Strategic and Epistemic Reasoning for the Game Description Language GDL-II

Ji Ruan and **Michael Thielscher** [1]

Abstract. The game description language GDL has been developed as a logic-based formalism for representing the rules of arbitrary games in general game playing. A recent language extension called GDL-II allows to describe nondeterministic games with any number of players who may have incomplete, asymmetric information. In this paper, we show how the well-known Alternating-time Temporal Epistemic Logic (ATEL) can be adapted for strategic and epistemic reasoning about general games described in GDL-II. We provide a *semantic* characterisation of GDL-II descriptions in terms of ATEL models. We also provide a *syntactic* translation of GDL-II descriptions into ATEL formulas, and we prove that these two characterisations are equivalent. We show that model checking in this setting is decidable by giving an algorithm, and we demonstrate how our results can be used to verify strategic and epistemic properties of games described in GDL-II.

1 Introduction

The general game description language GDL, which has been established as input language for general game-playing systems [5, 8], has recently been extended to GDL-II to incorporate games with nondeterministic actions and where players have incomplete/imperfect information [15]. We have previously analysed the epistemic logic behind GDL-II and in particular shown that the situation at any stage of a game can be characterised by a multi-agent epistemic (i.e., S5-) model [10]. However, this result merely provides a static characterisation of what players know (and don't know) at a certain stage. As such it cannot be used to reason about how players' knowledge evolves as the game progresses, nor does it allow to reason about the strategic ability of players to reach a desired state (possibly in cooperation with other players), etc. All these aspects presuppose the use of an underlying logic that goes beyond standard epistemic logic in that it combines both strategic and epistemic reasoning. Alternating-time Temporal Epistemic Logic (ATEL) [16], an extension to Alternating-time Temporal Logic (ATL) [1] with incomplete information, is such a formalism. For strategic reasoning alone, it has been shown that ATL can be applied to reason about complete-information games described in the original GDL using model checking methods [12]. Also, model checking for GDL is known to be EXPTIME-complete [12]. But unfortunately, the addition of incomplete information (and perfect recall) in GDL-II renders the model checking problem of ATL/ATEL undecidable [3].

In this paper we provide an adaption of ATEL for strategic and epistemic reasoning about general games described in GDL-II. Our main results are a characterisation of GDL-II descriptions in ATEL and a decidability result for the model checking problem. Specifically, we provide a *semantic* characterisation of GDL-II descriptions in terms of ATEL models and a *syntactic* translation of GDL-II descriptions into ATEL formulas, and we prove that these two characterisations are equivalent. We show that model checking in this setting is decidable and demonstrate how our results allow to verify strategic and epistemic properties of games described in GDL-II.

The paper is organised as follows. Section 2 gives preliminaries on GDL-II and ATEL. Section 3 presents a semantic characterisation of GDL-II using ATEL models and a syntactic mapping from GDL-II to ATEL formulas, along with the main equivalence result. Section 4 shows the decidability of model checking problem by giving an algorithm and discusses the strategic and epistemic properties that can be used for reasoning about GDL-II games. We conclude with a discussion on related work. Full proofs of our results can be found in an accompanying technical report [11].

2 Preliminaries

2.1 Game Description Language GDL-II

A complete game description consists of the names of (one or more) players, a specification of the initial position, the legal moves and how they affect the position, and the terminating and winning criteria. The emphasis of game description languages is on high-level, declarative game rules that are easy to understand and maintain. At the same time, GDL and its successor GDL-II have a precise semantics and are fully machine-processable. Moreover, background knowledge is not required—a set of rules is all a player needs to know to be able to play a hitherto unknown game. The description language GDL-II uses these *keywords*:

`role(?r)`	`?r` is a player
`init(?f)`	`?f` holds in the initial position
`true(?f)`	`?f` holds in the current position
`legal(?r,?m)`	`?r` can do move `?m`
`does(?r,?m)`	player `?r` does move `?m`
`next(?f)`	`?f` holds in the next position
`terminal`	the current position is terminal
`goal(?r,?v)`	goal value for role `?r` is `?v`
`sees(?r,?p)`	`?r` perceives `?p` in the next position
`random`	the random player

GDL (without `sees` and `random`) is suitable for describing finite, synchronous, and deterministic n-player games with complete information about the game state [8]. The extended game description language GDL-II allows to specify games with randomness and imperfect/incomplete information [15]. Valid game descriptions must satisfy certain syntactic restrictions, which ensure that all deductions

[1] School of Computer Science and Engineering, The University of New South Wales, Australia, email: {jiruan, mit}@cse.unsw.edu.au

"$\vdash$" used in the following definition are finite and decidable; for details we have to refer to [8] for space reasons.

A unique game model can be obtained from a valid GDL-II game description by using the notion of the *stable models* of logic programs with negation [4]. The syntactic restrictions in GDL-II ensure that all logic programs we consider have a *unique* and *finite* stable model [8, 15]. Hence, the game model for GDL-II has a finite set of players, finite states, and finitely many legal moves in each state [15]. By $G \vdash p$ we denote that ground atom p is contained in the unique stable model, denoted as $\text{SM}(G)$, for a stratified set of clauses G. In the following definition of a game model for GDL-II, *states* are identified with the set of atoms that are true in them.

Definition 1 (GDL-II Game Semantics)**.** *Let G be a valid GDL-II specification. The* semantics *of G is the state transition system $(R, s_0, t, l, u, \mathcal{I}, g)$ given by*

- *roles* $R = \{r \mid \texttt{role}(r) \in \text{SM}(G)\}$*;*
- *initial position* $s_0 = \text{SM}(G \cup \{\texttt{true}(f) \mid \texttt{init}(f) \in \text{SM}(G)\})$*;*
- *terminal positions* $t = \{s \mid \texttt{terminal} \in s\}$*;*
- *legal moves* $l = \{(r, m, s) \mid \texttt{legal}(r, m) \in s\}$*;*
- *state update function* $u(M, s) = \text{SM}(G \cup \{\texttt{true}(f) \mid \texttt{next}(f) \in \text{SM}(G \cup s \cup M^{\texttt{does}})\})$*, for all* joint *legal moves M (i.e., where each role in R takes one legal move) and states s;*[2]
- *information relation* $\mathcal{I} = \{(r, M, s, p) \mid \texttt{sees}(r, p) \in \text{SM}(G \cup s \cup M^{\texttt{does}})\}$*;*
- *goal relation* $g = \{(r, n, s) \mid \texttt{goal}(r, n) \in s\}$.

