Game Description Logics and Game Playing

Laurent Perrussel – Guifei Jiang – Dongmo Zhang

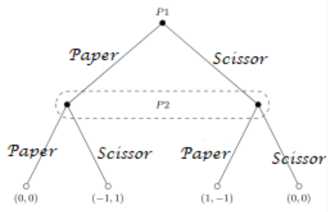
October 27, 2018 / KR-18 tutorial

IRIT – Université Toulouse Capitole (FR)
Nankai University (CN)
Western Sydney University (AU)

Motivation

Motivation(1/2)

- Game: describe and justify actions in a multi-agent context



- Autonomy for agent means
  - Decision making: justify actions (agent rationality)

    P1 plays scissor because...

  - handling or playing in different environments (facing a new game)

    P2 now plays Tic-tac-toe

Motivation (2/2)

Computer Science vs Game Theory?

- Game Theory

  Main goal: assessing the graph (i.e. the game) and find equilibrium or existence of winning strategies

- Computer Science

  Main goal: compact representation, computation of the possible next actions and choice

General Game Playing
Computer scientists challenge: build programs sufficiently general for playing different games.

Organization

Motivation

General Game Playing

Game Description Language

Implementing a Player

Game Description Logic: GDL with a (logic-flavored) semantics

Imperfect Information

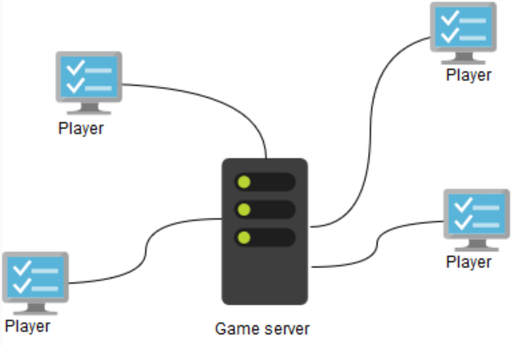
Reasoning for winning?

Still a lot to do! - Example: Equivalent games

Perspectives

General Game Playing

General Game Playing - Overall organization



More details at `http://ggp.org` and in [GT14]

Interaction between server and players:

$\Rightarrow$ Game rules & current state of the game

$\Leftarrow$ Moves

General Game Playing - Prerequisites

Limited to the shared aspect of the game

- **Type of game** No randomness - perfect information (board game)
- **Language** Processable by the server and players (game rules)
- **Timeclock** sync player moves and game run

  No prerequisite on players implementation (reasoning is not compulsory!)

General Game Playing - Key challenge

Overall goal: designing intelligent agent

Building players sufficiently general for playing different games

GGP competition: players compete by playing at different games.

*Challenge is **not** to build the best player for one game*

GGP player will never beat AlphaGo (at least in a Go game!)

General Game Playing - Specialized player

- Usually rules of the game hard-coded in the player
- Possibly exhaustive search
- Predefined library of best moves (tactics, ie. library of plans) combined with heuristics
- Library can be learned

General Game Playing - Representing games

Game Description Language (GDL)

- General

  General enough for describing different games: no primitives related to some specific game

- Game rules and remarkable states

  Initial and final states, legal actions...

- Compact

  Logic-based language, namely ***first-order logic***

General Game Playing - Processing GDL

Server

- Not relevant - Zero intelligence

Players

- No specific implementation

  Several implementation are available (Java, Prolog...)

- No specific way to play

  Reasoning, Heuristics, Monte-Carlo, CSP...

General Game Playing - GDL Example

Tic-Tac-Toe

Tic Tac Toe (or Noughts and Crosses, Xs and Os) is a game for two players who take turns placing their marks in a 3x3 grid. The first player to place three of his marks in a horizontal, vertical, or diagonal row wins the game.

General Game Playing - GDL Example

Tic-Tac-Toe GDL representation (1/3)

```
;;; Components
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
    (role white)
    (role black)
    ...

;;; init
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
    (init (cell 1 1 b))
    ...
    (init (cell 3 3 b))
    (init (control white))
```

General Game Playing - GDL Example

Tic-Tac-Toe GDL representation (2/3)

```
;;; legal moves
    (<= (legal ?w (mark ?x ?y))
        (true (cell ?x ?y b))
        (true (control ?w)))

    (<= (legal white noop)
        (true (control black)))
    ...

;;; next (effects)
    (<= (next (cell ?m ?n x))
        (does white (mark ?m ?n))
        (true (cell ?m ?n b)))
    ...
```

