J. Ruan
M. Thielscher

Logical-Epistemic Foundations of General Game Descriptions

Abstract. A *general* game player automatically learns to play arbitrary new games solely by being told their rules. For this purpose games are specified in the general *Game Description Language* (GDL), a variant of Datalog with function symbols that uses a few game-specific keywords. A recent extension of basic GDL allows the description of nondeterministic games with any number of players who may have incomplete, asymmetric information. In this paper, we analyse the epistemic structure and expressiveness of this language in terms of modal epistemic logic and prove two main results: (1) The operational semantics of GDL entails that the situation at any stage of a game can be characterised by a multi-agent epistemic (i.e., S5-) model; (2) GDL is sufficiently expressive to model any situation that can be described by a (finite) multi-agent epistemic model.

Keywords: Logical formalizations of game properties, General game playing, Epistemic logic.

1. Introduction

General game playing aims at building systems that automatically learn to play arbitrary new games solely by being told their rules [16, 15]. The Game Description Language (GDL) is a special purpose declarative language for defining games [13]. GDL is used in the AAAI General Game Playing Competition, where participants are provided with a previously unknown game specified in this language, and are required to dynamically and autonomously determine how best to play this game [6]. A recent extension to GDL allows the description of games that include random elements and incomplete information [21]. This opens the door to nondeterministic games in which players have incomplete and asymmetric information, as in Poker, Kriegspiel [17], or games which involve private communication among cooperating players like in Bughouse Chess[1] or negotiations like in Diplomacy.

The game description language is a variant of Datalog with function symbols that uses a few pre-defined keywords. By applying a standard semantics for logic programs, a game description G can be interpreted as a state transition system. The execution model underlying GDL then determines a *game*

[1] Bughouse is a chess variant played on two chessboards by four players in teams of two; for details see [17].

Special Issue: Logic and Games
Edited by **Thomas Ågotnes**

model for G, which defines all possible ways in which the game may develop and what information the players acquire as the game proceeds [13, 21]. However, an open question has been to what extent this game model, including its implicit epistemic structure due to incomplete and asymmetric information, satisfies the standard properties of epistemic logic, and how expressive GDL is in terms of which epistemic situations can be modelled as games in this language. The latter is particularly interesting because at first glance GDL seems to be constrained by the fact that all players have perfect knowledge of the game rules and in particular the initial position [21].

In this paper we analyse the epistemic structure and expressiveness of GDL in terms of standard epistemic logic. Seminal work in this area are [25, 8], and since then philosophers have developed the notions of knowledge and belief using Kripke's possible world semantics [10]. In the late 1980s these approaches were picked up and further developed by computer scientists, cf. [7, 3]. This development was originally motivated by the need to reason about communication protocols. One is typically interested in what knowledge different parties to a protocol have before, during and after a run (an execution sequence) of the protocol. Apart from computer science, there is much interest in the temporal dynamics of knowledge and belief in areas as diverse as artificial intelligence [14], multi-agent systems [18, 23], and game theory [2].

We present, and formally prove, two main results:

1. The game model for any (syntactically valid) GDL game entails that at any valid round of the game the situation that arises can be characterised by a multi-agent S5-model.
2. Given an arbitrary (finite) epistemic model it is possible to construct a GDL game description which produces the situation described by this model.

This is complemented by an analysis of entailment of epistemic formulas in GDL.

The remainder of the paper proceeds as follows. Section 2 recapitulates both game descriptions and epistemic logic. Section 3 analyses the entailment of epistemic formulas in GDL and shows how the situations that arise during a game can always be characterised by a standard epistemic model that entails the exact same formulas. Section 4 provides the construction of a GDL game for any given epistemic model. We conclude with a discussion of related and further work.

`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

Table 1. GDL-II keywords. Standard GDL comprises the top eight while the last two are added in view of incomplete state knowledge and elements of chance. The keywords are accompanied by the auxiliary, pre-defined predicate `distinct(X,Y)`, meaning the syntactic inequality of the two arguments [13].

2. Preliminaries

2.1. Describing Games with GDL-II

General Game Playing requires a formal language for describing the rules of arbitrary games. A complete game description consists of the names of the 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 the game description language GDL is on high-level, declarative game rules that are easy to understand and maintain. At the same time, GDL has a precise semantics and is fully machine-processable. Moreover, background knowledge is not required—a set of rules is all a player needs to know in order to be able to play a hitherto unknown game.

A variant of Datalog with function symbols, the game description language uses a few known *keywords* (cf. Table 1). GDL is suitable for describing finite, synchronous, and deterministic n-player games with complete information about the game state [13].[2] The extended game description language GDL-II (for: *GDL with incomplete information*)[3] allows the specification of

[2]Synchronous means that all players move simultaneously. In this setting, turn-taking games are modelled by allowing players only one legal move, without effect, if it is not their turn.

[3]A word on an unfortunate clash of terminology: in AI, an agent who does not know the full state of the environment is said to have *incomplete* information; in Game Theory, when a player does not know the full state when called upon to move, the game is said to be of *imperfect* information. We decided to stick with the standard AI terminology.

games with randomness and incomplete information [21]. In the following, we assume the reader to be familiar with basic notions and notations of logic programming, as can be found in e.g. [11]. The interested reader may take a peek at Figure 1 at this point to see an example of a GDL-II specification.

