GDL Meets ATL: A Logic for Game Description and Strategic Reasoning

Guifei Jiang[1,2], Dongmo Zhang[1], and Laurent Perrussel[2]

[1] AIRG, University of Western Sydney, Australia
[2] IRIT, University of Toulouse 1, France

Abstract. This paper presents a logical framework that extends the Game Description Language with coalition operators from Alternating-time Temporal Logic and prioritised strategy connectives. Our semantics is built upon the standard state transition model. The new framework allows us to formalise van Benthem's game-oriented principles in multi-player games, and formally derive Weak Determinacy and Zermelo's Theorem for two-player games. We demonstrate with a real-world game how to use our language to specify a game and design a strategy, and how to use our framework to verify a winning/no-losing strategy. Finally, we show that the model-checking problem of our logic is in 2EXPTIME with respect to the size of game structure and the length of formula, which is no worse than the model-checking problem in ATL*.

1 Introduction

Logical analysis of games has been an important topic across the study of game theory, mathematics, philosophy and computer science [1–7]. It deals with the problems of (1). *how to specify a game situation*, (2). *how to represent a game strategy* and, more importantly, (3). *how to model strategic reasoning of game players*. A number of logical formalisms have been proposed to tackle these problems from different perspectives.

To deal with the first problem, Parikh and Pauly proposed a logical framework in which a game is treated as a program so that different games can be combined by program connectives [5, 8]. Kaneko proposed another approach to game description by specifying each single game as a propositional logic theory [4]. A more practical approach to specify a game by the so-called *Game Description Language* (GDL) [9]. The language is much less expressive than the above mentioned game logics, but rich enough for describing any finite combinatorial games. This language has been used as an official language for General Game Playing since 2005.

To address the second problem, a number of approaches have been proposed for representing game strategies [2, 7, 10–12]. The simplest way to represent a game strategy is to express a strategy as an action so that simple strategies can be combined into more complicated strategies using PDL connectives [7, 11]. Zhang and Thielscher recently introduced another approach to represent game strategies [12]. With their framework, a strategy is represented as a logical formula in GDL. More importantly, strategies can be combined by using a pair of prioritised logical connectives.

Some recent work has been done to tackle the third problem, mostly based on either Pauly's Coalition Logic (CL) or Alur *et al*'s Alternating-time Temporal Logic (ATL) [1, 8]. Both logics use coalition modalities to specify strategic abilities of coalitions. However, these logics use existential quantifiers to express players' strategic abilities, such as '*a coalition of players has a strategy to achieve a game property*', while description of strategies is not part of the logical language [13]. There has been some work that extends ATL with explicit expression of game strategies by adding a strategy term into coalitional operators to mean a coalition commits to a strategy [3, 14, 15]. However, strategies in their languages have to be in atomic forms with no internal structures.

None of the above mentioned work can deal with all three problems within a single logical framework. This is actually a serious problem because without explicitly specifying game rules, without expressing the strategies under consideration, we are unable to reason about the effects of these strategies. We need a comprehensive logical language and associated inference mechanism to define, compare and reason about strategies. In this paper, we propose a logical language by extending GDL with coalition operators and prioritised connectives [1, 9, 12]. Inherited from GDL, the proposed logical language can describe any finite perfect information game. Furthermore, by using Zhang and Thielscher's prioritised connectives and ATL-like coalition operators, the language can represent complicated game strategies and specify strategic abilities of players. More importantly, we provide unified semantics for both GDL- and ATL- formulas, which allows us to formalize the game-playing principles introduced by van Benthem [7]. These principles make it possible to formally derive two well-known results for two-player games: *Weak Determinacy* and *Zermelo's Theorem*. Meanwhile, we use a generalised Gomuko game to demonstrate how to use our logical formalism to describe a game strategy and reason about strategies. Finally, we present an upper bound of the model-checking complexity of our logic.

The rest of this paper is structured as follows. Section 2 establishes the syntax and semantics of our proposed logical framework. Section 3 deals with the syntactical representation of strategies and demonstrates how to design strategies and reason about strategies in our framework. Section 4 discusses the model-checking problem of the logic. Finally we conclude the paper with a discussion of future work.

2 The Logical Framework

In this section, we will introduce a logical language for specifying game rules and representing game strategies, and provide semantics of the language based on the state transition model. We call the framework *the logic for Game Description and strategic Reasoning*, denoted by GDR for short.

2.1 The Syntax

The language, denoted by $\mathcal{L}$, consists of the following components. Let N be a finite set of agents and Φ a countable set of propositional atoms. For each $a \in N$, A^a is a finite set of actions that agent a can perform. We assume that $A^a \cap A^b = \emptyset$ if $a \neq b$ and let $\mathcal{A} = \bigcup_{a \in N} A^a$. A formula φ in $\mathcal{L}$ is defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathit{initial} \mid \mathit{terminal} \mid \mathit{does}(\alpha) \mid \\ \mathit{wins}(a) \mid \mathit{turn}(a) \mid \bigcirc\varphi \mid \varphi \triangledown \varphi \mid [C]\varphi \mid [\![C]\!]\varphi$$

where $p \in \Phi$, $a \in N$, $\alpha \in \mathcal{A}$ and $C \subseteq N$.

