsf{F}\,(\mathsf{target} \wedge \mathsf{allvisited} \wedge (\mathsf{pol}_1 \vee \mathsf{pol}_2))]^{iR}_{M_2,(0,0)} = \top_d$ (example strategy: drone 1 flies North in the first step, and East in the second, while drone 2 moves East and then North). □

Objective vs. subjective semantics of ability. We note that the above semantic rule corresponds to the notion of *objective ability*. That is, given a strategy, we only look at its outcome paths starting from the current global state of the system q. The alternative, *subjective ability*, requires the strategy to succeed on all the paths starting from states indistinguishable from q. Let $\sim_a(q) = \{q' \mid q \sim_a q'\}$. This can be formalized by the following adaptation of the semantic rule:

$$\begin{aligned}
[\langle\!\langle A\rangle\!\rangle\gamma]^{\mathfrak{S}}_{M,q} &= \bigsqcup_{s_A\in\Sigma^{\mathfrak{S}}_A} \bigsqcap_{a\in A} \bigsqcap_{q'\in\sim_a(q)} \bigsqcap_{\lambda\in out(q',s_A)} \{[\gamma]^{\mathfrak{S}}_{M,\lambda}\};\\
[[\![A]\!]\gamma]^{\mathfrak{S}}_{M,q} &= \bigsqcap_{s_A\in\Sigma^{\mathfrak{S}}_A} \bigsqcup_{a\in A} \bigsqcup_{q'\in\sim_a(q)} \bigsqcup_{\lambda\in out(q',s_A)} \{[\gamma]^{\mathfrak{S}}_{M,\lambda}\}.
\end{aligned}$$

A more detailed discussion on the epistemic aspects of strategic ability can be found in [64, 65]. We leave the proper treatment of diverse epistemic variants of mv-ATL$^*_{\rightarrow}$ for the future.

7.2. Model Checking Techniques and Formal Results

We emphasize again that the correctness of the techniques proposed in Section 5 *does not depend on the actual definition of the strategy sets* Σ_A. In consequence, the results carry over to the imperfect information case, and the techniques can be applied *in exactly the same way* to obtain model checking reductions from mv-ATL$^*_{\mathfrak{S}\rightarrow}$ to the corresponding 2-valued cases. This demonstrates the power of the translation method that can be directly applied to a vast array of possible semantics for ATL*. Again, multi-valued verification of mv-ATL$^*_{\mathfrak{S}\rightarrow}$ incurs only linear increase in the complexity compared to the 2-valued case.

8. Case Study: Multi-Valued Verification of the Drone Model

Besides the theoretical results discussed in the preceding sections, we present an experimental evaluation of our approach to verification of strategic abilities. To this end, we propose a new scalable benchmark based on the running example employed throughout the paper. We use the CGS template of the team of drones patrolling for pollution in a city (cf. Examples 4.3 and 7.1, as well as the graphs in Figures 5 and 10), but with a more complex map to make the study more realistic. The details and outcomes of the experiments are presented further on in this section.

8.1. Model Description

The benchmark is an extension of the drone model used in the previous sections. To recall, we consider a number of drones flying over a fixed area, with each drone modeled as a separate agent. The map is represented by a directed graph that defines the locations *Loc* and the paths used by the agents to move between those locations. We employ the map shown in Figure 11. For the experiments, we assume