Note that a state s contains all ground atoms that are true in the state, which includes the "fluent atoms" $\texttt{true}(f)$ in, respectively, $\{\texttt{true}(f) \mid \texttt{init}(f) \in \text{SM}(G)\}$ (for the initial state) and $\{\texttt{true}(f) \mid \texttt{next}(f) \in \text{SM}(G \cup s \cup M^{\texttt{does}})\}$ (for the successor state of s and M), and all other atoms that can be derived from G *and* these fluent atoms.

Different runs of a game can be described by *developments*, which are sequences of states and moves by each player up to a certain round. A player *cannot distinguish* two developments if he makes the same moves and perceptions in the two.

Definition 2. *[15] Let $\langle R, s_0, t, l, u, \mathcal{I}, g\rangle$ be the semantics of a GDL-II description G, then a* development *δ is a finite sequence*

$$\langle s_0, M_1, s_1, \ldots, s_{d-1}, M_d, s_d\rangle$$

such that for all $i \in \{1, \ldots, d\}$ ($d \geq 0$), M_i is a joint move and $s_i = u(M_i, s_{i-1})$.

The length *of a development δ, denoted as len(δ), is the number of states in δ. By $M(j)$ we denote agent j's move in the joint move M. Let $\delta|_i$ be the prefix of δ up to length $i \leq$ len(δ).*

A player $j \in R \setminus \{random\}$ cannot distinguish *two developments $\delta = \langle s_0, M_1, s_1, \ldots\rangle$ and $\delta' = \langle s_0, M_1', s_1' \ldots\rangle$ (written as $\delta \sim_j \delta'$) iff* len(δ) = len(δ') *and for $i =$* len(δ) $- 1$:

- $\{p \mid (j, M_i, s_{i-1}, p) \in \mathcal{I}\} = \{p \mid (j, M_i', s_{i-1}', p) \in \mathcal{I}\}$,
- $M_i(j) = M_i'(j)$, *and* $\delta|_i \sim_j \delta'|_i$.

As an example, Figure 1 provides a GDL-II description of a card trading game adapted from [6]. In this game, player Bob (abbreviated as b) plays against the $random$ player (line 1). The deck consists of Ace, King and Queen (A, K, Q); it is assumed that A beats K, K beats Q, but Q beats A (lines 2–4). The game starts at round 0 (line 5). Lines 7–11 describe the possible moves for the players in different rounds: First, $random$ deals one card to b and one card to itself. Then b decides whether to trade his card for the one in the deck or to keep the current one. The player with the better card wins the game. Lines 13–20 define what will be true in the next state. For example, if $random$ does `deal(?X, ?Y)`, then b will hold `?X` in the next state (line 15). Lines 22–24 specify what b can see in the next state: if $random$ does `deal(?X, ?Y)`, then b will see that it holds card `?X` in the next state (line 22), and if $random$ holds `?X` in round 1, then b sees this in the next state (i.e., in round 2) (lines 23–24). Line 26 specifies when the game is terminal, and lines 27–31 give the goal values of player b (where $\texttt{goal}(b, 100)$ implies a win for b). The corresponding game model is sketched in Figure 2. Due to limited space, the $noop$ actions are not shown in the joint actions; e.g., for $\langle noop, deal(A, K)\rangle$ we only show $deal(A, K)$. States that player b cannot distinguish are connected by a dashed line.

```
1  role(b). role(random).
2  isDeal(A,K). isDeal(A,Q). isDeal(K,A). isDeal(K,Q).
3  isDeal(Q,A). isDeal(Q,K).
4  beats(A, K). beats(K, Q). beats(Q, A).
5  init(round(0)).
6
7  legal(random,deal(?X,?Y)) <= true (round(0)),isDeal(?X,?Y).
8  legal(b, noop) <= true (round(0)).
9  legal(random, noop) <= true (round(1)).
10 legal(b, trade) <= true (round(1)).
11 legal(b, keep) <= true (round(1)).
12
13 next(round(1)) <= true(round(0)).
14 next(round(2)) <= true(round(1)).
15 next(holds(b, ?X)) <= does(random, deal(?X,?Y)).
16 next(holds(random, ?Y)) <= does(random, deal(?X,?Y)).
17 next(holds(b, ?X))  <= does(b, keep), true(holds(b, ?X)).
18 next(holds(b, ?Z))  <= does(b, trade), true(holds(b, ?X)),
19   true(holds(random, ?Y)), isDeal(?Z, ?X), isDeal(?Z, ?Y).
20 next(holds(random, ?X)) <= true(holds(random, ?X)).
21
22 sees(b, holds(b,?X)) <= does(random, deal(?X,?Y)).
23 sees(b, holds(random, ?X)) <= true(holds(random, ?X)),
24   true(round(1)).
25
26 terminal <= true(round(2)).
27 goal(b, 100) <= true(holds(b, ?X)),
28   true(holds(random, ?Y)), beats(?X, ?Y).
29 goal(b, 0) <= true(holds(b, ?X)),
30   true(holds(random, ?Y)), beats(?Y, ?X).
31 win <= goal(b, 100)
```

Figure 1. A card trading game in GDL-II

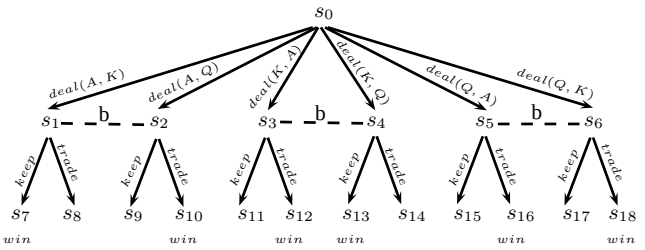


Figure 2. The game model for the card trading game.