Besides the standard propositional connectives[1], *initial* and *terminal* are two logical constants, specifying the initial state and the terminal states of a game. $\mathit{does}(.)$, $\mathit{wins}(.)$ and $\mathit{turn}(.)$ are specific propositional variables, called pseudo-function symbols, introduced in GDL for specifying a game rule. $\mathit{does}(\alpha)$ means that the action α is taken in this state. $\mathit{wins}(a)$ means it is in a winning state of player a. $\mathit{turn}(a)$ says it is player a's turn in the current state. The temporal formula $\bigcirc\varphi$ is read as φ holds in the next stage of the game. All these language components are inherited from GDL.

The prioritised disjunctive connective $\triangledown$ is borrowed from [12]. The formula $\varphi_1 \triangledown \varphi_2$ has to be read as "*achieve φ_1 first; if it is impossible, achieve φ_2*". Different from [12], we define the prioritised conjunction $\vartriangle$ by the prioritised disjunction:

$$\varphi_1 \vartriangle \varphi_2 =_{def} (\varphi_1 \wedge \varphi_2) \triangledown \varphi_1$$

It means "*achieve both φ_1 and φ_2; if they are conflict, only achieve φ_1*".

The two coalition operators $[C]$ and $[\![C]\!]$ are taken from ATL. $[C]\varphi$ says that coalition C has a joint strategy to achieve φ at the next state. $[\![C]\!]\varphi$ says that coalition C has a joint strategy for maintaining φ forever. From the semantics we will see that these operators are counterparts of $\langle\!\langle C \rangle\!\rangle\bigcirc$ and $\langle\!\langle C \rangle\!\rangle\Box$ in ATL.

To help the reader catch the intuition of the language, let us consider a special family of *mnk*-games [16].

Example 1. **(*mk*-Game)** An *mk*-game is a combinatorial game in which two players take turns in marking either a nought 'o' or a cross 'x' on an $m \times m$ board. The player who first gets k consecutive marks of her own symbol in a row (horizontally, vertically, or diagonally), will win the game. Obviously, an *mk*-game is a generalisation of Tic-Tac-Toe ($m = k = 3$) and Gomoku ($m = 19$ and $k = 5$).

Now we describe the *mk*-games in our language. Given a player $a \in \{\mathsf{x}, \mathsf{o}\}$, let $p^a_{i,j}$ denote that grid (i, j) is filled with player a's symbol, and $\alpha^a_{i,j}$ denote the action that player a fills grid (i, j) with her symbol, where $1 \leq i, j \leq m$. The game rules can then be specified by the following formulas:

[1] The other logical connectives $\vee$, $\rightarrow$, $\leftrightarrow$ and constants $\top$, $\bot$ can be defined in the standard way.

(1) $\mathit{initial} \to \mathit{turn}(\mathsf{x}) \wedge \neg \mathit{turn}(\mathsf{o}) \wedge \bigwedge_{i,j=1}^{m} \neg(p_{i,j}^{\mathsf{x}} \vee p_{i,j}^{\mathsf{o}})$

(2) $\mathit{wins}(a) \leftrightarrow (\bigvee_{i=1}^{m} \bigvee_{j=1}^{m-k+1} \bigwedge_{l=0}^{k-1} p_{i,j+l}^{a}) \vee (\bigvee_{i=1}^{m-k+1} \bigvee_{j=1}^{m} \bigwedge_{l=0}^{k-1} p_{i+l,j}^{a})$
$\vee (\bigvee_{i=1}^{m-k+1} \bigvee_{j=1}^{m-k+1} \bigwedge_{l=0}^{k-1} p_{i+l,j+l}^{a}) \vee (\bigvee_{i=1}^{m-k+1} \bigvee_{j=k}^{m} \bigwedge_{l=0}^{k-1} p_{i+l,j-l}^{a})$

(3) $\mathit{teminal} \leftrightarrow \mathit{wins}(\mathsf{x}) \vee \mathit{wins}(\mathsf{o}) \vee \bigwedge_{i,j=1}^{m} (p_{i,j}^{\mathsf{x}} \vee p_{i,j}^{\mathsf{o}})$

(4) $\mathit{does}(\alpha_{i,j}^{a}) \to \neg(p_{i,j}^{\mathsf{x}} \vee p_{i,j}^{\mathsf{o}}) \wedge \mathit{turn}(a) \wedge \neg \mathit{terminal}$

(5) $\bigcirc p_{i,j}^{a} \leftrightarrow p_{i,j}^{a} \vee \mathit{does}(\alpha_{i,j}^{a})$

(6) $\mathit{turn}(a) \to \bigcirc \neg \mathit{turn}(a) \wedge \bigcirc \mathit{turn}(-a)$, where $-a$ represents a's opponent.

Statement (1) says that all grids are empty in the initial state and player x has the first turn. (2) and (3) specify the winning states for each player and terminal states for the game, respectively. (4) specifies the precondition of each action (legality). (5) is the combination of the frame axioms and effect axioms: *a grid is marked with a player's symbol in the next state if the player takes the respective action in the current state or the grid has been filled before.* The last formula specifies turn-taking.

2.2 State Transition Model

In order to provide the semantics for the logic, we use the state transition model to specify a game in the semantic level [12].