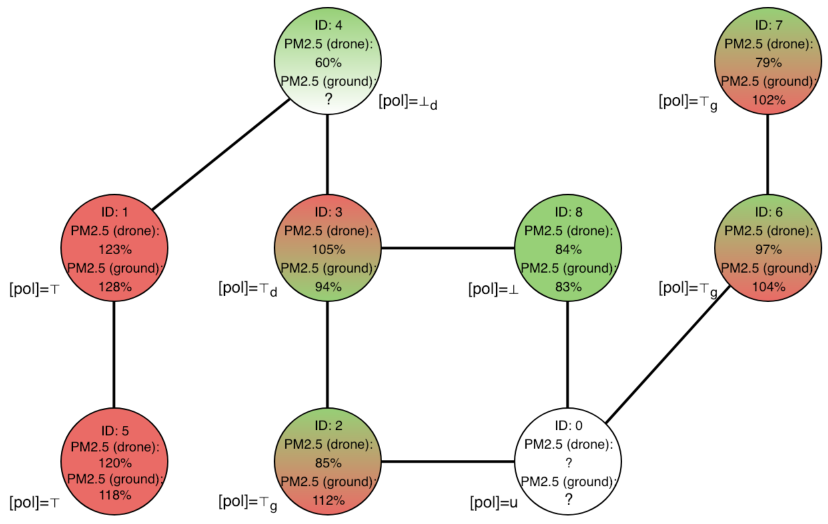


Figure 11. The map used in the experiments

that the connections between locations are symmetric (i.e., can be traversed both ways), and hence an undirected graph is a sufficient representation of the map.

The system consists of a number of drone agents and the environment. The set of all drones is denoted by D. A drone can use its sensors to measure the pollution at its current location. Moreover, it can communicate with the other drones at the same and adjacent locations using bluetooth, and obtain their current readings. The readings from all the ground sensors are broadcasted by the monitoring center, and hence are available to all drones at all times. This is modeled by an epistemic indistinguishability relation, with the following information available to the drone:

- Its current position (i.e., a location number);
- Reading from the drone sensor in its current position;
- Readings from the adjacent drones;
- Readings from all the ground sensors;
- A battery charge level;
- A set of already visited places.

We assume that the time span of the mission is at most 30 mins (currently, there are still relatively few types of drones that can fly longer than a couple of minutes, and they are mostly used in industrial and military contexts). With this provision, we can assume that the environment is stationary throughout the mission. That is, while traversing the map the drones will always get the same readings from a given location.

Each drone can perform five possible actions: *go North*, *South*, *East*, *West*, and *Wait*. Any movement consumes energy. When the battery level drops to zero, the only action that the drone can perform is *Wait*. This means it will stay at its current location forever (since our model does not feature battery recharging). However, such an immobilized drone can still broadcast information to the nearby drones.

As before, we use multi-valued atomic propositions $\mathsf{pol_d}$, $d \in D$, with values drawn from the lattice $\mathbf{2} + \mathbf{2} \times \mathbf{2} + \mathbf{2} \times \mathbf{2}$. The interpretation of $\mathsf{pol_d}$ is given by the combined readings of drone d's sensor and of the ground sensor at the current location of d.

The models are scaled with respect to the following parameters:

- A number of drones;
- An initial battery level (the same for each drone).

#drones	energy	#states	tgen	tverif	output
1	1	5	0.007	0.01	$\top_g$
1	2	14	0.005	0.05	$\top_d \sqcup \top_g$
1	3	32	0.02	0.10	$\top_d \sqcup \top_g$
1	4	61	0.04	0.26	$\top$
1	5	106	0.03	0.38	$\top$
1	10	601	0.37	2.13	$\top$
1	100	16740	8.84	85.91	$\top$
1	1000	178740	85.85	timeout	–
2	1	17	0.01	0.08	$\top_g$
2	2	98	0.05	0.54	$\top_d \sqcup \top_g$
2	3	422	0.34	2.16	$\top_d \sqcup \top_g$
2	4	1263	1.23	6.32	$\top$
2	5	3288	3.83	16.77	$\top$
2	10	55757	89.23	719.81	$\top$
2	100	–	timeout	–	–
3	1	65	0.02	0.46	$\top_g$
3	2	794	1.02	5.34	$\top_d \sqcup \top_g$
3	3	6626	12.44	43.09	$\top_d \sqcup \top_g$
3	4	31015	70.55	293.59	$\top$
3	5	122140	414.57	3003.56	$\top$
3	10	–	timeout	–	–

Figure 12. Experimental results for ϕ_{1L}

8.2. Formulas

In the rest of this section, d will refer to an arbitrary drone in the set D. The first formula to be verified is

$$\phi_1 \quad = \quad \mathsf{EF\,pol_d} \rightarrow \langle\langle d \rangle\rangle \mathsf{F\,pol_d}$$