2.2 ATEL with Finite Computations

Definition 3 (Language of ATEL, [16])**.** *The language of ATEL (with respect to a set of agents Ag and a set of atomic propositions Φ), is given by the following grammar:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \psi \mid \langle\langle X\rangle\rangle \bigcirc \varphi \mid \langle\langle X\rangle\rangle \Box \varphi \mid \langle\langle X\rangle\rangle \varphi \,\mathcal{U}\, \psi \mid K_i\varphi \mid C_X\varphi$$

where $p \in \Phi$ is an atomic proposition and $X \subseteq Ag$ is a set of agents. Other logic constants and connectives $\top, \bot, \vee, \rightarrow$ are defined as usual.

[2] For a joint move M where players $r_1, \ldots, r_k$ take moves $m_1, \ldots, m_k$, we define $M^{\texttt{does}} \stackrel{\text{def}}{=} \{\, \texttt{does}(r_1, m_1)., \ldots, \texttt{does}(r_k, m_k).\,\}$.

Intuitively, $\langle\langle X\rangle\rangle\bigcirc\varphi$ means a coalition of agents X can ensure that φ will hold in the next state; $\langle\langle X\rangle\rangle\Box\varphi$ means a coalition of agents X can ensure that φ will always hold; $\langle\langle X\rangle\rangle\varphi\,\mathcal{U}\,\psi$ means a coalition of agents X can ensure that φ will hold until ψ holds; $K_i\varphi$ means agent i knows φ; and $C_X\varphi$ means that φ is common knowledge among the agents in X. For example, "agent i knows that agent j knows p" can be expressed as K_iK_jp.

We will interpret ATEL formulas on finite tree-like structures (game trees) derived from GDL-II games (such as shown in Figure 2). A game is well-formed if it terminates after finite steps and if all players have at least one legal move in non-terminal states [8].[3] In addition, the setup of general game playing competitions is such that all agents are aware of the time progressing (*Synchronicity*) and remember what they have seen and done in the past (*Perfect Recall*) [15]. For this reason, we adapt the ATEL models from [16, 6] by adding an explicit set of terminal states. These terminal states correspond to those games states in which `terminal` holds.

Definition 4 (Model of ATEL)**.** *An* Action-based Alternating Epistemic Transition System *(*AAETS*) is a tuple*

$$\langle Q, s_0, T, Ag, \{Ac_i \mid i \in Ag\}, \rho, \tau, \{\sim_i : i \in Ag\}, \Phi, V\rangle$$

where

- Q *is a finite, non-empty set of* possible states*;*
- $s_0 \in Q$ *is the initial state;*
- $T \subseteq Q$ *is a non-empty set of terminal states;*
- Ag *is a finite, non-empty set of* n agents*;*
- Ac_i *is a finite, non-empty set of* actions *for each* $i \in Ag$*;*
- $\rho : Ac_{Ag} \to 2^Q$ *is an* action precondition function*, which for each action* $a \in Ac_{Ag}$*(=*$\bigcup_{i\in Ag} Ac_i$*) defines the set of states* $\rho(a)$ *from which* a *may be executed (for any action* a *it is assumed that* $T \cap \rho(a) = \emptyset$*, which means that no actions are possible in terminal states);*
- $\tau : Ac_1 \times \cdots \times Ac_n \times Q \to Q$ *is a partial* system transition function*, which defines the state* $\tau(M, s)$ *that would result from agents' actions* M *(*$= a_1, \ldots, a_n$*) on state* s*, given that* $s \in \rho(M(i))$ *for all agent* $i \in Ag$ *(i.e., agent* i*'s action* $M(i) = a_i$ *can be executed at state* s*);*
- Φ *is a finite, non-empty set of* atomic propositions*;*
- *each* $\sim_i \subseteq Q \times Q$ *is an equivalence relation (called the* accessibility relation*) for agent* i*;*
- $V : Q \mapsto 2^\Phi$ *is a* valuation function *that assigns each state a set of atomic propositions (said to be* true *in that state).*

A *computation* of an AAETS is a finite sequence of states $\lambda = s_0s_1...s_k \in Q^+$ such that for each $0 < u \leq k$, there is a joint action M' such that $s_u = \tau(M', s_{u-1})$. A computation λ starting in state s is referred to as an *s-computation*. If $0 \leq u < |\lambda|$ (the size of λ), then we denote by $\lambda[u]$ the u-th state in λ, by $\lambda[0, u]$ the finite prefix $s_0...s_u$ of λ, and by *last*(λ) the last state of λ.

In addition to finiteness, we stipulate the following three properties for our AAETS in accordance with the general game playing setting as discussed above.

Definition 5 (Tree, Synchronicity and Perfect Recall)**.** *An* AAETS $\mathcal{A}$ *has* tree property *iff any state* s *is reached from initial state* s_0 *via a unique computation. We denote such a computation by* $\lambda(s_0, s)$*.*

An AAETS $\mathcal{A}$ *has* synchronicity *iff for all* $s, t \in \mathcal{A}$ *and agents* i*,* $s \sim_i t$ *implies that the computations from the initial state* s_0 *to* s *(i.e.,* $\lambda(s_0, s)$*) and from* s_0 *to* t *(i.e.,* $\lambda(s_0, t)$*) have the same length, i.e.,* $|\lambda(s_0, s)| = |\lambda(s_0, t)|$*.*

An AAETS $\mathcal{A}$ *has* perfect recall *iff for all finite computations* $\lambda, \lambda' \in Q^+$ *and agents* i*,* $\lambda \sim_i \lambda'$ *implies that* $last(\lambda) \sim_i last(\lambda')$ *and* $\lambda[0, |\lambda| - 2] \sim_i \lambda'[0, |\lambda'| - 2]$*.*

Given an agent $i \in Ag$ and a state $s \in Q$, we denote the *options* available to i in s—the actions that i may perform in s—by $options(i, s) = \{m \mid m \in Ac_i \text{ and } s \in \rho(m)\}$. We then define a *perfect recall strategy* for an agent $i \in Ag$ to be a function $\sigma_i : Q^+ \to Ac_i$ that must satisfy

- the *legality* constraint that $\sigma_i(\lambda) \in options(i, last(\lambda))$ for all finite computations $\lambda \in Q^+$, and
- the *uniformity* constraint that for any two finite computations $\lambda_1, \lambda_2 \in Q^+$, if $\lambda_1 \sim_i \lambda_2$ then $\sigma_i(\lambda_1) = \sigma_i(\lambda_2)$.