Definition 1 (State Transition Model). *A state transition model M is a tuple $(N, W, \mathcal{A}, \overline{w}, T, U, g, t, V)$, where*

- *N is a non-empty finite set of players;*
- *W is a non-empty set of states;*
- *$\mathcal{A} = \bigcup_{a \in N} A^a$ where A^a is a non-empty finite set of actions for agent $a \in N$.*
- *$\overline{w} \in W$ is the initial state;*
- *$T \subseteq W$ is the set of terminal states;*
- *$U : \mathcal{A} \times W \hookrightarrow W$ is a partial function specifying the state transitions.*
- *$g : N \mapsto 2^W$ is a goal function, specifying the winning states for each player;*
- *$t : W \backslash T \mapsto N$ is a label function, specifying players' turns.*
- *$V : W \mapsto 2^{\Phi}$ is a standard valuation function.*

To keep our formalism simple, we assume that all actions are performed asynchronously, and different players have different actions, i.e., $A^a \cap A^b = \emptyset$ for any $a \neq b \in N$.

Definition 2 (Complete Path). *A sequence $w_0 \overset{\alpha_0}{\to} w_1 \overset{\alpha_1}{\to} \cdots \overset{\alpha_{m-1}}{\to} w_m$ is a complete path if*

1. *$w_0 = \overline{w}, w_m \in T$, and $w_i \in W$ for all $0 < i < m$;*
2. *$\alpha_i \in \mathcal{A}$ for all $0 \leq i < m$;*
3. *$U(w_i, \alpha_i) = w_{i+1}$ for all $0 \leq i < m$.*

Given a complete path $\delta = w_0 \overset{\alpha_0}{\rightarrow} w_1 \overset{\alpha_1}{\rightarrow} \cdots \overset{\alpha_{m-1}}{\rightarrow} w_m$, we call the s $w_0, \cdots, w_m$ on δ *reachable states.* w_j is the state at *stage* j. Let $\delta[i]$ b i-th reachable state on path δ, and $\delta[i+1]$ be the direct successor of δ δ. The set of all reachable states of δ is denoted by W^{δ}. For each action also let $W^a = \{w \in W : t(w) = a\}$ to denote the set of all states in it is a's turn. Further more, we denote the set of all complete paths in a transition model M as $\mathcal{P}(M)$. Thus the set of all reachable states in M, de by $W^{\mathcal{P}(M)}$, is $\bigcup_{\delta \in \mathcal{P}(M)} W^{\delta}$.

A strategy for a player a is a plan of player a that determines what a she is to take at each reachable state when it is her turn. Formally, *a st* for player a is a total function $f_a : W^a \cap W^{\mathcal{P}(M)} \mapsto \mathcal{A}$ such that $U(w, f$ is not undefined for all $w \in W^a \cap W^{\mathcal{P}(M)}$. A joint strategy for a coalitio combination of its members' strategies. Formally, *a joint strategy* for a coa $C \subseteq N$ is a total function $f_C : \bigcup_{a \in C}(W^a \cap W^{\mathcal{P}(M)}) \mapsto \mathcal{A}$ such that $f_C($ $f_a(w)$ if $w \in W^a$. The set of all joint strategies of coalition C is denoted b

Given a complete path δ and a non-terminal stage i $(0 \leq i < m)$ on $\sigma(\delta, i)$ denote the action taken at stage i on path δ. We say a complete p *complies with* player a's strategy f_a if for all $w \in W^a \cap W^{\delta}$, for all $i \in$ $\delta[i] = w$, then $\sigma(\delta, i) = f_a(w)$. That is, for any reachable state w on δ it is a's turn, the action taken at w on δ is the same as what the strate specifies. Similarly, a complete path δ *complies with* a joint strategy f_C if f $w \in W^a \cap W^{\delta}$, for all $i \in \mathbb{N}$, if $\delta[i] = w$, then $\sigma(\delta, i) = f_C(w)$.

In fact, during game playing, a player often begins to use a strategy reaching some game state. This means a complete path δ may start to cc with a joint strategy f_C after reaching some state. Let $\mathcal{P}(f_C, w)$ denote tl of complete paths δ reaching the state w at some stage i where the agents start to use the strategies f_C. Formally,

$$\mathcal{P}(f_C, w) = \{\delta \in \mathcal{P}(M) \mid \exists i\ \delta[i] = w \text{ and } \forall j \geq i\ \sigma(\delta, j) = f_C(\delta[j])\}$$

2.3 The Semantics

Before presenting the semantics for GDR, we need another notation. Let denote the *initial segment* of path δ up to stage i. Any complete path λ shares the same initial segment with δ up to stage i is denoted by $\delta[0, i]$ Given $\varphi \in \mathcal{L}$, let

$$\mathcal{P}(\varphi, \delta[0, i]) = \{\delta' \in \mathcal{P}(M) \mid \delta[0, i] \sqsubseteq \delta' \text{ and } M, \delta', i \models \varphi\}$$

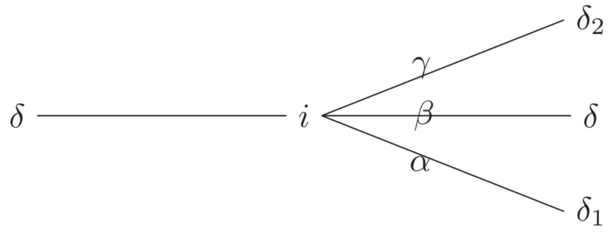
denote the set of all complete paths that share the same initial segment of δ up to stage i and satisfy φ at stage i. We are now in the position to giv truth conditions for GDR formulas.