#drones	energy	#states	tgen	Lower approx.		Upper approx.	
				tverif	output	tverif	output
1	1	5	0.008	0.03	$\top_g$	0.02	$\top_g$
1	2	14	0.006	0.05	$\top_d \sqcup \top_g$	0.06	$\top_d \sqcup \top_g$
1	3	32	0.01	0.14	$\top_d \sqcup \top_g$	0.21	$\top_d \sqcup \top_g$
1	4	61	0.02	0.22	$\top$	0.28	$\top$
1	5	106	0.03	0.39	$\top$	0.48	$\top$
1	10	601	0.27	2.12	$\top$	2.38	$\top$
1	100	16740	9.70	59.27	$\top$	64.40	$\top$
1	1000	178740	92.78	615.23	$\top$	669.85	$\top$
1	10000	1798740	890.85	6125.77	$\top$	6054.80	$\top$
1	12000	2158740	1122.84	timeout	–	timeout	–
2	1	17	0.01	0.18	$\top_g$	0.10	$\top_g$
2	2	98	0.06	0.66	$\top_d \sqcup \top_g$	0.64	$\top_d \sqcup \top_g$
2	3	422	0.33	2.63	$\top_d \sqcup \top_g$	2.40	$\top_d \sqcup \top_g$
2	4	1263	1.21	8.43	$\top$	7.31	$\top$
2	5	3288	3.53	19.08	$\top$	18.19	$\top$
2	10	55757	89.30	323.42	$\top$	321.15	$\top$
2	100	–	timeout	–	–	–	–
3	1	65	0.05	0.47	$\top_g$	0.52	$\top_g$
3	2	794	0.95	5.71	$\top_d \sqcup \top_g$	5.90	$\top_d \sqcup \top_g$
3	3	6626	12.31	55.60	$\top_d \sqcup \top_g$	50.19	$\top_d \sqcup \top_g$
3	4	31015	78.63	271.55	$\top$	239.28	$\top$
3	5	122140	382.30	1719.81	$\top$	932.31	$\top$
3	10	–	timeout	–	–	–	–

Figure 13. Experimental results for ϕ_{1R}

It says that if drone d *might* detects pollution to some degree, then d has a strategy to guarantee that this will indeed be the case. Note that the formula is an implication, and hence – due to the results in Section 5.2 – a straightforward reduction to classical model checking is problematic. Because of that, we use the recursive reduction algorithm of Section 5.5. That is, we split ϕ_1 into its left hand side ($\mathsf{EF}\ \mathsf{pol_d}$) and right hand side ($\langle\langle d\rangle\rangle\mathsf{F}\ \mathsf{pol_d}$). We also observe that the left hand side of the implication, expressed in ATL and transformed to the negation normal form, becomes $[\![\emptyset]\!]\mathsf{F}\ \mathsf{pol_d}$. Thus, in order to determine the value of ϕ_1, we need to carry out multi-valued model checking of the following two formulas:

- $\phi_{1L} = [\![\emptyset]\!]\mathsf{F}\ \mathsf{pol_d}$, and
- $\phi_{1R} = \langle\langle d\rangle\rangle\mathsf{F}\ \mathsf{pol_d}$,

each of them satisfying the preconditions of Theorem 5.4.

We observe that the above specification is relatively weak: it requires that if pollution is present somewhere then the drone is able to find it at some location. In order to allow for a finer-grained specification, we add to the drone model a family of atomic propositions $\mathsf{at}_{\mathsf{d,loc}}$ with classical, 2-valued interpretation. More precisely, $\mathsf{at}_{\mathsf{d,loc}}$ evaluates to $\top$ in the states where drone d is at location $loc \in Loc$, and to $\bot$ everywhere else. In addition, we allow for cooperation between the drones. More exactly, we will be looking at joint strategies of the team of all drones D with the following property: if any of the drones might detect pollution at location loc, then the drones can ensure that one of them will indeed detect it:

$$\phi_2 \quad = \quad \bigwedge_{loc \in Loc} \big(\mathsf{EF} \bigvee_{d \in D} (\mathsf{at}_{\mathsf{d,loc}} \wedge \mathsf{pol}_{\mathsf{d}}) \quad \rightarrow \quad \langle\!\langle D \rangle\!\rangle \mathsf{F} \bigvee_{d \in D} (\mathsf{at}_{\mathsf{d,loc}} \wedge \mathsf{pol}_{\mathsf{d}})\big).$$

Again, the formula is an implication, and thus requires separate treatment of the left and right hand sides of "$\rightarrow$." Here, we only report the verification results for the right-hand subformula, i.e.:

- $\phi_{2R}^{loc} = \langle\!\langle D \rangle\!\rangle \mathsf{F} \bigvee_{d \in D} (\mathsf{at}_{\mathsf{d,loc}} \wedge \mathsf{pol}_{\mathsf{d}})$

for an arbitrary selected value of loc.