DEFINITION 1. A *valid GDL-II specification* is a finite set of clauses G where

- `role` only appears as fact or in clause bodies;
- `init` only appears as head of clauses and does not depend on any of `true`, `legal`, `does`, `next`, `terminal`, or `goal`;
- `true` only appears in clause bodies;
- `does` only appears in clause bodies, and none of `legal`, `terminal`, or `goal` depends on `does`;
- `next` and `sees` only appear as head of clauses;
- `distinct` only appears in clause bodies;[4]
- there are no cycles involving a negative edge in the dependency graph[5] for G; that is, G must be stratified [1, 4].
- each variable in a clause occurs in at least one positive atom in the body; that is, in the jargon of logic programming, G must be allowed [12].
- If p and q occur in a cycle in the dependency graph and G contains a clause $p(s_1, \ldots, s_m)$<= $q_1(\vec{t_1}), \ldots, q(v_1, \ldots, v_k), \ldots, q_n(\vec{t_n})$, then for every $v_i \in \{v_1, \ldots, v_k\}$,
  - v_i is ground, or
  - $v_i \in \{s_1, \ldots, s_m\}$, or
  - v_i is an element of some $\vec{t_j}$ $(1 \leq j \leq n)$ such that q_j does not appear in a cycle with p.

  This last condition imposes a restriction on the combination of function symbols and recursion to ensure decidability of all relevant derivations [13].

These syntactic restrictions are imposed in order to ensure that a set of GDL-II rules can be effectively and unambiguously interpreted by a state transition system as a formal game model, as follows.

---

[4]The meaning of this predicate is given by assuming the unary clause `distinct`(s, t)., for every pair s, t of syntactically different ground (i.e., variable-free) terms.

[5]The nodes of the dependency graph for G are the relation constants in the vocabulary. There is an edge from r_2 to r_1 whenever there is a rule with r_1 in the head and r_2 in the body. That edge is labeled with the negation symbol $\neg$ whenever r_2 is in a negative literal.

2.2. GDL-II Semantics

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 [5].

DEFINITION 2. Given a set of clauses G and an interpretation I (i.e., a set of ground atoms), let G^I be the set of negation-free implications h <= $b_1 \wedge \ldots \wedge b_k$ obtained by taking all ground instances of clauses in G and

- deleting all clauses with a negative body literal $\neg b_i$ such that $b_i \in I$,
- deleting all negative body literals from the remaining clauses.

Then I is a *stable model* for G if and only if I is the least model for G^I.

A useful property of stable models is that they provide a unique model whenever the underlying set of clauses is stratified [5], as is always the case in GDL-II. In the following, by $G \vdash p$ we denote that ground atom p is contained in this unique standard model for a stratified set of clauses G. The syntactic restrictions in GDL-II ensure that all logic programs we consider have a *unique* and *finite* stable model [13]. Hence, for the following game model underlying GDL-II we assume a finite set of players, finite states, and finitely many legal moves in each state.

Specifically, then, the derivable instances of `role(?r)` from a given game description define the players. The initial state is composed of the derivable instances of `init(?f)`. In order to determine the legal moves in any given state, this state has to be encoded first, using the keyword `true`. Let, to this end, $S = \{f_1, \ldots, f_n\}$ be a state (i.e., a finite set of ground terms), then the game rules G are augmented by the n facts

$$S^{\texttt{true}} \stackrel{\text{def}}{=} \{\, \texttt{true}(f_1). \quad \ldots \quad \texttt{true}(f_n).\}$$

Those instances of `legal(?r,?m)` that are derivable from $G \cup S^{\texttt{true}}$ define all legal moves `M` for player `R` in position S. In the same way, the clauses for `terminal` and `goal(?r,?n)` define termination and goal values *relative* to the encoding of a given position.

Determining a position update and the percepts of the players requires the encoding of both the current position and a *joint move*. Suppose joint move M is such that players $r_1, \ldots, r_k$ make moves $m_1, \ldots, m_k$, then let

$$M^{\texttt{does}} \stackrel{\text{def}}{=} \{\, \texttt{does}(r_1, m_1). \quad \ldots \quad \texttt{does}(r_k, m_k).\, \}$$

The instances of `next(?f)` derivable from $G \cup M^{\texttt{does}} \cup S^{\texttt{true}}$ compose the updated position; likewise, the derivable instances of `sees(?r,?p)` describe what a player perceives when the given joint move is made in the given position.

All of the above is summarised in the following definition.

DEFINITION 3. [21] Let G be a valid GDL specification. The *semantics* of G is the state transition system $(R, s_0, t, l, u, \mathcal{I}, g)$ given by

- roles $R = \{r : G \vdash \texttt{role}(r)\}$;
- initial position $s_0 = \{f : G \vdash \texttt{init}(f)\}$[6];
- terminal positions $t = \{S : G \cup S^{\texttt{true}} \vdash \texttt{terminal}\}$;
- legal moves $l = \{(r, m, S) : G \cup S^{\texttt{true}} \vdash \texttt{legal}(r, m)\}$;
- state update function $u(M, S) = \{f : G \cup M^{\texttt{does}} \cup S^{\texttt{true}} \vdash \texttt{next}(f)\}$[7], for all joint moves M (i.e., one for each role in R) and states S;
- information relation $\mathcal{I} = \{(r, M, S, p) : G \cup M^{\texttt{does}} \cup S^{\texttt{true}} \vdash \texttt{sees}(r, p)\}$;
- goal relation $g = \{(r, n, S) : G \cup S^{\texttt{true}} \vdash \texttt{goal}(r, n)\}$.