A *perfect recall strategy* for a coalition $X = \{i_1, \ldots, i_k\} \subseteq Ag$ is a tuple of strategies $\langle\sigma_1, \ldots, \sigma_k\rangle$, one for each agent $i \in X$. We denote i's component of σ_X by σ_X^i. The *outcome* of applying a strategy for coalition X on a finite computation λ is defined as $out(\sigma_X, \lambda) = \{s \mid \exists M \text{ such that } M(i) = \sigma_X^i(\lambda) \text{ for } i \in X, \text{ and } s = \tau(M, last(\lambda)\}$.

Given a perfect recall strategy σ_X for some coalition X, and a state $s \in Q$, we define $comp(\sigma_X, s)$ to be the set of all finite computations that may occur if every agent $i \in X$ follows the corresponding strategy σ_i, starting when the system is in state $s \in Q$ and ending with a terminal state in T: $comp(\sigma_X, s) = \{\lambda \mid \lambda[0] = s, last(\lambda) \in T \text{ and } \forall 0 \leq u < |\lambda| - 1 : \lambda[u + 1] \in out(\sigma_X, \lambda[0, u])\}$.

Note that herein lies the major difference between our models and the models defined in other ATL/ATEL papers [1, 16, 6]. In our case, all the computations in $comp(\sigma_X, s)$ are finite in accordance with well-formed GDL-II games, whereas they are infinite in other papers. This results in the following modified semantics.

Definition 6 (Semantics of ATEL)**.** *For a finite* AAETS $\mathcal{A}$ *and a state* s*, let* $\lambda(s_0, s)$ *denote the finite computation starting from* s_0 *and ending with* s*. The truth of ATEL formulas is inductively defined as follows:*

$\mathcal{A}, s \models p$ *iff* $p \in V(s)$ *(where* $p \in \Phi$*);*
$\mathcal{A}, s \models \neg\varphi$ *iff* $\mathcal{A}, s \not\models \varphi$*;*
$\mathcal{A}, s \models \varphi \vee \psi$ *iff* $\mathcal{A}, s \models \varphi$ *or* $\mathcal{A}, s \models \psi$*;*
$\mathcal{A}, s \models \langle\langle X\rangle\rangle\bigcirc\varphi$ *iff* $\exists\sigma_X$*, such that* $\forall\lambda \in comp(\sigma_X, s)$ *we have* $|\lambda| > 1$ *and* $\mathcal{A}, \lambda[1] \models \varphi$*;*
$\mathcal{A}, s \models \langle\langle X\rangle\rangle\Box\varphi$ *iff* $\exists\sigma_X$*, such that* $\forall\lambda \in comp(\sigma_X, s)$ *we have* $\mathcal{A}, \lambda[u] \models \varphi$ *for all* $0 \leq u < |\lambda|$*;*
$\mathcal{A}, s \models \langle\langle X\rangle\rangle\varphi\,\mathcal{U}\,\psi$ *iff* $\exists\sigma_X$*, such that* $\forall\lambda \in comp(\sigma_X, s)$ *there exists some* $u < |\lambda|$ *such that* $\mathcal{A}, \lambda[u] \models \psi$*, and for all* $0 \leq v < u$ *we have* $\mathcal{A}, \lambda[v] \models \varphi$*;*
$\mathcal{A}, s \models K_i\varphi$ *iff* $\forall s'$ *such that* $\lambda(s_0, s) \sim_i \lambda(s_0, s')$*,* $\mathcal{A}, s' \models \varphi$*;*
$\mathcal{A}, s \models C_X\varphi$ *iff* $\forall s'$ *such that* $\lambda(s_0, s) \sim_X^* \lambda(s_0, s')$*,* $\mathcal{A}, s' \models \varphi$ *where* $\sim_X^*$ *is the transitive and reflexive closure of* $\cup_{i\in X} \sim_i$*.*

3 Mapping GDL-II into ATEL

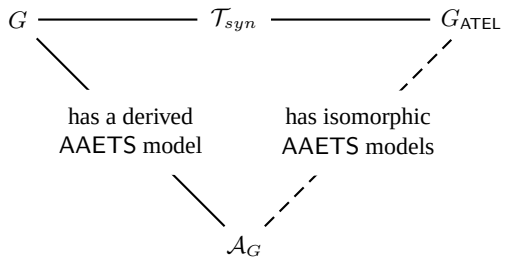
GDL-II serves as a language to describe games, while ATEL is for reasoning about such games. We build two links between GDL-II and ATEL:

- On the *semantic* level, every GDL-II description G induces an AAETS *model* $\mathcal{A}_G$.

[3] In general, termination is not guaranteed as GDL-II can describe games that run forever, but all games considered in general game playing competitions are required to be well-formed [5, 8]. Some games (such as TicTacToe) terminate naturally, and in other games (such as Chess) a step counter can be added to enforce termination after finitely many moves.

- On the *syntactic* level, every GDL-II description G can be translated into an ATEL *theory* G_{ATEL} (defined as $\mathcal{T}_{syn}(G)$ below).

Thus, we are able to interpret ATEL formulas φ over G either via ATEL semantics, i.e., define $G \models \varphi$ as $\mathcal{A}_G \models_{\mathsf{ATEL}} \varphi$. Or, we can use the syntactic characterisation, i.e., define $G \models \varphi$ as $\models_{\mathsf{ATEL}} \mathcal{T}_{syn}(G) \rightarrow \varphi$. As our main result, we will prove that these two characterisations are equivalent. The following diagram depicts the main idea, and we are now going to present it in detail.