The considered formulas emphasize the importance of the comparison operator $\rightarrow$ for actual specification of properties. Many (if not most) relevant properties of multi-agent systems are expressed as an implication: if the assumptions are satisfied, the target property should hold as well. The multi-valued variant of such requirements demands that ψ is satisfied to at least the same degree as φ.

8.3. Semantics and Algorithms

We note that formula ϕ_{1L} refers only to the abilities of the empty coalition, and hence does not involve reasoning about imperfect information. In consequence, one can as well evaluate it using the perfect information semantics of mv-ATL*. Then, the translation in Section 5.1 reduces the multi-valued verification of ϕ_{1L} to model checking of 2-valued ATL with perfect information. We implement the latter by means of the standard fixpoint algorithm from [2].

In contrast, the semantics of formulas ϕ_{1R} and ϕ_{2R} refers to strategies with imperfect information. Accordingly, the translation in Section 5.1 reduces the problem to model checking of 2-valued ATL with imperfect information. Since the exact model checking of abilities under imperfect information is hard, both theoretically [45, 66] and in practice [9, 67, 57], we go around the complexity by using the fixpoint-based approximate model checking algorithm proposed recently in [63]. That is, the 2-valued model checking of ϕ_{1R} proceeds by a model-independent translation to its upper and lower variants ϕ_{1R}^U, ϕ_{1R}^L, both of which can be verified by fixpoint algorithms. If the verification output for ϕ_{1R}^U and ϕ_{1R}^L matches, it is guaranteed correct for ϕ_{1R}, too; otherwise, the outcome is inconclusive. The 2-valued model checking for ϕ_{2R} is obtained analogously.

As we will see, the output of the lower and the upper approximation always matched in our experiments (cf. Figures 13 and 14), thus providing fully conclusive outcome.

8.4. Experimental Results

The results of the experiments are presented in Figures 12 (for formula ϕ_{1L}), 13 (formula ϕ_{1R}), and 14 (formula ϕ_{2R}). For each of the formulas, we considered several configurations of the drone model.

The main scaling factor was the number of the drones in the system. The second source of complexity was the initial energy level (the same for every drone). The initial location on the map was always 0, for all the drones in the system.

The experiments were conducted on an Intel Core i7-6700 CPU with dynamic clock speed of 2.60–3.50 GHz, 32 GB RAM, running under 64bit Windows 10. The times are given in seconds; the timeout was set to 2 hours. As the performance results show, multi-valued verification of strategic ability scales up similarly to two-valued model checking [63], which confirms the theoretical results in Section 5.5.

The software used to conduct the experiments can be found at the address https://github.com/blackbat13/stv. The software is implemented in Python 3. As it is an on-going development, it does not accept any input language. Instead, model generators are used. Models are generated as transition graphs and stored explicitly in the memory.

As the experiments show, the result of the formula depends mostly on the initial energy of the drones. If given enough energy, drones can visit every place on the map, hence detecting any pollution. On the other hand, even a drone with very small capacity of the battery can detect something. As can be seen in Figures 12 and 13, for the cases in which initial energy of the drones were less than 3, answer was more informative than simple false, as it would have been if we had used two-valued logic. It shows that multi-valued logics can provide the designer or analyst with much more useful information beyond a simple yes/no answer.

energy	#states	tgen	Lower approx.		Upper approx.	
			tverif	output	tverif	output
1	65	0.04	0.45	$\bot$	0.46	$\bot$
2	794	1.05	5.60	$\top_g$	5.43	$\top_g$
3	6626	12.05	62.64	$\top_g$	45.45	$\top_g$
4	31015	77.23	233.45	$\top_g$	227.44	$\top_g$
5	122140	379.29	951.14	$\top_g$	854.41	$\top_g$
6	349121	1276.82	2706.15	$\top_g$	2423.55	$\top_g$
7	880562	3446.30	6383.17	$\top_g$	6052.17	$\top_g$
8	1850861	9718.62	timeout	–	timeout	–