General Game Playing - GDL Example

Tic-Tac-Toe GDL representation (3/3)

```
;;; goal
    (<= (goal white 100)
        (line x)
        (not (line o)))

    (<= (goal white 0)
        (not (line x))
        (line o))
    ...

;;; terminal

    (<= terminal
        (line x))
    ...
```

Game Description Language

GDL - Primitives

Prolog/Datalog like rules with predefined keywords (prefix notation)

Static perspective

- `role` players of the game

  (role white)

- `init` initial state

  (init (cell 1 1 b))

- `terminal` terminal state

  (<= terminal (line x))

- `true` current state

  (true (cell 2 2 b))

GDL - Primitives (2/2)

Dynamic perspective

- `legal` rules of the game - possible moves

  (<= (legal x noop) (true (control o)))

- `does` performing action (in the current state)

  (<= (next (cell ?x ?y ?player)) (does ?player (mark ?x ?y)))

- `next` update function

  (<= (next (control o)) (true (control x)))

- `goal` objectives of the players

  (<= (goal ?player 100) (line ?player))

GDL - syntax constraints

"Enforcing" game flavor

- sequence of keywords is *prohibited*
- `role` only atomic (fixed players)
- `next` predicate only in heads
- `init` and `true` predicates only in bodies
- `does` predicate only in bodies
- recursion restriction

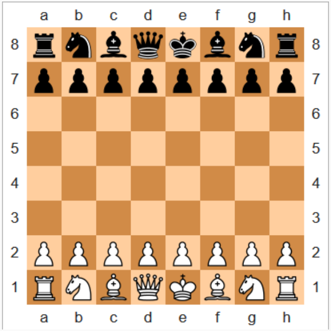
GDL - semantics

A logic programming perspective

- Minimal data set D which are models of a game G: set of grounded atoms
  - ground literal (`not p`) is satisfied iff `p` is not in D
- GDL game description: logic program with predefined predicate and shape
  - Complete definition of `role`, `init`
  - `legal` and `goal` only defined wrt `true`
  - `next` only defined wrt `true` and `does`
- Unique minimal model satisfying the state of the game (ie `true` predicate)
- Several minimal models when considering the dynamics (ie `does` predicate)

GDL - Chess example (1/6)

- Around 1000 lines!
- initial state already complex
- legal moves differ for each piece type
- basic rules + specific rules (pawn promotion...)
- no number in GDL: rules for encoding them!



GDL - Chess example (2/6)

Initial state

- Two players
- Chess board and pieces
  - blank cells
  - black and white rooks (`wr`, `br`)
  - black and white pawn (`wp`)
- First player

```
(role white)
(role black)
(init (cell a 1 wr))
(init (cell a 2 wp))
(init (cell a 3 b))
...
(init (cell h 8 br))
(init (control white))
```

GDL - Chess example (3/6)

Goal states

- Check mate the opponent
  - ⇒ should be defined for the `white` and `black` players
- Draw is a good compromise
- Not being checkmate is also a goal!

```
...
(<= (goal white 100)
    (checkmate black))
(<= (goal white 50)
    stalemate)
(<= (goal white 0)
(checkmate white))
...
```

GDL - Chess example (4/6)

End of the game

- One player is stuck
  - ⇒ regardless king is in check or not
- After 200 rounds, game is stopped
  - ⇒ Numbers and counting should be defined

```
(<= (stuck ?pl)
    (role ?pl)
    (not (has_legal_move ?pl)))
...
(<= terminal
    (true (control ?player))
    (stuck ?player))
(<= terminal
    (true (step 201)))
...
(succ 1 2)
(succ 2 3)
...
```

GDL - Chess example (5/6)

Legal moves

- Define the moves for each piece
  - what means adjacent?
  - what means diagonal?
  - ...
- Define legality
  - context is OK (players, piece is on the cell, move is meaningful...)

```
(<= (knight_move ?piece ?u ?v
                  ?x ?y ?owner)
    (piece_owner_type ?piece
                 ?owner knight)
    (adjacent_two ?v ?y)
    (adjacent ?u ?x))
...
(<= (legal ?player (move ?piece
                      ?u ?v ?x ?y))
  (true (control ?player))
  (true (cell ?u ?v ?piece))
  (occupied_by_opp ?x ?y ?player)
  (legal2 ?player (move ?piece
                   ?u ?v
                   ?x ?y))
   ...
```

GDL - Chess example (6/6)