Since ATEL does not support first-order predicates, we follow [13, 12] in applying a pre-processing step to GDL-II descriptions by replacing all predicates with variables, such as `isdeal(?X,?Y)`, by all relevant instances (also called ground atoms), such as `isdeal(A,K)`. This maps an arbitrary GDL-II description into an equivalent variable-free specification. We then translate such ground atoms to atomic propositions in ATEL.

Definition 7 (Translation t and t_{pre})**.** *Let $At_{GDL\text{-}II}$ be the set of ground atoms in GDL-II, and At_{ATEL} be the set of atomic propositions in ATEL. The translation t maps every GDL-II formulas to an ATEL formula as follows.*

Base case:

$$t(\mathtt{p}) \;=\; p \quad \textit{for all } \mathtt{p} \in At_{GDL\text{-}II}$$

where $p \in At_{\mathsf{ATEL}}$.

Extended cases:

$$\begin{array}{lcl} t(\texttt{not p}) & = & \neg t(\mathtt{p}); \\ t(\{\mathtt{p_1}, \ldots, \mathtt{p_k}\}) & = & \{t(\mathtt{p_1}), \ldots, t(\mathtt{p_k})\} \textit{ for all literals } \mathtt{p_i}. \end{array}$$

Note that a literal is either `p` *or* `not p` *for all* $\mathtt{p} \in At_{GDL\text{-}II}$*. Furthermore, let t_{pre} be defined by $t_{pre}(\mathtt{p}) = t(\mathtt{p})_{pre}$ for the base case (and similar to t for the extended cases). An atom p_{pre} will represent the value of atom p in the previous state (in ATEL). For convenience, we abbreviate $does(i,m)_{pre}$ by $done(i,m)$.*

We define how to induce an AAETS from a GDL-II description.

Definition 8 (Semantic interpretation of GDL-II in ATEL)**.** *Given a GDL-II description G with semantics $\langle R, s'_0, t, l, u, \mathcal{I}, g\rangle$, an* AAETS *for G (denoted as $\mathcal{A}_G$) is a tuple*

$$\langle Q, s_0, T, Ag, \{Ac_i \mid i \in Ag\}, \rho, \tau, \{\sim_i \mid i \in Ag\}, \Phi, V\rangle$$

where

- *Q is the set of states of G;*
- *$s_0 \in Q$ is the initial game state s'_0;*
- *T is the set of terminal states as t;*
- *Ag is the set of roles $R \setminus \{random\}$ (assume n agents);*
- *$Ac_i = \{m \mid (i,m,s) \in l, s \in Q\}$ is the set of moves of agent i;*
- *$\tau : Ac_1 \times \cdots \times Ac_n \times Q \mapsto Q$ is a partial function that maps a set of action and a state to another state such that $\tau(M,s) = u(M,s)$;*
- *$\sim_i \subseteq Q \times Q$ is the accessibility relation for agent $i \in Ag$ given by $(s,s') \in \sim_i$ (also written as $s \sim_i s'$) iff role i cannot distinguish any two developments δ and δ' such that $\delta = \langle s_0 \ldots s\rangle$ and $\delta' = \langle s_0 \ldots s'\rangle$ (cf. Definition 2);*
- *$\Phi \subseteq At_{\mathsf{ATEL}}$ is a set of atomic propositions translated by t and t_{pre} from the atoms in $At_{GDL\text{-}II}$;*
- *$V : Q \rightarrow 2^{\Phi}$ is an interpretation function which associates with each state s the set of atoms that satisfies the following requirements: if $\mathtt{p} \in s$ then $t(\mathtt{p}) \in V(s)$; if $s = u(M,s')$ then for any agent i we have $done(i, M(i)) \in V(s)$, and for any $\mathtt{p} \in s'$ we have $t_{pre}(\mathtt{p}) \in V(s)$; and $p_{init} \in V(s_0)$.*

The interpretation function requires that if there is a transition from s' to s, then the moves that were made on s' are recorded in $V(s)$ and the atoms true in s' are also recorded in s using the corresponding atoms labelled with pre. The accessibility relation of states is given according to the developments that end with such states. This ensures that $\mathcal{A}_G$ always has synchronicity and perfect recall.

Next, we give the full syntactic translation of GDL-II descriptions into ATEL formulas.

Definition 9 (Syntactic Translation $\mathcal{T}_{syn}$)**.** *Given a game description G, we define its ATEL theory $G_{\mathsf{ATEL}} = \mathcal{T}_{syn}(G)$ as a conjunction:*

$$\Gamma_{ini} \wedge \Gamma_{norm} \wedge \Gamma_{leg} \wedge \Gamma_{act} \wedge \Gamma_{mem} \wedge \Gamma_{next} \wedge \Gamma_{sees}.$$

These conjuncts are described in detail as follows.

- Γ_{ini}. The initial state is captured by a conjunction of: all the fluent atoms that are initially true plus the extra atomic proposition p_{init} (which is only true in s_0). Formally,

$$\Gamma_{ini} = \textstyle\bigwedge_{\mathtt{init(f)} \in G} true(f) \wedge (p_{init} \wedge \langle\langle\rangle\rangle\bigcirc\langle\langle\rangle\rangle\Box\neg p_{init}).$$

- Γ_{norm}. For normal rules (i.e., rules without any of the keywords `role`, `init`, `next` or `sees` in the head), we group those by heads:

$$\begin{array}{llll} r_1: & \mathtt{p} & \Leftarrow & bd_1 \\ \ldots & \ldots & \ldots & \ldots \\ r_k: & \mathtt{p} & \Leftarrow & bd_k \end{array}$$