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, and a player *cannot distinguish* two developments if he makes the same moves and perceptions in the two.

DEFINITION 4. [21] 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 sequence

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

such that

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

The *length* of a development δ, denoted as $len(\delta)$, is the number of states in δ, and $M(j)$ denotes agent j's move in the joint move M.

A player $j \in R \setminus \{random\}$[8] *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

[6]Note that only f, not $\texttt{init}(f)$, can be contained in states.

[7]Note that only f, not $\texttt{next}(f)$, can be contained in states.

[8]The random player acts randomly and thus increases the uncertainty of the other agents. We do not usually consider its ability to distinguish two developments.

- $len(\delta) = len(\delta')$
- for all $i \in \{1, \ldots, len(\delta) - 1\}$:
  - $\{p : (j, M_i, s_{i-1}, p) \in \mathcal{I}\} = \{p : (j, M'_i, s'_{i-1}, p) \in \mathcal{I}\}$
  - $M_i(j) = M'_i(j)$.

This definition captures two features of general game playing competitions: all agents are aware of the time progressing (*Synchronicity*) and remember what they have seen and done in the past (*Perfect Recall*). Note that perfect recall does not mean that if an agent sees an atom p, it will keep seeing this. Also in general, the persistence of facts in the course of developments is only possible if there are rules (frame axioms) to guarantee it.

2.3. Modal Epistemic Logic

In order to analyse the epistemic logic behind GDL-II and its semantics, we recapitulate basic notions from standard Modal Epistemic Logic [3].

Definition 5. (Language) A basic Modal Epistemic Logic Language for *epistemic formulas* is given by the following Backus-Naur Form:

$$\phi := P \mid \neg\phi \mid (\phi \wedge \phi) \mid K_i\phi \mid C_X\phi$$

where P is an atomic proposition, i is an agent, and X is a non-empty set of agents. $\top, \bot, \vee, \rightarrow$ are defined as usual.

Intuitively, $K_i\phi$ means agent i knows ϕ, and $C_X\phi$ means that ϕ is common knowledge among the agents in X; for example, "agent k knows that agent j knows P" can be expressed as K_iK_jP. To give precise meanings to this language, we use multi-agent epistemic models.

Definition 6. A *multi-agent epistemic model* E is a structure $\langle W, \{\sim_i : i \in Ag\}, V\rangle$, where W is a set of *possible worlds*, Ag is a set of *agents*, each $\sim_i \subseteq W \times W$ is an equivalence relation (called the *accessibility relation*)[9] for agent i, and $V : W \mapsto 2^{Atoms}$ is a *valuation function* that assigns each world a set of atomic propositions (said to be *true* in that world).

Definition 7. Given an epistemic model E and an epistemic formula ϕ, the entailment relation $\models$ is defined as follows:

[9]Note that in general the accessibility relation $\sim_i$ in E does not have to be an equivalence relation, but since epistemic models in this paper are meant to represent the knowledge of agents—rather than belief or other modalities—we restrict $\sim_i$ to be an equivalence relation; hence all epistemic models are *S5-models* [3].

- $E, w \models P$ iff $P \in V(w)$;
- $E, w \models \neg\phi$ iff $E, w \not\models \phi$;
- $E, w \models \phi \wedge \psi$ iff $E, w \models \phi$ and $E, w \models \psi$;
- $E, w \models K_i\phi$ iff for all w', if $w \sim_i w'$ then $E, w' \models \phi$;
- $E, w \models C_X\phi$ iff for all w', if $w \sim_X w'$ then $E, w' \models \phi$.

where $\sim_X$ is the transitive and reflexive closure of $\cup_{i\in X} \sim_i$.

3. From GDL-II to Epistemic Models

This section relates the game descriptions in GDL-II to epistemic models so that we can reason about these games using the modal epistemic logic presented in the previous section.

The choice of S5-models is based on our intention to model the knowledge of players, which is defined via the notion of *indistinguishable worlds*: agent i cannot distinguish two worlds if and only if i observes the same information in these two worlds; in other words, if agent i *knows* ϕ, then ϕ must be true in all worlds that agent i cannot distinguish from the current world. GDL-II itself only allows us to talk about factual knowledge of agents, e.g., "agent j `sees` P." It does not allow us to talk about an agent's knowledge or higher-order knowledge (the knowledge about the knowledge of agents), as in, "agent i knows ϕ" or "agent i knows that agent j knows ϕ." The epistemic language of modal logic S5 bridges this gap.

Meanwhile, there is certain information commonly known by all agents. Specifically, in general game playing the game description itself is such common knowledge among all agents. More precisely, not only do all agents *know* the game description they are going to play, but also they *know* that each other player *knows* this, and so on. This is implicit in the execution model for GDL, as the Game Master makes sure that every agent gets the same game description before starting the game. Accordingly, the initial state of a game is common knowledge as well. This motivates our use of the $C_X\phi$ operator in the epistemic language (cf. Definition 5), which allows us to reason about such knowledge explicitly.