Definition 3 (Truth Conditions). *Let $M = (N, W, \mathcal{A}, \overline{w}, T, L, U, g, t, V)$ be a state transition model. Given a complete path δ and a stage i on δ, we define the notion that $\varphi \in \mathcal{L}$ is true at i on δ, denoted by $M, \delta, i \models \varphi$, as follows:*

$$
\begin{array}{lll}
M,\delta,i \models p & \textit{iff} & p \in V(\delta[i]) \\
M,\delta,i \models \neg\varphi & \textit{iff} & M,\delta,i \not\models \varphi \\
M,\delta,i \models \varphi_1 \wedge \varphi_2 & \textit{iff} & M,\delta,i \models \varphi_1 \textit{ and } M,\delta,i \models \varphi_2 \\
M,\delta,i \models does(\alpha) & \textit{iff} & \alpha = \sigma(\delta,i) \\
M,\delta,i \models initial & \textit{iff} & \delta[i] = \overline{w} \\
M,\delta,i \models terminal & \textit{iff} & \delta[i] \in T \\
M,\delta,i \models wins(a) & \textit{iff} & \delta[i] \in g(a) \\
M,\delta,i \models turn(a) & \textit{iff} & t(\delta[i]) = a \\
M,\delta,i \models \bigcirc\varphi & \textit{iff} & M,\delta,i+1 \models \varphi \\
M,\delta,i \models \varphi_1 \triangledown \varphi_2 & \textit{iff} & M,\delta,i \models \varphi_1,\ \textit{or}\ (\mathcal{P}(\varphi_1,\delta[0,i]) = \emptyset \textit{ and } M,\delta,i \models \varphi_2) \\
M,\delta,i \models [C]\varphi & \textit{iff} & \exists f_C \in F_C\ \forall \delta' \in \mathcal{P}(f_C,\delta[i])\ \forall j \in \mathbb{N}\ \textit{if } \delta[i] = \delta'[j], \\
 & & \textit{then } M,\delta',j+1 \models \varphi. \\
M,\delta,i \models [\![C]\!]\varphi & \textit{iff} & \exists f_C \in F_C\ \forall \delta' \in \mathcal{P}(f_C,\delta[i])\ \forall j \in \mathbb{N}\ \textit{if } \delta[i] = \delta'[j], \\
 & & \textit{then } \forall k \geq j,\ M,\delta',k \models \varphi.
\end{array}
$$

The interpretation for GDL components is straightforward. The last two statements define the semantical conditions for the coalition operators, which is similar to the ones in ATL. $[C]\varphi$ (or $[\![C]\!]\varphi$) is true if coalition C has a joint strategy to make φ true in the next stage (or maintain φ from now on) for all possible complete paths in the set $\mathcal{P}(f_C, \delta[i])$ Note that index j denotes the stage when complete path δ' reaches the state $\delta[i]$. It is possible that $i \neq j$, since two paths may reach the same state at different stages. To show the interpretation of the prioritised disjunction $\triangledown$, let's consider the following example.

Assume that a path δ in a state transition model M diverges at stage i due to executing different actions as illustrated below:



Consider a formula $does(\alpha)\triangledown does(\beta)$, which says "do action α; if α is not doable, do β". We check whether it is true at stage i on path δ. Since δ takes action β instead of α at i, we have $M,\delta,i \models \neg does(\alpha) \wedge does(\beta)$. However, this does not mean α is not doable at i. In fact, it can be done through δ_1 ($M,\delta_1,i \models does(\alpha)$) and δ_1 shares the same initial segment of δ up to stage i (i.e., $\delta_1 \in \mathcal{P}(\varphi_1, \delta[0,i])$). Thus α is doable at i. According to the semantics, $M,\delta,i \not\models does(\alpha)\triangledown does(\beta)$. This is because δ picks up β even though α is doable at state i.

It is worth noting that when $i = m$, we have $M,\delta,m \not\models does(\alpha)$ for all $\alpha \in \mathcal{A}$, $M,\delta,m \models \bigcirc\varphi$ and $M,\delta,m \models [C]\varphi$ for all $\varphi \in \mathcal{L}$. Given a state transition model M and a formula $\varphi \in \mathcal{L}$, we say φ is *globally true* through a complete path δ, denoted by $M,\delta \models \varphi$, if $M,\delta,i \models \varphi$ for all i on δ. φ is *valid* in model M, written as $M \models \varphi$, if $M,\delta \models \varphi$ for all complete paths $\delta \in \mathcal{P}(M)$. Furthermore, $\models \varphi$ means that φ is valid in all state transition models.

3 Strategic Reasoning

We now show how to use our logical framework to reason about game strategies. We first formalise Johan Benthem's description of game-oriented principles and use them to derive two well-known results in combinatorial game theory.

3.1 Game-Oriented Principles

Johan Benthem described a number of game-oriented principles in his temporal forcing logic [7]. These principles specify the fundamental properties of any finite games, and they can be formally presented and proved in our framework.