Figure 14. Experimental results: ϕ_{2R}^{loc} for $\#drones = 3$ and $loc = 7$

9. Conclusions

In this paper we study a variant of alternating-time temporal logic, denoted as mv-ATL$^*_\rightarrow$, where the truth values are taken from an arbitrary distributive lattice. We argue that multi-valued model checking of mv-ATL$^*_\rightarrow$ specifications can be useful, especially for systems whose models cannot be fully analyzed due to their complexity and/or inaccessibility of the relevant information. Other examples include systems with information coming from multiple, potentially conflicting sources. We propose the semantics of mv-ATL$^*_\rightarrow$ first in the simplest case of perfect information strategies and models with crisp, classical transition functions. Then, we show how to extend the framework to the case of multi-valued transitions, as well as other notions of strategies (in particular, variants of strategic reasoning for agents with limited observation capabilities).

In terms of technical results, we prove that our multi-valued semantics of mv-$\mathrm{ATL}^*_{\rightarrow}$ provides a conservative extension of the classical 2-valued variant. More importantly, we propose efficient (i.e., polynomial-time) translations from multi-valued model checking to the 2-valued case. We formally characterize the conditions under which the translation can be carried out by non-recursive one-to-many reduction, and propose a recursive procedure for the remaining instances of the problem. The proposed techniques are elegant enough to be directly applicable to other semantic variants of strategic ability, for example, those referring to imperfect information scenarios. This allows for non-classical model checking of abilities while benefiting from the ongoing development of classical model checkers and game solvers.

Finally, we back up our proposal by a series of experiments in a simulated scenario of drones patrolling for pollution in a city. Besides promising performance results, the experiments demonstrate also the use of the *relevant implication*, based on comparison of truth values, which is among the main novel contributions of this paper. The operator can be used to provide a multi-valued counterpart of material implication, with an intuitive and appealing interpretation. This is especially important in multi-agent systems where many relevant properties are indeed based on implication, which makes them difficult to formalize in the multi-valued case.

In the future, we plan to extend the framework of mv-$\mathrm{ATL}^*_{\rightarrow}$ to richer specification languages, such as Strategy Logic [68, 69, 70]. We would also like to take a closer look at multi-valued models arising from state and action abstractions, and to the application of multi-valued model checking to verification of strategic ability under imperfect information.

Acknowledgements. The authors thank Arthur Queffelec for his help in the implementation of the model checking algorithm. Moreover, Wojciech Jamroga acknowledges the support of the 7th Framework Programme of the European Union under the Marie Curie IEF project ReVINK (PIEF-GA-2012-626398). Wojciech Jamroga, Damian Kurpiewski, and Wojciech Penczek acknowledge the support of the National Centre for Research and Development (NCBR), Poland, under the PolLux projects VoteVerif (POLLUX-IV/1/2016) and STV (POLLUX-VII/1/2019). Damian Kurpiewski and Wojciech Penczek acknowledge also the support of CNRS/PAN under the project PARTIES.

References

[1] Alur R, Henzinger TA, Kupferman O. 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.

[2] Alur R, Henzinger TA, Kupferman O. Alternating-Time Temporal Logic. *Journal of the ACM*, 2002. **49**:672–713. doi:10.1145/585265.585270.

[3] Konikowska B, Penczek W. Model Checking for Multi-Valued Computation Tree Logics. In: Fitting M, Orłowska E (eds.), Beyond Two: Theory and Applications of Multiple Valued Logic, pp. 193–210. Physica-Verlag, 2003.

[4] Konikowska B, Penczek W. Model checking of multivalued logic of knowledge and time. In: Proceedings of AAMAS. ACM, 2006 pp. 169–176.

[5] Bauer A, Leucker M, Schallhart C. Monitoring of Real-Time Properties. In: Proceedings of FSTTCS, volume 4337 of *Lecture Notes in Computer Science*. Springer, 2006 pp. 260–272. doi:10.1007/11944836_25.

[6] Bauer A, Leucker M, Schallhart C. The Good, the Bad, and the Ugly, But How Ugly Is Ugly? In: Proceedings of RV, volume 4839 of *Lecture Notes in Computer Science*. Springer, 2007 pp. 126–138. doi:10.1007/978-3-540-77395-5_11.