Such rules decide whether `p` is true in the current state. Let $R_{\mathtt{p}} = \{r_1, ..., r_k\}$, H be the heads of all such rules, and AH be the set of atoms that do not appear in the heads of any rules and are not "`does`" atoms, then:

$$\Gamma_{norm} = \langle\langle\rangle\rangle\Box(\textstyle\bigwedge_{\mathtt{p}\in H}(\mathsf{CP}(R_{\mathtt{p}}) \wedge \mathsf{LF}(R_{\mathtt{p}})) \wedge \bigwedge_{\mathtt{p}\in AH} \neg t(\mathtt{p}))$$

where $\mathsf{CP}(R_{\mathtt{p}}) = t(\mathtt{p}) \leftrightarrow (\bigvee_{j\in[1..k]}(\bigwedge t(bd_j)))$, and $\mathsf{LF}(R_{\mathtt{p}})$ is a *loop* formula (see below). Notice that bd_i (a set of literals) is the body of rule i. Specially, if bd_i is empty, then $\bigwedge t(bd_j)$ is $\top$, and if `p` does not appear in the head of any rules and is not a "`does`" atom, then it must be false, which is captured by $\neg t(\mathtt{p})$.
Formula $\mathsf{CP}(R_{\mathtt{p}})$ applies Clark's completion [2] to a given set of GDL-II rules $R_{\mathtt{p}}$. An example from the card trading game is: $\mathsf{CP}(R_{\mathtt{win}}) = win \leftrightarrow goal(b, 100)$. But in general the semantics of the completion of a (stratified) logic program is too weak to fully characterise the standard model in the presence of redundant rules like $\mathtt{p} \Leftarrow \mathtt{p}$. The standard model remains the same when such "superfluous" clauses are added, but Clark's completion is weakened by them [7]. This issue is solved by a propositional formula denoted as $\mathsf{LF}(R_{\mathtt{p}})$ (which is also called a loop formula); we refer to [7] for a detailed algorithm to compute such a formula.

- Γ_{leg}. In all non-terminal states, each agent must make one legal move. This means that if $legal(i,m)$ is true in the current state, then agent i can enforce $done(i,m)$ to be true in a next state and on the other hand if $done(i,m)$ is true in the current state, then $legal(i,m)$, must be true in the previous state, i.e., $legal(i,m)_{pre}$ is true in the current state. This is captured by the following:
$\Gamma_{leg} = \langle\langle\rangle\rangle\Box(\neg terminal \rightarrow$
$\bigwedge_{i\in Ag, m\in Ac_i}(legal(i,m) \leftrightarrow \langle\langle i\rangle\rangle\bigcirc done(i,m))\wedge$
$\bigwedge_{i\in Ag, m\in Ac_i}(done(i,m) \rightarrow legal(i,m)_{pre})).$
- Γ_{act}. For all non-initial states, each agent should have done exactly one action in the previous state, and agents always know what they did:
$\Gamma_{act} = \langle\langle\rangle\rangle\Box(\neg p_{init} \rightarrow \bigwedge_{i\in Ag} \mathsf{XOR}_{m\in Ac_i} done(i,m))\wedge$
$\langle\langle\rangle\rangle\Box(\bigwedge_{i\in Ag, m\in Ac_i}(done(i,m) \leftrightarrow K_i done(i,m))).$
where XOR is the *exclusive OR* operator.
- Γ_{mem}. For non-terminal states, we use the atom $t_{pre}(\mathtt{p})$ to record the truth-value of $t(\mathtt{p})$ for it to be used in the next states. Let At be $At_{\text{GDL-II}}$ but without "`does`" atoms, then:
$\Gamma_{mem} = \langle\langle\rangle\rangle\Box(\neg terminal \rightarrow$
$\bigwedge_{\mathtt{p}\in At}((t(\mathtt{p}) \leftrightarrow \langle\langle\rangle\rangle\bigcirc t_{pre}(\mathtt{p})) \wedge (\neg t(\mathtt{p}) \leftrightarrow \langle\langle\rangle\rangle\bigcirc \neg t_{pre}(\mathtt{p}))).$
- Γ_{next}. Suppose these are all the rules with head $\mathtt{next(f)}$:

$$\begin{array}{llll} r_1: & \mathtt{next(f)} & \Leftarrow & bd_1 \\ \ldots & \ldots & \ldots & \ldots \\ r_k: & \mathtt{next(f)} & \Leftarrow & bd_k \end{array}$$

There are two alternatives for translating these rules into ATEL formulas. From the perspective of a *current state*, the truth of `f` in the next state is determined by the truth of the propositions in the bodies of rules $r_1, ..., r_k$ in the current state and the actions that are chosen by the agents for the transition. From the perspective of a *next state*, the truth of `f` in this state is determined by the previous truth of the propositions in the bodies of rules $r_1, ..., r_k$ and the actions that have just been done. We adopt the second perspective. Let HN be the heads of all such rules, and define
$\Gamma_{next} = \langle\langle\rangle\rangle\Box(\neg p_{init} \rightarrow$
$\bigwedge_{\mathtt{next(f)}\in HN}(true(f) \leftrightarrow (\bigvee_{j\in[1..k]}(\bigwedge t_{pre}(bd_j))))).$
- Γ_{sees}. The rules with "`sees`" are similar to those with "`next`", but instead of defining what will be true they specify what will be seen by the agents next. Suppose these are all the rules with head $\mathtt{sees(i,x)}$:

$$\begin{array}{llll} r_1: & \mathtt{sees(i,x)} & \Leftarrow & bd_1 \\ \ldots & \ldots & \ldots & \ldots \\ r_k: & \mathtt{sees(i,x)} & \Leftarrow & bd_k. \end{array}$$

Again we adopt the perspective of a next state. Let HS be the heads of all "`sees`" rules, and define
$\Gamma_{sees} = \langle\langle\rangle\rangle\Box(\neg p_{init} \rightarrow$
$\bigwedge_{\mathtt{sees(i,x)}\in HS}((K_i sees(i,x) \leftrightarrow (\bigvee_{j\in[1..k]}(\bigwedge t_{pre}(bd_j))))\wedge$
$(K_i\neg sees(i,x) \leftrightarrow \neg(\bigvee_{j\in[1..k]}(\bigwedge t_{pre}(bd_j)))))).$

Note that the size of the ATEL theory is polynomial in the size of a variable-free GDL-II description. Our translation is correct in the sense that the resulting ATEL formula is satisfiable in the ATEL model derived from the same game description.