Before we present our results in all technical detail, we introduce a running example adopted from [3] to illustrate that GDL-II is indeed expressive enough to allow for modelling complex epistemic situations.

```
role(generalA).  role(generalB).  role(random).
succ(0,1).  succ(1,2).  ...  succ(8,9).
time(3am).  time(9pm).

init(round(0)).

gets_message(?g,?m)  <= role(?g), does(?g1,send(?m)), does(random,pass),
                        distinct(?g,?g1), distinct(?g,random).
gets_new_message(?g) <= gets_message(?g,?m).
has_a_message(?g)    <= true(message(?g,?m)).

legal(random,noop)          <= true(round(0)).
legal(random,pass)          <= not true(round(0)).
legal(random,stop)          <= not true(round(0)).
legal(generalA,settime(?t)) <= true(round(0)), time(?t).
legal(generalB,noop)        <= true(round(0)).
legal(generalA,send(?t))    <= true(round(1)), true(attack(?t)).
legal(?g,noop)              <= true(control(?g)),
                               not has_a_message(?g).
legal(?g,send(ack(?m)))     <= true(control(?g)), true(message(?g,?m)).
legal(generalA,noop)        <= true(control(generalB)).
legal(generalB,noop)        <= true(control(generalA)).

sees(?g,?m)                <= gets_message(?g,?m).
next(message(generalA,?t)) <= does(generalA,settime(?t)).
next(attack(?t))           <= does(generalA,settime(?t)).
next(attack(?t))           <= true(attack(?t)).
next(message(?g,?m))       <= gets_message(?g,?m).
next(message(?g,?m))       <= true(message(?g,?m)),
                              not gets_new_message(?g).
next(control(generalA))    <= true(round(0)).
next(control(generalA))    <= true(round(1)).
next(control(generalA))    <= true(control(generalB)).
next(control(generalB))    <= true(control(generalA)),not true(round(0)).
next(round(?n))            <= true(round(?m)), succ(?m,?n).
terminal                   <= true(round(9)).
```

Figure 1. A GDL-II description of the Two Generals' Coordinated Attack Game: G_{ca}.

Example 1. (Coordinated Attack Problem) A valley separates two hills. Two armies, each on its own hill and led by General A and B, respectively, are preparing to attack their common enemy in the valley. The two generals must have their armies attack the valley at the same time in order to succeed. The only way for the two generals to communicate is by sending messengers through the valley. Unfortunately, there is a chance that any given messenger sent through the valley will be stopped by the enemy, in which case the message is lost but the content is not leaked. The problem is to come up with algorithms that the generals can use, including sending messages and processing received messages, that can allow them to correctly agree upon a time to attack.

It has been proved that such a coordinated attack is impossible [3]. We use this example to show that complex epistemic situations can arise in GDL-II games; specifically, we will use the game semantics to show why a coordinated attack is not possible.

Figure 1 gives a GDL-II description of the Coordinated Attack Problem as a 3-player game, where generals A and B are modelled as two roles and the enemy is modelled by the standard 'random' role (line 1). For the sake of simplicity, general A starts by selecting from just two possible attack times: '3am' or '9pm' (line 3 and 15), and then sends his choice as a message to B (line 17 and 26). Subsequently, each general takes control in turn (lines 33–34), and if one receives a message m then he sends an acknowledgement $ack(m)$ back to the other general (line 20), otherwise he does *noop* (lines 18–19); simultaneously, the 'random' role always chooses randomly between either *pass*, which allows the message to go to the other general (line 13), or *stop*, which intercepts the message (line 14). For the sake of simplicity, we assume that the game terminates at round 9 (line 36) and leave out the specification of goal values.

The semantics of a game description G according to Definition 3 derives a state transition system from a set of rules. In the following, we use the operational semantics implicit in Definition 4 to define the special concept of *epistemic game models* for GDL-II.

Definition 8 (GDL-II Epistemic Game Model). Given an arbitrary GDL-II description G and its semantics $\langle R, s_0, t, l, u, \mathcal{I}, g \rangle$, an *epistemic game model* of G, denoted by $\mathsf{E}(G)$, is a structure $\langle W, Ag, \{\sim_i : i \in Ag\}, V \rangle$ where

- W is the set of developments of G;
- Ag is the set of roles $R \setminus \{random\}$;
- $\sim_i \subseteq W \times W$ is the accessibility relation for agent $i \in Ag$ given by $(\delta, \delta') \in \sim_i$ (also written as $\delta \sim_i \delta'$) iff role i cannot distinguish developments δ and δ';
- $V : W \to 2^{\Sigma}$ is an interpretation function which associates with each development δ the set of ground terms in Σ that are true in the last state of δ.

In the following, we restrict our attention to finite epistemic models.