Theorem 1. *For any state transition model M, $a \in N$, $C, D \subseteq N$ and $\varphi, \phi, \psi \in \mathcal{L}$,*

(A1) *If φ does not contain $\bigcirc$ and $does(.)$, then*

$$M \models [\![a]\!]\varphi \leftrightarrow \varphi \wedge (terminal \vee (turn(a) \wedge [N][\![a]\!]\varphi) \vee \bigvee_{b \in N \setminus \{a\}} (turn(b) \wedge [\emptyset][\![a]\!]\varphi))$$

(A2) *If φ does not contain $\bigcirc$ and $does(.)$,*

$$M \models \varphi \wedge [\![\emptyset]\!]((turn(a) \wedge \varphi \to [N]\varphi) \wedge \bigwedge_{b \in N \setminus \{a\}} (turn(b) \wedge \varphi \to [\emptyset]\varphi)) \to [\![a]\!]\varphi$$

(A3) *If $C \cap D = \emptyset$, then*

$$M \models [\![C]\!]\phi \wedge [\![D]\!]\psi \to [\![C \cup D]\!](\phi \wedge \psi)$$

(A1) is a fixed-point recursion, which says that to maintain a property for a whole game, a player must make sure it is true now and, before the game is terminated, either there is a strategy so that she can maintain the property at next step if it is his turn, or he can maintain the property at next step for all strategies of the player who is taking his turn. Note that the property under consideration must be state-wise (no time spanning). (A2) provides a sufficient condition for a player to construct a strategy that maintains a property. (A3) shows that strategic ability of disjoint coalitions is *superadditive* [8].

It is easy to see that these three statements are the generalisation of Benthem's game-oriented principles (Fact 3: C1-C3 in [7]). We formalise them and generalise them into the multi-agent case. Interestingly, the well-known results for two player games: *weak determinacy* and *Zermelo's theorem* are corollaries of the above theorem.

Proposition 1 (Weak Determinacy). *Let M^z be any state transition model for finite two-player games and $N = \{a, b\}$, then*

$$M^z \models [\![a]\!](terminal \to wins(a)) \vee [\![b]\!]\neg[\![a]\!](terminal \to wins(a))$$

Weak determinacy says in any finite two-player game with perfect information either one player has a winning strategy or the other player has a strategy that ensures her opponent has no winning strategy.

Proposition 2 (Zermelo's Theorem).

$$M^z \models \llbracket a \rrbracket(terminal \to wins(a)) \vee \llbracket b \rrbracket(terminal \to wins(b))$$

$$\vee \llbracket a \rrbracket(terminal \to Tie) \vee \llbracket b \rrbracket(terminal \to Tie)$$

where $Tie =_{def} (\neg wins(a) \wedge \neg wins(b))$.

Zermelo's theorem says in any finite two-player game with perfect information and three outcomes (win, lose and tie), at least one of the players has a strategy to win or to lead to a tie [17].

3.2 Strategy Representation

In this subsection, we discuss how to represent strategies in our language. We try to represent a strategy by a normal formula. We know that a strategy is player-specific and specifies a single action at each reachable state for a player when it is his turn. If there is a formula that satisfies this condition, this formula can then be a syntactical representation of a strategy for that player. This idea leads to the following definition.

Definition 4. *Given a state transition model* M*, a formula* $\varphi \in \mathcal{L}$ *is a strategy rule for player* a *if for all* $w \in W^a \cap W^{\mathcal{P}(M)}$, $\mathcal{A}^a(\varphi, w) = \{\alpha \in A^a : \exists\delta\exists i\ M, \delta, i \models \varphi,\ \delta[i] = w$ *and* $\sigma(\delta, i) = \alpha\}$ *is a singleton.*

Obviously not every formula $\varphi \in \mathcal{L}$ can be a strategy rule. The main challenge is how to create a strategy rule from non-strategy rules (normal formulas). We borrow Zhang and Thielscher's idea to combine strategies using the prioritised strategy connectives [12]. To demonstrate how they work, we use our running example.

Consider a simple mk-game where $m = 5$ and $k = 3$. After some practice or backward induction reasoning, we may find that the following simple ideas may help player x to win:

1. Fill the center.
2. If filling any grid leads to win, do it.
3. Fill a next (an empty grid next to her own symbol).
4. Fill any grid.
5. Try (1) first; if fails, try (2); if fails, try (3); if fails, do (4).

These ideas can be formally represented in GDR language as follows ($a \in \{\mathsf{x}, \mathsf{o}\}$):

1. $fill\_center^a =_{def} does(\alpha^a_{3,3})$
2. $attack^a =_{def} \bigvee_{j,k=1}^{5} (does(\alpha^a_{j,k}) \wedge \bigcirc wins(a))$

3. $fill\_next^a =_{def} \bigvee_{j,k=1}^{5} (p^a_{j,k} \wedge (\ does(\alpha^a_{j-1,k}) \vee does(\alpha^a_{j-1,k-1}) \ \vee does(\alpha^a_{j-1,k+1}) \vee$
   $does(\alpha^a_{j,k-1}) \vee does(\alpha^a_{j,k+1}) \vee does(\alpha^a_{j+1,k-1}) \vee does(\alpha^a_{j+1,k}) \ \vee does(\alpha^a_{j+1,k+1})))$[2]
4. $fill\_any^a =_{def} \bigvee_{j,k=1}^{5} does(\alpha^a_{j,k})$
5. $combined^a =_{def} fill\_center^a \triangledown attack^a \triangledown fill\_next^a \triangledown fill\_any^a$

It is easy to see that the formula $combined^a$ can be satisfied in any state when it is a's turn (due to $fill\_any^a$). However, it is not a strategy rule because it can suggest more than one actions in one state. To make it to be a strategy rule, we need the following technical treatment.