Actions and update

- General rules for the game

  e.g. blank cell
- specific rules for specific moves

  e.g. "en passant"
- update the step number

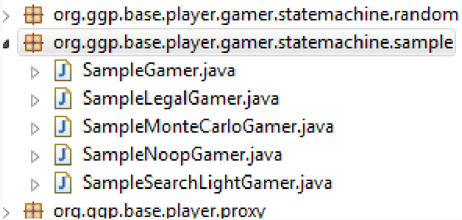
```
(<= (next (cell ?u ?v b))
      (does ?player (move ?p ?u
                     ?v ?x ?y)))

(<= (next (cell ?x1 ?y1 b))
    (does ?player (move ?piece
       ?x1 ?y1 ?x2 ?y2))
     (pawn_capture_en_passant
       ?player ?x1 ?y1 ?x2 ?y2))

(<= (next (step ?y))
    (true (step ?x))
        (succ ?x ?y))
```

Implementing a Player

Implementing a Player

- Free implementation
- Reasoning is not compulsory
- Main technique:
  - Search-Space and Heuristics
  - Compute the value of the next state

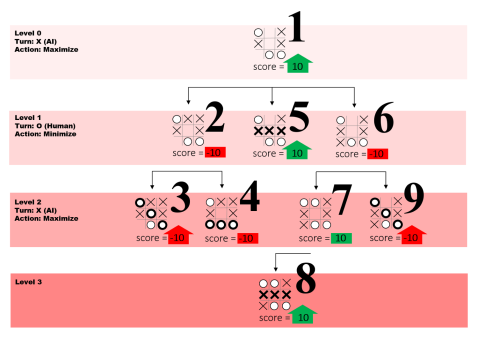
eg. (1) Minimax
eg. (2) Monte-Carlo Tree Search



Minimax

Extensive search

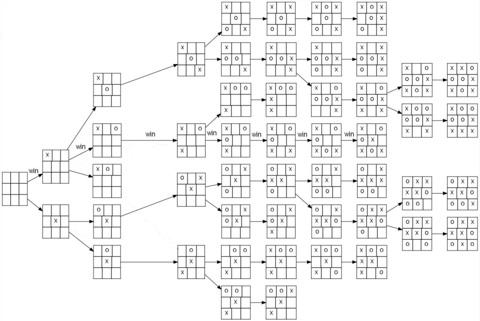
- depth first
- min value for the opponents (worst case)
- max value for myself (best case)
- depth of search may be limited (heuristics)



Minimax

Monte Carlo Tree Search

- Run random simulation
- Sampling the game tree
- Estimation of actions
- MCTS converge to Minimax



Game Description Logic: GDL with a (logic-flavored) semantics

Signature and Language

Towards reasoning about Perfect Information Games

First step is to build a logic based on GDL [Jia16]

Signature Agents, actions, propositions:

$$(N, \mathcal{A}, \Phi)$$

Language predefined symbols and temporal operators

$$\varphi \ ::= \ p \mid \mathit{initial} \mid \mathit{terminal} \mid \mathit{legal}(r, a) \mid \mathit{wins}(r) \mid \mathit{does}(r, a) \mid \neg\varphi \mid \varphi \wedge \psi \mid \bigcirc\varphi$$

Tic-Tac-Toe

GDL description of Tic-tac-Toe:

1. $\mathit{initial} \leftrightarrow \mathit{turn}(\mathsf{x}) \wedge \neg \mathit{turn}(\mathsf{o}) \wedge \bigwedge_{i,j=1}^{3} \neg(p^{\mathsf{x}}_{i,j} \vee p^{\mathsf{o}}_{i,j})$
2. $\mathit{wins}(r) \leftrightarrow \bigvee_{i=1}^{3} \bigwedge_{l=0}^{2} p^{r}_{i,1+l} \vee \bigvee_{j=1}^{3} \bigwedge_{l=0}^{2} p^{r}_{1+l,j} \vee \bigwedge_{l=0}^{2} p^{r}_{1+l,1+l} \vee \bigwedge_{l=0}^{2} p^{r}_{1+l,3-l}$
3. $\mathit{terminal} \leftrightarrow \mathit{wins}(\mathsf{x}) \vee \mathit{wins}(\mathsf{o}) \vee \bigwedge_{i,j=1}^{3} (p^{\mathsf{x}}_{i,j} \vee p^{\mathsf{o}}_{i,j})$
4. $\mathit{legal}(r, a_{i,j}) \leftrightarrow \neg(p^{\mathsf{x}}_{i,j} \vee p^{\mathsf{o}}_{i,j}) \wedge \mathit{turn}(r) \wedge \neg \mathit{terminal}$
5. $\mathit{legal}(r, \mathit{noop}) \leftrightarrow \mathit{turn}(-r)$
6. $\bigcirc p^{r}_{i,j} \leftrightarrow p^{r}_{i,j} \vee (\mathit{does}(r, a_{i,j}) \wedge \neg(p^{\mathsf{x}}_{i,j} \vee p^{\mathsf{o}}_{i,j}))$
7. $\mathit{turn}(r) \rightarrow \bigcirc \neg \mathit{turn}(r) \wedge \bigcirc \mathit{turn}(-r)$