[7] Biere A, Cimatti A, Clarke E, Zhu Y. Symbolic Model Checking Without BDDs. In: Proceedings of Tools and Algorithms for Construction and Analysis of Systems (TACAS), volume 1579 of *Lecture Notes in Computer Science*. Springer, 1999 pp. 193–207.

[8] Penczek W, Lomuscio A. Verifying Epistemic Properties of Multi-Agent Systems via Bounded Model Checking. In: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS). ACM Press, New York, NY, USA. ISBN 1-58113-683-8, 2003 pp. 209–216.

[9] Lomuscio A, Qu H, Raimondi F. MCMAS: An Open-Source Model Checker for the Verification of Multi-Agent Systems. *International Journal on Software Tools for Technology Transfer*, 2015. doi:10.1007/s10009-015-0378-x. Availabe online.

[10] Jamroga W, Konikowska B, Penczek W. Multi-Valued Verification of Strategic Ability. In: Proceedings of the 15th International Conference on Autonomous Agents & Multiagent Systems AAMAS 2016. 2016 pp. 1180–1189.

[11] Fitting M. Many-Valued Modal Logics. *Fundamenta Informaticae*, 1991. **15(3-4)**:335–350.

[12] Fitting M. Many-Valued Modal Logics II. *Fundamenta Informaticae*, 1992. **17**:55–73.

[13] Vijzelaar S, Fokkink W. Multi-valued Abstraction Using Lattice Operations. In: Proceedings of ACSD. 2015 pp. 70–79. doi:10.1109/ACSD.2015.18.

[14] Gurfinkel A, Chechik M. Multi-Valued Model Checking via Classical Model Checking. In: Proceedings of CONCUR, volume 2761 of *LNCS*. Springer-Verlag, 2003 pp. 266–280.

[15] Bruns G, Godefroid P. Model Checking with Multi-valued Logics. In: ICALP. 2004 pp. 281–293.

[16] Shoham S, Grumberg O. Multi-valued model checking games. *J. Comput. Syst. Sci.*, 2012. **78**(2):414–429.

[17] Pan H, Li Y, Cao Y, Ma Z. Model Checking Computation Tree Logic over Finite Lattices. *Theoretical Computer Science*, 2016. **612**:45–62.

[18] B Aminof AM O Kupferman. Improved model checking of hierarchical systems. *Information Computation*, 2012. **210**:68–86.

[19] Kwiatkowska M, Norman G, Parker D. PRISM: Probabilistic Symbolic Model Checker. In: Proceedings of TOOLS, volume 2324 of *Lecture Notes in Computer Science*. Springer, 2002 pp. 200–204. doi:10.1007/3-540-46029-2_13.

[20] Godefroid P, Jagadeesan R. On the Expressiveness of 3-Valued Models. In: Proceedings of VMCAI'03, volume 2575 of *LNCS*. Springer-Verlag, 2003 pp. 206–222.

[21] Huth M, Jagadeesan R, Schmidt DA. A Domain Equation for Refinement of Partial Systems. *Mathematical Structures in Computer Science*, 2004. **14**:469–505.

[22] Huth M, Pradhan S. Consistent Partial Model Checking. In: Proc. of the Workshop on Domains VI, volume 73 of *ENTCS*. Elsevier, 2004 pp. 45–85.

[23] Kouvaros P, Lomuscio A. Parameterised Verification of Infinite State Multi-Agent Systems via Predicate Abstraction. In: Proceedings of AAAI. 2017 pp. 3013–3020.

[24] Belardinelli F, Lomuscio A, Malvone V. Approximating Perfect Recall When Model Checking Strategic Abilities. In: Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018. 2018 pp. 435–444.

[25] Belardinelli F, Lomuscio A, Malvone V. An Abstraction-Based Method for Verifying Strategic Properties in Multi-Agent Systems with Imperfect Information. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019. 2019 pp. 6030–6037.

[26] Maluszynski J, Szalas A. Logical foundations and complexity of 4QL, a query language with unrestricted negation. *Journal of Applied Non-Classical Logics*, 2011. **21**(2):211–232. doi:10.3166/jancl.21.211-232.

[27] Maluszynski J, Szalas A. Partiality and Inconsistency in Agents' Belief Bases. In: Proceedings of KES-AMSTA, volume 252 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2013 pp. 3–17. doi:10.3233/978-1-61499-254-7-3.