Let $B = \{(i, j) : 1 \le i, j \le 5\}$ be the game board and $\prec$ be the lexicographic order on B, i.e., $(1, 1) \prec (1, 2) \prec \cdots \prec (1, 5) \prec (2, 1) \prec \cdots \prec (5, 5)$. For each grid $\xi \in B$, let $c^a_\xi = \bigvee_{(i,j)\prec\xi} does(\alpha^a_{i,j})$, which represents the idea of player a to fill any grid from $(1, 1)$ up to ξ. We let

$$S^a_{53} =_{def} (fill\_center^a \triangledown attack^a \triangledown fill\_next^a \triangledown fill\_any^a) \vartriangle c^a_{(5,5)} \vartriangle \cdots \vartriangle c^a_{(1,1)}$$

Observation 1. *Let M be the state transition model of* 53*-game. For any player $a \in \{\mathsf{x}, \mathsf{o}\}$, S^a_{53} is a strategy rule for player a.*

3.3 Reasoning about Strategic Abilities of Game Players

In this subsection, we continue to use mk-games to demonstrate how to reason about strategic abilities of a player.

A standard strategy stealing argument from combinatorial game theory shows that there is no winning strategy for the second player in any mk-game. This can be easily described as follows:

Proposition 3. *Let M^{mk} be any state transition model for mk-games. Then*

$$M^{mk} \models \neg[\![\mathsf{o}]\!](terminal \to wins(\mathsf{o}))$$

By Proposition 2, we derive that the first player x has a no-losing strategy:

$$M^{mk} \models [\![\mathsf{x}]\!](terminal \to \neg wins(\mathsf{o}))$$

According to Observation 1, S^a_{53} is a strategy rule for player a. We now prove that this is actually a winning strategy for the player who has the first move in 53-game. To this end, we say that a state transition model M *complies with* player a's strategy f_a specified by a strategy rule S^a, denoted by M_{S^a}, if for all complete paths $\delta \in \mathcal{P}(M)$, δ complies with f_a. Formally, $\mathcal{P}(M_{S^a}) = \{\delta \in \mathcal{P}(M) : M, \delta \models turn(a) \to S^a\}$. With this notion, we can reason about that the strategy specified by S^{x}_{53} is a winning strategy for player x in 53-game.

Observation 2. *For all $\delta \in \mathcal{P}(M_{S^{\mathsf{x}}_{53}})$, $M^{53}, \delta \models terminal \to wins(\mathsf{x})$*

[2] To avoid too much complexity, we ignore the cases when the indexes go over their range.

This is just a simple example. In fact, our language can be used to describe more complicated strategies for more complicated games. We have provided a constructive approach to show Tic-tac-toe game can be forced in a draw. Currently we are working on the logical description of the winning strategy for Gomoku game, invented by Allis *et al.* [18, 19], and formally prove it to be a winning strategy instead of the computer assisted proof [20–22].

4 Model Checking

The *model-checking problem* for GDR is the problem of determining: for a given GDR formula φ, a state transition model M, a complete path δ and a stage i in M, whether or not $M, \delta, i \models \varphi$. By establishing a translation from GDR to ATL*, we show an upper bound of the model-checking problem for GDR, which is no worse than the model-checking problem for ATL*.

4.1 From GDR Model to ATL* Model

In this part, we will show any state transition model can be transformed into a ATL* model, using the methods in [23]. The main idea is that we encode notions like *terminal, turn, wins* through valuation π, rather than through separate relations or functions. For this purpose, we redefine the set of *atomic propositions* of GDR, denoted by At_{GDR}, as follows:

$$At_{\text{GDR}} =_{def} \Phi \cup \{terminal\} \cup \{turn(a), wins(a) \mid a \in N\}$$

Given a state transition model $M = (N, W, \mathcal{A}, \overline{w}, T, U, g, t, V)$ and a set of atomic propositions At_{GDR}, we define an associate *action-based alternating transition system*(AATS) $\mathcal{T}_M = \{W', \overline{w}, N,\ A'_1, \cdots, A'_n, \rho, \tau, \Phi', \pi\}$ with the same set of agents $N = \{1, \cdots, n\}$ and initial state $\overline{w}$, and such that Φ' is constructed in the following manner.

Definition 5. *Define a translation ST: $At_{GDR} \to At_{ATL^\star}$ associating every atom in At_{GDR} with an atom in AT_{ATL^*}:*

$$\begin{array}{ll} ST(turn(a)) = turn(a) & ST(wins(a)) = wins(a) \\ ST(terminal) = terminal & ST(p) = p \text{ for all } p \in \Phi \end{array}$$

Then we next define the set of atomic propositions Φ' as a set of atoms such that

- for all $\tilde{p} \in At_{\text{ATL}^\star}$, $\tilde{p} \in \Phi'$.
- $done(\alpha) \in \Phi'$ for all $\alpha \in \bigcup_{i \in N} A'_i$ representing actions that are done in the transition from previous state to current state[3].
- $initial \in \Phi'$ and $s_\perp \in \Phi'$ where $s_\perp$ is a special atom to specify a 'sink state' which is the only successor of a terminal state and itself.

The other components of $\mathcal{T}_M$ are constructed as follows:

[3] Note this concept as well as the following two concepts sink state and fin_i is borrowed from [23].