Proposition 1 (Correctness)**.** *For a GDL-II game description G,*

$$\mathcal{A}_G \models G_{\mathsf{ATEL}}.$$

To show that $\mathcal{T}_{syn}$ is an adequate syntactic characterisation of GDL-II descriptions, we define an equivalence relation on AAETSs.

Definition 10 (AAETS Isomorphism)**.** *Let $\mathcal{A} = \langle Q, s_0, T, Ag, \{Ac_i \mid i \in Ag\}, \rho, \tau, \{\sim_i \mid i \in Ag\}, \Phi, V\rangle$ and $\mathcal{A}' = \langle Q', s_0', T', Ag, \{Ac_i' \mid i \in Ag\}, \rho', \tau', \{\sim_i' \mid i \in Ag\}, \Phi, V'\rangle$ be two* AAETS*s, then they are isomorphic (denoted as $\mathcal{A} \cong \mathcal{A}'$) iff there is a function* f *such that:*

- f *maps every state in Q to a state in Q' and it is a bijection; in particular $\mathtt{f}(s_0) = s_0'$, and for all $s \in T$, $\mathtt{f}(s) \in T'$;*
- f *maps every acton in Ac_i to an action in Ac_i' and it is a bijection;*
- *for every state s and action m, $s \in \rho(m)$ iff $\mathtt{f}(s) \in \rho'(\mathtt{f}(m))$;*
- *for every state s, s' and joint action M, $s = \tau(M, s')$ iff $\mathtt{f}(s) = \tau'(\mathtt{f}(M), \mathtt{f}(s'))$;*
- *for every state s and agent i, $s \sim_i s'$ iff $\mathtt{f}(s) \sim_i' \mathtt{f}(s')$;*
- *for every proposition p and state $s \in Q$, $p \in V(s)$ iff $p \in V'(\mathtt{f}(s))$.*

The existence of an isomorphism between two AAETSs implies that they satisfy the same formulas.

Proposition 2. *Given two* AAETS*s $\mathcal{A}$ and $\mathcal{A}'$ along with an arbitrary ATEL formula φ,*

$$\mathcal{A} \cong \mathcal{A}' \text{ implies } (\mathcal{A} \models \varphi \text{ iff } \mathcal{A}' \models \varphi).$$

Proposition 3. *Let G be a game description and φ an ATEL formula, then the following holds*

$$\models G_{\mathsf{ATEL}} \rightarrow \varphi \text{ iff } \mathcal{A}_G \models \varphi.$$

Proof. (Sketch) The direction from left to right follows from Proposition 1. The direction from right to left is proved by showing that for any AAETS $\mathcal{A}$ with synchronicity and perfect recall such that $\mathcal{A} \models G_{\mathsf{ATEL}}$, there is an isomorphism between $\mathcal{A}$ and $\mathcal{A}_G$; the result then follows from Proposition 2. See [11] for details. □

This is a main result in this paper. It shows that G_{ATEL} completely characterises G in ATEL in the sense that it entails any formula that is satisfied in the AAETS derived from G directly, and vice versa.

4 Model Checking Strategic and Epistemic Properties

Our main result in this paper allows us to consider the following model checking problem: *given a game represented by GDL-II, and a property represented by an ATEL formula, decide whether the property is true for the game description*. If the agents have incomplete information and perfect recall, the model checking problem for ATEL in traditional semantics is undecidable (see [3]). Hence, had we used the standard semantics, the above problem would also be undecidable since we can reduce the problem by deriving an ATEL model from a GDL-II description and then perform the ATEL model checking. However, with our new ATEL semantics—over AAETS models with *finite* computations—the model checking problem becomes decidable as we can give an algorithm for it.

We only sketch the algorithm below for the case of $\langle\langle X\rangle\rangle\bigcirc\varphi$ due to space limitations. This algorithm terminates because only a finite number of strategies and computations needs to be checked.

```
mcheck(A, s, <<X>> O Phi) {
found := True;
foreach sigma(X) do {
  foreach c in comp(sigma(X), A, s) do {
  if |c|=1 or mcheck(A,c[1],Phi) == False
  then found := False;                  }
if found == True then return True; }
return False;  }
```

How complex is the problem? We know for sure that the problem is at least 2EXPTIME-hard due to [9], which shows that the outcome problem of the Private-PEEK game is complete in double exponential time (2EXPTIME-complete). The Private-PEEK game can be rather straightforwardly specified in GDL-II and the outcome problem can be equivalently expressed as deciding whether $\langle\langle 1\rangle\rangle \Diamond win_i$ is true in the initial state of the game. Finding an upper bound for the complexity, however, is left for future work.

In [12], ATL is used to characterise some interesting playability properties for the original GDL games. With ATEL being the language to express properties for GDL-II games, we can now not only express the above properties but also a new class of properties that are not expressible in ATL. We discuss two kinds of such properties.

Coherence Knowledge Properties. There are some properties φ that involve pure knowledge, i.e., where no temporal modalities occur in φ. For such φ, we call $\langle\langle\rangle\rangle \Box \varphi$ a *coherence knowledge property*. We know that $\mathcal{A}, s_0 \models \langle\langle\rangle\rangle \Box \varphi$ iff φ is true in all reachable states from s_0.