As an example, from the game description G_{ca} for the Coordinated Attack Problem, we derive a game model $\mathsf{E}(G_{ca})$ in two steps (see Figure 2). The first step is to use the game semantics for GDL-II to determine all states that are reachable from the initial state. A joint move is

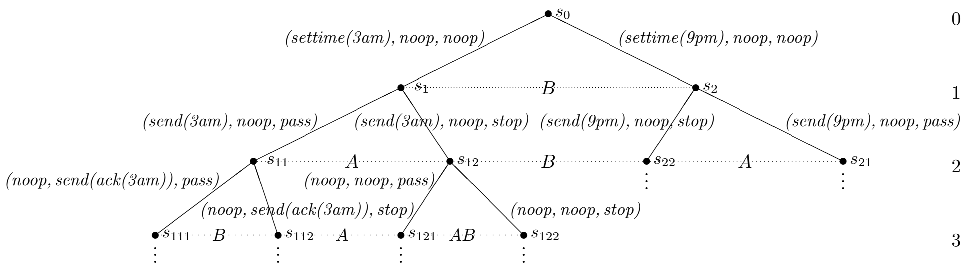

Figure 2. Epistemic game model $\mathsf{E}(G_{ca})$ for the Two Generals' Coordinated Attack Game.

depicted as (a, b, c), where a, b, c are the moves of, respectively, general A, general B, and 'random'. For instance, there are two possible joint moves at s_0, $M_1 = (settime(3am), noop, noop)$ and $M_2 = (settime(9pm), noop, noop)$, which transit s_0 to s_1 and s_2 respectively. From s_1 there are again two possible joint moves which result in s_{11} where B receives A's message, and s_{12} where B receives nothing. Accordingly at state s_{11}, it is legal for B to send an acknowledgement, and s_{11} transits to two possible states s_{111} and s_{112}. This process goes on utill a terminal state is reached.

The second step is to collect all the developments and then determine the individual accessibility relations. For example, consider the two developments $\delta_1 = \langle s_0, M_1, s_1 \rangle$ and $\delta_2 = \langle s_0, M_2, s_2 \rangle$. It is easy to check that $\delta_1 \not\sim_A \delta_2$ since General A moves differently in M_1 and M_2. On the other hand, $\delta_1 \sim_B \delta_2$ since B makes the same move in M_1 and M_2 and perceives nothing.

Based on our concept of an epistemic game model for GDL-II, we can define how to interpret formulas in the basic epistemic language over such models in a fashion similar to Definition 7.

Definition 9. Given an epistemic game model $\mathsf{E}(G)$, a development δ, and an epistemic formula ϕ, the entailment relation $\models$ is defined as follows:

- $\mathsf{E}(G), \delta \models P$ iff $P \in V(last(\delta))$;
- $\mathsf{E}(G), \delta \models \neg\phi$ iff $\mathsf{E}(G), \delta \not\models \phi$;
- $\mathsf{E}(G), \delta \models \phi \wedge \psi$ iff $\mathsf{E}(G), \delta \models \phi$ and $\mathsf{E}(G), \delta \models \psi$;
- $\mathsf{E}(G), \delta \models K_i\phi$ iff for all δ', if $\delta \sim_i \delta'$ then $\mathsf{E}(G), \delta' \models \phi$;
- $\mathsf{E}(G), \delta \models C_X\phi$ iff for all δ', if $\delta \sim_X \delta'$ then $\mathsf{E}(G), \delta' \models \phi$.

where $last(\delta)$ is the last state of development δ, and $\sim_X$ is the transitive and reflexive closure of $\cup_{i\in X} \sim_i$.

Coming back to our running example, a simple and elegant argument can be given now on why a coordinated attack is never possible. First, using the epistemic language (Definition 5) we can express knowledge conditions such as:

- $K_A P$ for "general A knows that P," where P is an atomic expression, e.g., $attack(3am)$, which means "attack is set to be at 3am;"
- $\neg K_B K_A P$ for "general B does not know whether general A knows P;"
- $C_{\{A,B\}} P$ for "P is common knowledge for both A and B."

Let $P = attack(3am)$ and $\delta_1, \delta_{11}, \delta_{111}$ be the left-most developments with length 1, 2, and 3 in Figure 2, then we can verify each of the following: $\mathsf{E}(G_{ca}), \delta_1 \models K_A P \wedge \neg K_B K_A P$; $\mathsf{E}(G_{ca}), \delta_{11} \models K_B K_A P \wedge \neg K_A K_B K_A P$; and $\mathsf{E}(G_{ca}), \delta_{111} \models K_A K_B K_A P \wedge \neg K_B K_A K_B K_A P$. It implies that $\mathsf{E}(G_{ca}), \delta \models \neg C_{\{A,B\}} P$ for $\delta = \delta_1, \delta_{11}, \delta_{111}$, that is, the attack time is not common knowledge among A and B even after the successful delivery of all messages during three rounds. We can generalise this to developments of arbitrary length. Such common knowledge is a precondition of coordinated attack, which is why it is never possible to achieve the latter.

In general, it is easy to show that the epistemic game model we constructed for GDL-II is equivalent to the standard concept of models and entailment in Modal Epistemic Logic: Specifically, we can pick up an arbitrary valid round[10] and build a finite epistemic model for this round such that the truth of epistemic formulas is preserved.

Theorem 1. *Given an arbitrary GDL-II description G and any valid round of playing $k \geq 0$ (with round 0 corresponding to the initial state), we can derive a finite epistemic model $\mathsf{E}^k(G)$ which characterises this round of the game, formally:*

$$\mathsf{E}^k(G), \delta \models \phi \textit{ iff } \mathsf{E}(G), \delta \models \phi.$$

Proof. Let $\mathsf{E}(G) = \langle W, Ag, \{\sim_i : i \in Ag\}, V\rangle$ be constructed from G according to Definition 8, and assume that the game playing is at round k.