[28] Ball T, Kupferman O. An Abstraction-Refinement Framework for Multi-Agent Systems. In: Proceedings of Logic in Computer Science (LICS). IEEE Computer Society, 2006 pp. 379–388. doi:10.1109/LICS.2006.10.

[29] Lomuscio A, Michaliszyn J. Verifying Multi-Agent Systems by Model Checking Three-valued Abstractions. In: Proc. of the 2015 Inter. Conf. on Autonomous Agents and Multiagent Systems, (AAMAS 2015). 2015 pp. 189–198.

[30] Belardinelli F, Lomuscio A. 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 2017. 2017 pp. 1259–1267.

[31] de Alfaro L, Faella M, Henzinger T, Majumdar R, Stoelinga M. Model checking discounted temporal properties. *Theoretical Computer Science*, 2005. **345**:139–170.

[32] Lluch-Lafuente A, Montanari U. Quantitative μ-Calculus and CTL Based on Constraint Semirings. *Electr. Notes Theor. Comput. Sci.*, 2005. **112**:37–59.

[33] Jamroga W. A Temporal Logic for Markov Chains. In: Proceedings of AAMAS'08. 2008 pp. 697–704.

[34] Jamroga W. A Temporal Logic for Stochastic Multi-Agent Systems. In: Proceedings of PRIMA'08, volume 5357 of *Lecture Notes in Computer Science*. 2008 pp. 239–250.

[35] Huth M, Kwiatkowska M. Quantitative Analysis and Model Checking. In: Logic in Computer Science. 1997 pp. 111–122.

[36] Baier C, Kwiatkowska M, Norman G. Computing Probability Bounds for Linear Time Formulas over Concurrent Probabilistic Systems. *Electronic Notes in Theoretical Computer Science*, 1999. **21**(19).

[37] Bulling N, Jamroga W. What Agents Can Probably Enforce. *Fundamenta Informaticae*, 2009. **93**(1-3):81–96.

[38] Huang X, Su K, Zhang C. Probabilistic Alternating-Time Temporal Logic of Incomplete Information and Synchronous Perfect Recall. In: Proceedings of AAAI-12. 2012 .

[39] Chen T, Forejt V, Kwiatkowska M, Parker D, Simaitis A. PRISM-games: A Model Checker for Stochastic Multi-Player Games. In: Proceedings of Tools and Algorithms for Construction and Analysis of Systems (TACAS), volume 7795 of *Lecture Notes in Computer Science*. Springer, 2013 pp. 185–191.

[40] Birkhoff G. Lattice Theory (2nd edition). American Mathematical Society, 1948.

[41] Davey BA, Priestley HA. Introduction to Lattices and Order. Cambridge University Press, 1990.

[42] Godefroid P, Jagadeesan R. Automatic Abstraction Using Generalized Model Checking. In: Proceedings of Computer Aided Verification (CAV), volume 2404 of *Lecture Notes in Computer Science*. Springer, 2002 pp. 137–150. doi:10.1007/3-540-45657-0_11.

[43] Cousot P, Cousot R. 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. doi:10.1145/512950.512973.

[44] Clarke E, Grumberg O, Long D. Model Checking and Abstraction. *ACM Transactions on Programming Languages and Systems*, 1994. **16**(5):1512–1542.

[45] Schobbens PY. Alternating-Time Logic with Imperfect Recall. *Electronic Notes in Theoretical Computer Science*, 2004. **85**(2):82–93.

[46] Hansson H, Jonsson B. A Logic for Reasoning about Time and Reliability. *Formal Asp. Comput.*, 1994. **6**(5):512–535. doi:10.1007/BF01211866.

[47] Jamroga W. Some Remarks on Alternating Temporal Epistemic Logic. In: Dunin-Keplicz B, Verbrugge R (eds.), Proceedings of Formal Approaches to Multi-Agent Systems (FAMAS 2003). 2003 pp. 133–140.

[48] Ågotnes T. A note on Syntactic Characterization of Incomplete Information in ATEL. In: Procedings of Workshop on Knowledge and Games. 2004 pp. 34–42.

[49] Jamroga W, van der Hoek W. Agents that Know how to Play. *Fundamenta Informaticae*, 2004. **63**(2–3):185–219.