In GDL-II, agents may not always know their legal moves. In order to check this, we can express the property that "if a move is legal for an agent then the agent knows it" as a formula:

$$\langle\langle\rangle\rangle \Box \bigwedge_{i \in Ag, m \in Ac_i} (legal(i,m) \rightarrow K_i legal(i,m)).$$

The following is not necessarily true for a GDL-II game: if the game has terminated, then this is common knowledge,

$$\langle\langle\rangle\rangle \Box (terminal \rightarrow C_{Ag} terminal).$$

If we want to ensure that a GDL-II description G has this property, then we can either verify $\mathcal{A}_G, s_0 \models \langle\langle\rangle\rangle \Box (terminal \rightarrow C_{Ag} terminal)$, or prove

$$\models G_{\mathsf{ATEL}} \rightarrow \langle\langle\rangle\rangle \Box (terminal \rightarrow C_{Ag} terminal).$$

Properties with knowledge and strategic power. This class of properties mixes knowledge and coalition modalities, allowing us to talk about the agent's knowledge and power simultaneously.

The following property says that if i has a winning strategy, then he knows it:

$$\langle\langle i\rangle\rangle \Diamond win_i \rightarrow K_i \langle\langle i\rangle\rangle \Diamond win_i.$$

Suppose that in the current state $\mathcal{A}, s \models K_i \langle\langle i\rangle\rangle \Diamond win_i$. This does not give a winning strategy for agent i explicitly. But agent i can check the following on any state that he cannot distinguish from s (i.e., $s \sim_i s'$):

$$\mathcal{A}, s' \models \langle\langle i\rangle\rangle \bigcirc (done(i,m) \wedge \langle\langle i\rangle\rangle \Diamond win_i).$$

If the above holds, then agent i can safely choose $does(i,m)$, and it still guarantees him a winning position in the next state.

We conclude with the following strategic property, which says that if i knows φ, then he can ensure that agent j knows ψ next:

$$K_i \varphi \rightarrow \langle\langle i\rangle\rangle \bigcirc K_j \psi.$$

5 Related Work and Conclusion

There are just a few papers on reasoning about games in GDL and its extension GDL-II. In [12], a method based on ATL is given to verify properties of general games, but this is restricted to original GDL and hence to games where players can maintain complete state information. Our paper extends this approach to GDL-II and uses a version of ATEL for this purpose. Our characterisation formula is inspired by the one given in [12], but we make these improvements: (1) we can deal with imperfect-information games; (2) we show that the models that satisfy this formula are isomorphic to $\mathcal{A}_G$, rather than a weaker relation as alternating bisimulation given in [12]; (3) we do not require an extra "sink" state as there is no need to make computations infinite with our new semantics.

In [14], it is shown how GDL-II can be formally translated into the Situation Calculus as a first-order axiomatisation that allows players to reason about their percepts and what they know about the legality and effects of moves based on the game description. In [10], the epistemic structure and expressiveness of GDL-II is analysed in terms of epistemic modal logic. It was shown that the operational semantics of GDL-II entails that the situation at any stage of a game can be characterised by a multi-agent epistemic (i.e., S5-) model and GDL-II is sufficiently expressive to model any situation that can be described by a (finite) multi-agent epistemic model. Our work extends the static epistemic model into a dynamic AAETS, and therefore a lager class of strategic and epistemic properties can be addressed by our approach.

Acknowledgements We thank the anonymous reviewers for their helpful comments. This research was supported under Australian Research Council's (ARC) *Discovery Projects* funding scheme (project DP 120102023). Michael Thielscher is the recipient of an ARC Future Fellowship (project FT 0991348). He is also affiliated with the University of Western Sydney.

REFERENCES

[1] R. Alur, T. A. Henzinger, and O. Kupferman, 'Alternating-time temporal logic', *Journal of the ACM*, **49**(5), 672–713, (September 2002).
[2] K. Apt, 'Introduction to logic programming', in *Handbook of Theoretical Computer Science*, ed., J. van Leeuwen, 494–574, Elsevier, (1990).
[3] C. Dima and F. L. Ţiplea, 'Model-checking ATL under imperfect information and perfect recall semantics is undecidable', Research Report 1102.4225, arXiv, (February 2011).
[4] M. Gelfond and V. Lifschitz, 'The stable model semantics for logic programming', in *Proceedings of IJCSLP*, eds., R. Kowalski and K. Bowen, pp. 1070–1080, Seattle, OR, (1988). MIT Press.
[5] M. Genesereth, N. Love, and B. Pell, 'General game playing: Overview of the AAAI competition', *AI Magazine*, **26**(2), 62–72, (2005).
[6] W. Jamroga and W. van der Hoek, 'Agents that know how to play', *Fundam. Inform.*, **63**(2-3), 185–219, (2004).
[7] F. Lin and Y. Zhao, 'Assat: computing answer sets of a logic program by sat solvers', *Artif. Intell.*, **157**(1-2), 115–137, (2004).
[8] N. Love, T. Hinrichs, D. Haley, E. Schkufza, and M. Genesereth, 'General Game Playing: Game Description Language Specification', Technical Report LG–2006–01, Stanford University, (2006).
[9] John H. Reif, 'The complexity of two-player games of incomplete information', *J. Comput. Syst. Sci.*, **29**(2), 274–301, (1984).
[10] J. Ruan and M. Thielscher, 'The epistemic logic behind the game description language', in *Proceedings of AAAI*, pp. 840–845, (2011).
[11] J. Ruan and M. Thielscher, 'Strategic and Epistemic Reasoning for the Game Description Language GDL-II: the Technical Report', CSE-TR-201213, University of New South Wales (2012).
[12] J. Ruan, W. van der Hoek, and M. Wooldridge, 'Verification of games in the game description language', *Journal of Logic and Computation*, **19**(6), 1127–1156, (2009).
[13] S. Schiffel and M. Thielscher, 'Fluxplayer: A successful general game player', in *Proceedings of AAAI*, pp. 1191–1196, (2007).
[14] S. Schiffel and M. Thielscher, 'Reasoning about general games described in GDL-II', in *Proceedings of AAAI*, pp. 846–851, (2011).
[15] M. Thielscher, 'A general game description language for incomplete information games', in *Proceedings of AAAI*, pp. 994–999, (2010).
[16] W. van der Hoek and M. Wooldridge, 'Time, knowledge, and cooperation: Alternating-time temporal epistemic logic and its applications', *Studia Logica*, **75**(1), 125–157, (2003).