Based on $\mathsf{E}(G)$, we construct a finite epistemic model $\mathsf{E}^k(G) = \langle W', \{\sim_i : i \in Ag'\}, V'\rangle$ for round k as follows:

1. W' is the set of any game development $\delta \in W$ with $len(\delta) = k + 1$;
2. Ag' is the same set of agents as Ag;
3. $\sim_i'$ is the equivalence relation $\sim_i$ restricted on the new domain W', i.e., $\sim_i' = \sim_i \cap (W' \times W')$;

[10]A round k is valid in a game if at least one development has length $k + 1$.

4. V is a valuation function such that $P \in V'(\delta)$ iff $P \in V(\delta)$ for any atomic proposition P and $\delta \in W'$.

We show by induction on the structure of formula ϕ that for all $\delta \in W'$: $\mathsf{E}^k(G), \delta \models \phi$ iff $\mathsf{E}(G), \delta \models \phi$. The propositional cases follow from the fact that the valuation does not change. For the case of $\phi := K_i\psi$, by definition, we have that $\mathsf{E}^k(G), \delta \models K_i\psi$ iff for all δ', if $\delta \sim_i' \delta'$ then $\mathsf{E}^k(G), \delta' \models \psi$. If two developments δ, δ' have different lengths, then any agent can distinguish them, so if $\delta \sim_i' \delta'$, then $len(\delta) = len(\delta') = k+1$, which means that $\delta' \in W'$ as well. So by induction, for all δ', if $\delta \sim_i' \delta'$ then $\mathsf{E}^k(G), \delta' \models \phi$ iff for all δ', if $\delta \sim_i' \delta'$ then $\mathsf{E}(G), \delta' \models \phi$; therefore $\mathsf{E}^k(G), \delta \models K_i\psi$ iff $E(G), \delta \models K_i\psi$. For the case of $\phi := C_X\psi$, the reasoning is similar as the developments in the transitive and reflexive closure of $\cup_{i \in X} \sim_i$ are also of the same length $k+1$. ■

Taking the running example G_{ca} (Figure 2), we can see that $\mathsf{E}^1(G_{ca})$ consists of two developments of length 2: $\delta_1 = \langle s_0, M_1, s_1 \rangle$ and $\delta_2 = \langle s_0, M_2, s_2 \rangle$. The equivalence relations are given naturally by $\sim_A = \{(\delta_1, \delta_1), (\delta_2, \delta_2)\}$ and $\sim_B = \{(\delta_1, \delta_1), (\delta_2, \delta_2), (\delta_1, \delta_2), (\delta_2, \delta_1)\}$, and so is the valuation.

As a corollary, we can show, for instance, that the round number is common knowledge for all the agents in all rounds. In our running example, this if formally obtained as

$$\mathsf{E}(G_{ca}), \delta \models \bigwedge_k (round(k) \to C_{\{A,B\}} round(k)).$$

4. From Epistemic Models to GDL-II

We now look at the other direction and show that for *any* given finite multi-agent epistemic model E we can construct a valid GDL-II game description such that E arises when playing the game. This formally shows that GDL-II is sufficiently expressive to allow for modelling arbitrarily complex epistemic situations as games. As a matter of fact, a (very abstract) game can always be constructed where a single move suffices to bring about an arbitrary given epistemic model.

THEOREM 2. *For an arbitrary finite multi-agent epistemic model* $E = \langle W, \{\sim_i : i \in Ag\}, V \rangle$*, a GDL-II game description* G *can be constructed such that* E *arises from playing* G*, namely* E *is isomorphic to* $\mathsf{E}^1(G)$ *which characterises the situation after the first move.*

```
1 role(1).  ...  role(n).  role(random).
2 world(w1).  ...  world(wk).
3
4 init(round(0)).
5
6 legal(random,select(?w)) <= true(round(0)), world(?w).
7 legal(?r,noop)           <= true(round(0)),
8                                role(?r), distinct(?r,random).
9
10 val(w1,P1).  ...  val(wk,Pm).
11 next(?p) <= does(random,select(?w)), val(?w,?p).
12
13 equiv(1,wa,wa).  equiv(1,wa,wb).  ...  equiv(n,wx,wy).
14 sees(?r,class(?w2)) <= does(random,select(?w1)), equiv(?r,?w1,?w2).
```

Figure 3. A GDL-II description for any epistemic model E

Proof. Let $W = \{w_1, \ldots, w_k\}$ and $Ag = \{1, \ldots, n\}$. G can be constructed as shown in Figure 3: The game has $n+1$ roles, namely, the n agents plus the standard 'random' role (line 1). Initially, 'random' has a legal move $select(w)$ for any world $w \in W$ (lines 2–6) while all other players can only do *noop* (lines 7–8). The move $select(w)$ results in a state in which all atomic propositions hold that are true in world w (line 11). The rule in line 10 uses an explicit enumeration of all pairs (w, P) such that $P \in V(w)$. Furthermore, in order to arrive at the desired epistemic structure, the players get to see *all* worlds in their equivalence class $\{w' : (w, w') \in \sim_i\}$ (line 14). The rule in line 13 uses an explicit enumeration of all triples (i, w_a, w_b) such that $w_a \sim_i w_b$. We omit definitions for `terminal` and `goal` since they are not relevant here.