- $W' = W_1 \cup W_2$ where $W_1 = W$ and $W_2 = \{s_w \mid w \in W_1\}$ including sink states.
- $A'_i = A_i \cup \{noop_i\} \cup \{fin_i\}$ where for all $1 \le i \le n$ A_i is the same as that in M, $noop_i$ means player i does nothing, and fin_i is an action for the terminal and sink states
- $\rho : \bigcup_{i \in N} A'_i \mapsto W'$ is an action precondition function such that
  - for any $\alpha \in \bigcup_{i \in N} A_i$, $\rho(\alpha) = \{w \in W \mid U(w, \alpha) \text{ is defined.}\}$;
  - for any $1 \le i \le n$, $\rho(noop_i) = W \backslash T$;
  - for any $1 \le i \le n$, $\rho(fin_i) = W_2 \cup T$.
- $\tau : A'_1 \times \cdots \times A'_n \times W' \hookrightarrow W'$ is a partial system transition function such that
  - for all $\overrightarrow{\alpha} \in A_1 \times \cdots \times A_n$ and $w \in W$, if $t(w) = i$, then $\tau(\overrightarrow{\alpha}, w) = U(\alpha_i, w)$ where $\alpha_i \in A_i$;
  - for all $w \in T$, $\tau(\langle fin_1, \cdots, fin_n \rangle, w) = s_w$;
  - for all $s_w \in W_2$, $\tau(\langle fin_1, \cdots, fin_n \rangle, s_w) = s_w$.
- $\pi : W' \mapsto 2^{\Phi'}$ is a valuation function such that for any $w \in W'$ $\pi(w)$ is a set of atoms satisfying the following conditions:
  1. for all $w \in W_1$,
     - for any $p \in \Phi$, $p \in V(w)$ iff $p \in \pi(w)$;
     - $w \in T$ iff $terminal \in \pi(w)$;
     - for any $a \in N$, $t(w) = a$ iff $turn(a) \in \pi(w)$;
     - for any $a \in N$, $w \in g(a)$ iff $wins(a) \in \pi(w)$;
     - $initial \in \pi(\overline{w})$;
     - $done(\alpha) \in \pi(w)$ iff $\sigma(\delta, i) = \alpha$ and $\delta[i+1] = w$.
  2. for all $s_w \in W_2$, $\pi'(s_w) = \pi'(w) \cup \{s_\perp\} \cup \{done(fin_i) \mid 1 \le i \le n\} \backslash \{done(\alpha) \mid \alpha \in \bigcup_{i \in N} A_i\}$.

Given a complete path δ in M, we extend δ to an infinite path (computation) in $\mathcal{T}_M$ with the sink state labelled by the terminal state of δ. We denote it by $\tilde{\delta}$. We use $\tilde{\delta}[i, \infty]$ to denote the infinite subsequence of $\tilde{\delta}$ starting at stage i.

4.2 Translation from GDR to ATL*

We next define a translation map from GDR formulas to ATL* formulas to make GDR embedded into ATL*.

Definition 6. *A translation $ST^\star$ from GDR formulas to ATL* formulas is defined as follows:*

- $ST^\star(\tilde{p}) = ST(\tilde{p})$ *for all* $\tilde{p} \in At_{GDR}$
- $ST^\star(initial) = initial$
- $ST^\star(\neg\varphi) = \neg ST^\star(\varphi)$
- $ST^\star(\varphi_1 \wedge \varphi_2) = ST^\star(\varphi_1) \wedge ST^\star(\varphi_2)$
- $ST^\star(does(\alpha)) = \bigcirc done(\alpha)$
- $ST^\star(\varphi_1 \triangledown \varphi_2) = ST^\star(\varphi_1) \vee (\langle\langle\emptyset\rangle\rangle \neg ST^\star(\varphi_1) \wedge ST^\star(\varphi_2))$
- $ST^\star(\bigcirc\varphi) = \bigcirc(\neg s_\perp \rightarrow ST^\star(\varphi))$
- $ST^\star([C]\varphi) = \langle\langle C \rangle\rangle \bigcirc (\neg s_\perp \rightarrow ST^\star(\varphi))$
- $ST^\star([\![C]\!]\varphi) = \langle\langle C \rangle\rangle (\Box)(\neg s_\perp \rightarrow ST^\star(\varphi))$

With this, we have the following correspondent result between the state transition model and its associated AATS with respect to the translation.

Lemma 1. *Given a state transition model M, a complete path δ in M and a stage i on δ, for any GDR formula $\varphi \in \mathcal{L}$,*

(1) *if $ST^{\star}(\varphi)$ is an ATL* state formula, $M, \delta, i \models_{GDR} \varphi$ iff $\mathcal{T}_M, \tilde{\delta}[i] \models_{ATL^{\star}} ST^{\star}(\varphi)$;*
(2) *if $ST^{\star}(\varphi)$ is an ATL* path formula, $M, \delta, i \models_{GDR} \varphi$ iff $\mathcal{T}_M, \tilde{\delta}[i, \infty] \models_{ATL^{\star}} ST^{\star}(\varphi)$.*

It follows that the model-checking problem for GDR is no more complicated than ATL*. Since *the model-checking problem for ATL* is 2EXPTIME-complete*[1], then we have the following result.