[50] Ågotnes T. Action and Knowledge in Alternating-time Temporal Logic. *Synthese*, 2006. **149**(2):377–409.

[51] Jamroga W, Ågotnes T. Constructive Knowledge: What Agents Can Achieve under Incomplete Information. *Journal of Applied Non-Classical Logics*, 2007. **17**(4):423–475.

[52] Schnoor H. Strategic planning for probabilistic games with incomplete information. In: Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS). 2010 pp. 1057–1064.

[53] Jamroga W, Dix J. Model Checking Abilities of Agents: A Closer Look. *Theory of Computing Systems*, 2008. **42**(3):366–410.

[54] Bulling N, Dix J, Jamroga W. Model Checking Logics of Strategic Ability: Complexity. In: Dastani M, Hindriks K, Meyer JJ (eds.), Specification and Verification of Multi-Agent Systems, pp. 125–159. Springer, 2010.

[55] Dima C, Tiplea F. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. *CoRR*, 2011. **abs/1102.4225**.

[56] Bulling N, Jamroga W. Comparing Variants of Strategic Ability: How Uncertainty and Memory Influence General Properties of Games. *Journal of Autonomous Agents and Multi-Agent Systems*, 2014. **28**(3):474–518.

[57] Pilecki J, Bednarczyk M, Jamroga W. Synthesis and Verification of Uniform Strategies for Multi-Agent Systems. In: Proceedings of CLIMA XV, volume 8624 of *Lecture Notes in Computer Science*. Springer, 2014 pp. 166–182.

[58] Busard S, Pecheur C, Qu H, Raimondi F. Improving the Model Checking of Strategies under Partial Observability and Fairness Constraints. In: Formal Methods and Software Engineering, volume 8829 of *Lecture Notes in Computer Science*, pp. 27–42. Springer. ISBN 978-3-319-11736-2, 2014. doi:10.1007/978-3-319-11737-9_3.

[59] Huang X, van der Meyden R. Symbolic Model Checking Epistemic Strategy Logic. In: Proceedings of AAAI Conference on Artificial Intelligence. 2014 pp. 1426–1432.

[60] Cermak P, Lomuscio A, Mogavero F, Murano A. MCMAS-SLK: A Model Checker for the Verification of Strategy Logic Specifications. In: Proc. of CAV'14, volume 8559 of *LNCS*. Springer-Verlag, 2014 pp. 525–532.

[61] Jamroga W, Knapik M, Kurpiewski D, Mikulski L. Approximate verification of strategic abilities under imperfect information. *Artif. Intell.*, 2019. **277**. doi:10.1016/j.artint.2019.103172.

[62] Kurpiewski D, Knapik M, Jamroga W. On Domination and Control in Strategic Ability. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019. 2019 pp. 197–205.

[63] Jamroga W, Knapik M, Kurpiewski D. 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.

[64] Ågotnes T, Goranko V, Jamroga W, Wooldridge M. Knowledge and Ability. In: van Ditmarsch H, Halpern J, van der Hoek W, Kooi B (eds.), Handbook of Epistemic Logic, pp. 543–589. College Publications, 2015.

[65] Jamroga W. Logical Methods for Specification and Verification of Multi-Agent Systems. ICS PAS Publishing House, 2015. ISBN 978-83-63159-25-2.

[66] Jamroga W, Dix J. Model Checking ATL$_{ir}$ is Indeed Δ_2^P-complete. In: Proceedings of EUMAS, volume 223 of *CEUR Workshop Proceedings*. 2006 .

[67] Busard S. Symbolic Model Checking of Multi-Modal Logics: Uniform Strategies and Rich Explanations. Ph.D. thesis, Universite Catholique de Louvain, 2017.

[68] Mogavero F, Murano A, Vardi M. Reasoning About Strategies. In: Proceedings of FSTTCS. 2010 pp. 133–144.

[69] Mogavero F, Murano A, Perelli G, Vardi M. Reasoning About Strategies: On the Model-Checking Problem. *ACM Transactions on Computational Logic*, 2014. **15**(4):1–42.

[70] Berthon R, Maubert B, Murano A, Rubin S, Vardi MY. Strategy logic with imperfect information. In: Proceedings of LICS. 2017 pp. 1–12. doi:10.1109/LICS.2017.8005136.