It is easy to see that the set of rules in Figure 3 satisfy all syntactic requirements of valid GDL specifications according to Definition 1. We show that G indeed gives E according to the semantics in Definition 3. The initial state is $s_0 = \{round(0)\}$, and $G \cup s_0^{\texttt{true}}$ entails $\texttt{legal}(\texttt{random}, select(w_j))$ for all $j \in [1..k]$, and $\texttt{legal}(1, noop), \ldots, \texttt{legal}(n, noop)$. Accordingly, each agent in Ag can only do *noop*, while 'random' may select an arbitrary world from E. Define joint move $M^j \stackrel{\text{def}}{=} (noop, ..., noop, select(w_j))$ and consider new states $s_x = u(M^x, s_0)$, and $s_y = u(M^y, s_0)$, corresponding to the developments $\delta_x = \langle s_0, M^x, s_x \rangle$ and $\delta_y = \langle s_0, M^y, s_y \rangle$ respectively. If $w_x \sim_i w_y$, then agent i gets to see both $class(w_x)$ and $class(w_y)$ in both states s_x and s_y, in which case the agent cannot distinguish δ_x from δ_y (note also his actions are the same in both M_1 and M_2). On the other hand, if $w_x \not\sim_i w_y$ then agent i can distinguish the two developments based on his percepts. Altogether this process gives us an epistemic game model $\mathsf{E}(G)$.

Then we define a standard epistemic model $\mathsf{E}^1(G) = \langle W', \{\sim'_i : i \in Ag\}, V' \rangle$ from $\mathsf{E}(G)$ as follows: W' is the set of all developments of length 2 from $\mathsf{E}(G)$; $\sim'_i$ and V' are restrictions of $\sim_i$ and V on W' respectively.

Now E and $\mathsf{E}^1(G)$ are isomorphic: Each world $w_j \in W$ corresponds to the state s_j and hence to the development $\delta_j \in W'$. $(w_x, w_y) \in \sim_i$ iff $(\delta_x, \delta_y) \in \sim'_i$ for agent i, and for all atomic proposition P we have that $P \in V(w_j)$ iff $P \in V'(\delta_j)$. ∎

This theorem shows that GDL-II is sufficiently expressive to model any situation that can be described by a (finite) epistemic model. Combining Theorem 1 and 2, we conclude that GDL-II and epistemic models are equally expressive in terms of the *description* of incomplete information situations.

It is worth pointing out, however, that this does not mean that GDL-II has the equal expressivity as the epistemic logic in general, because the epistemic language can express more subtle properties. For example, a rule that explicitly refers to agents' knowledge like "if A and B achieve common knowledge about P, then the game terminates" can be expressed as a formula "$C_{\{A,B\}}P \to \texttt{terminal}$" in epistemic logic, but such a rule cannot be specified in GDL-II. Nevertheless, as a game specification language, GDL-II is sufficiently expressive to describe objective game rules in a very succinct way, while an explicit representation of games states in epistemic models may require exponential space.

When it comes to *reasoning* about GDL-II game rules and what they entail about the knowledge of players, additional language elements are necessary, such as provided by epistemic logic, which can then be interpreted in the epistemic model that is derived from a GDL-II description as shown in Section 3.

5. Conclusion and Further Work

In this paper, we analysed the epistemic structure and expressiveness of GDL-II in terms of modal epistemic logic and presented two results: (1) The operational semantics of GDL-II entails that the situation in any round of a game can be characterised by a multi-agent epistemic model, (2) GDL-II is sufficiently expressive to model any situation that can be described by a finite multi-agent epistemic model.

In terms of related work, in [22] we have related GDL-II to the general mathematical concept of extensive-form games in order to show that any such game can be described faithfully in GDL-II. Other related work describes the use of Alternating-time Temporal Logic to represent and verify properties of general games [20], but this is restricted to original GDL and hence to games with perfect information. There is of course a large body of work on epistemic logic of incomplete-information situations (such as [24] for formalising agents' perception and knowledge), but ours is the first application of this line of research to formally analyse the epistemic structure behind the general Game Description Language.

We outline some issues for further work. Apart from theoretical results, we are interested in investigating a more practical side of the problem. Our results in this paper provide the foundation for automated reasoning about epistemic properties of games. The next step will be to use model checking methods for the purpose of game verification. For example, given that agents may have only partial observation ability, it is easy to construct games in which agents do not have sufficient information to derive their legal moves; this may render a game unfair or even not playable. We can express the property that "all agents know their legal moves" in the basic epistemic language as

$$\bigwedge_{i \in Ag} \bigwedge_{m \in \Sigma} (legal(i,m) \rightarrow K_i legal(i,m)).$$

To check such properties systematically amounts to the following model checking problem: given a GDL-II description G, a round number k, and an epistemic formula ϕ, verify that $\mathsf{E}(G), \delta_{\leq k} \models \phi$ for all δ of length $\leq k$. Our preliminary study [9] shows how in principle it is possible to apply model checking in our setting.

We are also interested in reasoning about how players' knowledge evolves as the game progresses, and the strategic ability of players to reach a desirable 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. Our recent paper [19] has made a start.

Finally, recall that GDL-II assumes a finite set of players, finite states, and finitely many legal moves in each state. Lifting such assumptions can easily lead to infinite game models. It is an interesting line of future research to extend the results in this paper to such infinite models.