Theorem 2. *The model-checking problem for GDR is in 2EXPTIME.*

5 Conclusion

We have presented a unified logical framework for game description, strategy representation and strategic reasoning. The language of the framework combines GDL, ATL and prioritised strategy connectives. To minimize the complexity of this language, we took a cautious way of doing that. We do not introduce until operator $\mathcal{U}$ and $legal()$. We have demonstrated that our language is rich enough to express generic game results such as Weak Determinacy and Zermelo's Theorem as well as complicated game strategies.

Most of the related work has been discussed in the introduction. Besides that, the following is also worth mentioning.

Ruan *et al.* studied the relationship between GDL and ATL [23]. Different from our motivation, their goal is to use ATL to reason about GDL-specified games. Therefore, instead of integrating language components, they focus on how to transfer a GDL game specification into an ATL specification.

We would also like to mention that there are three essential differences between Zhang and Thielscher's work [12] and ours. Firstly, their definition of the prioritized connectives was based on the semantics of strategies rather than on the semantics of the logic. Strictly speaking, their prioritized connectives are not part of their logical language. However, our ones are part of the logical language. Secondly, we define these connectives as binary operators while theirs are multiple tuple operators. The prioritized conjunction operators diverge when arguments are more than two due to non-associativity. Thirdly, their work does not have the facility for reasoning about strategic abilities of players.

There are many directions for future investigations. Firstly, our approach may be used in the development of general game players. Since GDL is the native language for general game playing, with further extension of prioritised strategy connectives, a player would be able to combine simple actions into more complicated actions. Secondly, with our approach we may be able to provide logical solutions for already solved games, such as Connect Four and Gomoku, instead of the computer assisted solutions [20–22]. With the help of model-checking approach, we may be also develop solutions for some unsolved games. Moreover, it would be interesting to investigate the epistemic extension of the current framework so as to study the beliefs of other players' strategies.

References

1. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. Journal of the ACM 49(5), 672–713 (2002)
2. Chatterjee, K., Henzinger, T.A., Piterman, N.: Strategy logic. Information and Computation 208(6), 677–693 (2010)
3. van der Hoek, W., Jamroga, W., Wooldridge, M.: A logic for strategic reasoning. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 157–164. ACM (2005)
4. Kaneko, M., Nagashima, T.: Game logic and its applications. Studia Logica 57(2/3), 325–354 (1996)
5. Parikh, R.: The logic of games and its applications. In: Karplnski, M., van Leeuwen, J. (eds.) Topics in the Theory of Computation Selected Papers of the International Conference on 'Foundations of Computation Theory', FCT 1983. North-Holland Mathematics Studies, vol. 102, pp. 111–139. North-Holland (1985)
6. Pauly, M., Parikh, R.: Game logic-an overview. Studia Logica 75(2), 165–182 (2003)
7. van Benthem, J.: Reasoning about strategies. In: Computation, Logic, Games, and Quantum Foundations. The Many Facets of Samson Abramsky, pp. 336–347. Springer (2013)
8. Pauly, M.: A modal logic for coalitional power in games. Journal of Logic and Computation 12(1), 149–166 (2002)
9. Genesereth, M., Love, N., Pell, B.: General game playing: Overview of the AAAI competition. AI Magazine 26(2), 62–72 (2005)
10. Mogavero, F., Murano, A., Vardi, M.Y.: Reasoning about strategies. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, pp. 133–144. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2010)
11. Ramanujam, R., Simon, S.E.: Dynamic logic on games with structured strategies. In: Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR 2008), pp. 49–58 (2008)
12. Zhang, D., Thielscher, M.: Representing and reasoning about game strategies. To Appear in J. Philosophical Logic (2014)
13. van Benthem, J.: In praise of strategies. In: Eijck, J.V., Verbrugge, R. (eds.) Games, Actions, and Social Software. ILLC scientific publications, Institute for Logic, Language and Computation (ILLC), University of Amsterdam (2008)
14. Herzig, A., Lorini, E., Walther, D.: Reasoning about actions meets strategic logics. In: Grossi, D., Roy, O., Huang, H. (eds.) LORI 2013. LNCS, vol. 8196, pp. 162–175. Springer, Heidelberg (2013)
15. Walther, D., van der Hoek, W., Wooldridge, M.: Alternating-time temporal logic with explicit strategies. In: Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge, pp. 269–278. ACM (2007)
16. Van den Herik, H.J., Uiterwijk, J.W., Van Rijswijck, J.: Games solved: Now and in the future. Artificial Intelligence 134(1), 277–311 (2002)

17. Polak, B.: Backward induction: Chess, strategies, and credible threats (2007), http://oyc.yale.edu/economics/econ-159/lecture-15
18. Allis, L.V., van den Herik, H.J., Huntjens, M.P.H.: Go-moku solved by new search techniques. Computational Intelligence 12, 7–23 (1996)
19. Allis, L., van der Meulen, M., van den Herik, H.: Proof-number search. Artificial Intelligence 66(1), 91–124 (1994)
20. Allis, L.V.: A knowledge-based approach of connect-four. Vrije Universiteit, Subfaculteit Wiskunde en Informatica (1988)
21. Allis, L.V.: Searching for solutions in games and artificial intelligence. Ph.D. thesis, University of Limburg, The Netherlands (1994)
22. Wágner, J., Virág, I.: Solving renju. ICGA Journal 24(1), 30–35 (2001)
23. Ruan, J., Van Der Hoek, W., Wooldridge, M.: Verification of games in the game description language. Journal of Logic and Computation 19(6), 1127–1156 (2009)