Acknowledgements. A preliminary version of this paper has been published in the Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI'11). We thank Jérôme Lang for suggesting the example for the paper, the anonymous reviewers from AAAI'11 and Studia Logica for their helpful comments. This research was supported under

Australian Research Council's (ARC) *Discovery Projects* funding scheme (project DP 120102023). The second author is the recipient of an Australian Research Council Future Fellowship (project FT 0991348) and is also affiliated with the University of Western Sydney.

References

[1] Apt, Krzysztof, Howard A. Blair, and Adrian Walker, Towards a theory of declarative knowledge, in J. Minker, (ed.), *Foundations of Deductive Databases and Logic Programming*, chap. 2, Morgan Kaufmann, 1987, pp. 89–148.

[2] Aumann, Robert, and Adam Brandenburger, Epistemic conditions for Nash equilibrium, *Econometrica* 63:1161–1180, 1995.

[3] Fagin, Ronald, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi, *Reasoning About Knowledge*, The MIT Press: Cambridge, MA, 1995.

[4] Gelder, Allen Van, The alternating fixpoint of logic programs with negation, in *Proceedings of the 8th Symposium on Principles of Database Systems*, ACM SIGACT-SIGMOD, 1989, pp. 1–10.

[5] Gelfond, Michael, and Vladimir Lifschitz, The stable model semantics for logic programming, in R. Kowalski, and K. Bowen, (eds.), *Proceedings of the International Joint Conference and Symposium on Logic Programming (IJCSLP)*, MIT Press, Seattle, OR, 1988, pp. 1070–1080.

[6] Genesereth, Michael, Nathaniel Love, and Barney Pell, General game playing: Overview of the AAAI competition, *AI Magazine* 26(2):62–72, 2005.

[7] Halpern, Joseph Y., and Moshe Y. Vardi, The complexity of reasoning about knowledge and time, in *Proceedings 18th ACM Symposium on Theory of Computing*, 1986, pp. 304–315.

[8] Hintikka, Jaakko, *Knowledge and Belief*, Cornell University Press, Ithaca, NY, 1962.

[9] Huang, Xiaowei, Ji Ruan, and Michael Thielscher, Model checking for reasoning about incomplete information games, in *Proceedings of the Australasian Conference on Artificial Intelligence*, Springer LNAI 8272, Dunedin, New Zealand, 2013, pp. 246–258.

[10] Kripke, Saul, Semantical analysis of modal logic, *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 9:67–96, 1963.

[11] Lloyd, John, *Foundations of Logic Programming*, second, extended edn., Series Symbolic Computation, Springer, 1987.

[12] Lloyd, John W., and Rodney W. Topor, A basis for deductive database systems II, *J. Log. Program.* 3(1):55–67, 1986.

[13] Love, Nathaniel, Timothy Hinrichs, David Haley, Eric Schkufza, and Michael Genesereth, General Game Playing: Game Description Language Specification, Tech. Rep. LG–2006–01, Stanford Logic Group, Computer Science Department, Stanford University, 2006.

[14] Moore, Robert C., A formal theory of knowledge and action, in J.F. Allen, J. Hendler, and A. Tate, (eds.), *Readings in Planning*, Morgan Kaufmann Publishers, San Mateo, CA, 1990, pp. 480–519.

[15] Pell, Barney, *Strategy Generation and Evaluation for Meta-Game Playing*, Ph.D. thesis, Computer Laboratory, University of Cambridge, 1993.

[16] Pitrat, Jacques, A general game playing program, in N. Findler, and B. Meltzer, (eds.), *Artificial Intelligence and Heuristic Programming*, Edinburgh University Press, 1971, pp. 125–155.

[17] Pritchard, David, *The Encyclopedia of Chess Variants*, Godalming, 1994.

[18] Rao, Anand S., and Michael P. Georgeff, Modeling rational agents within a BDI-architecture, in James F. Allen, Richard Fikes, and Erik Sandewall, (eds.), *KR*, Morgan Kaufmann, 1991, pp. 473–484.

[19] Ruan, Ji, and Michael Thielscher, Strategic and epistemic reasoning for the game description language GDL-II, in *Proceedings of the European Conference on Artificial Intelligence (ECAI 2012)*, IOS Press, Montpellier, France, 2012, pp. 696–701.

[20] Ruan, Ji, Wiebe van der Hoek, and Michael Wooldridge, Verification of games in the Game Description Language, *Journal Logic and Computation* 19(6):1127–1156, 2009.

[21] Thielscher, Michael, A general game description language for incomplete information games, in *Proceedings of the Conference on the Advancement of Artificial Intelligence (AAAI)*, Atlanta, 2010, pp. 994–999.

[22] Thielscher, Michael, The general game playing description language is universal, in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Barcelona, 2011, pp. 1107–1112.

[23] Wooldridge, Michael, *An Introduction to Multiagent Systems*, John Wiley & Sons, 2002.

[24] Wooldridge, Michael, and Alessio Lomuscio, A computationally grounded logic of visibility, perception, and knowledge, *Logic Journal of the IGPL* 9(2):257–272, 2001.

[25] Wright, Georg H. von, *An Essay in Modal Logic*, North-Holland, Amsterdam, 1951.

Ji Ruan
School of Computer and Mathematical Sciences
Auckland University of Technology
Auckland 1142, New Zealand
`ji@jiruan.net`

Michael Thielscher
School of Computer Science and Engineering
The University of New South Wales
Sydney, NSW 2052, Australia
`mit@cse.unsw.edu.au`