State-Transition Model (Perfect-Information Game)

$$M = (W, I, T, L, U, g, \pi)$$

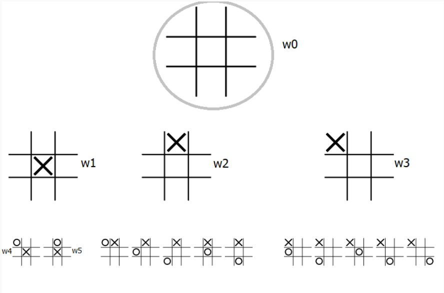
- W is a non-empty finite set of *possible states*.
- $I \subseteq W$, representing a set of *initial* states.
- $T \subseteq W \setminus I$, representing a set of *terminal* states.
- $L \subseteq W \backslash T \times N \times 2^{\mathcal{A}}$ is a *legality* relation, specifying legal actions for each agent at non-terminal states. Let $L_r(w) = \{a \in \mathcal{A} \ : \ (w, r, a) \in L\}$ be the set of all legal actions for agent r at state w. To make the game playable, we require $L_r(w) \neq \emptyset$ for every $r \in N$ and $w \in W \backslash T$.
- $U : W \times \mathcal{A}^{|N|} \rightarrow W \backslash I$ is an *update* function, specifying the state transition for each state and *joint action* (synchronous moves).
- $g : N \rightarrow 2^W$ is a *goal* function, specifying the winning states of each agent.
- $\pi : W \rightarrow 2^{\Phi}$ is a standard valuation function.

$M = (W, I, T, L, U, g, \pi)$

- Set of states W can be very large

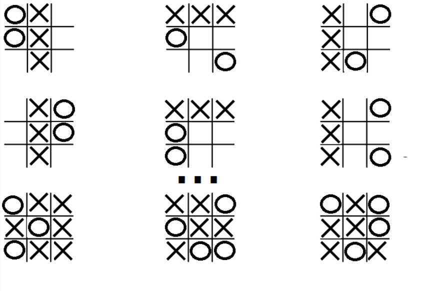
  5 478 states for Tic-Tac-Toe

- Set $I = \{w_0\}$ usually a singleton



$M = (W, I, T, L, U, g, \pi)$

- Set T of terminal states consider all cases
  - *958 terminal states*
- winning or draw states
- winning states g specific to each agent and subset of T



ST Model - Details (3/3)

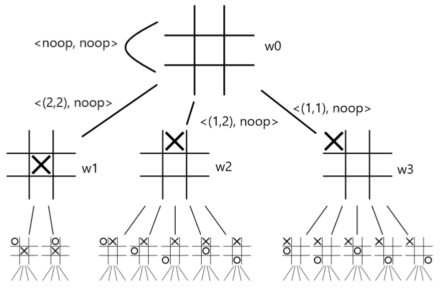
$M = (W, I, T, L, U, g, \pi)$

- Legal transitions (L)

  9 legal actions from $\langle(1,1), noop\rangle$ *to* $\langle(3,3), noop\rangle$ *in* w_0

- Update is deterministic.

  Update can be defined while illegal (eg. $\langle noop, noop\rangle$



Path

Path δ is an infinite sequence of states and actions

$$w_0 \xrightarrow{d_1} w_1 \xrightarrow{d_2} w_2 \cdots \xrightarrow{d_j} \cdots$$

such that for all $j \geq 1$ and for any $r \in N$,

1. $w_j = U(w_{j-1}, d_j)$ (state update);
2. $(w_{j-1}, d_j(r)) \in L_r$ (that is, any action that is taken must be legal);
3. if $w_{j-1} \in T$, then $w_{j-1} = w_j$ (that is, a loop after reaching a terminal state).

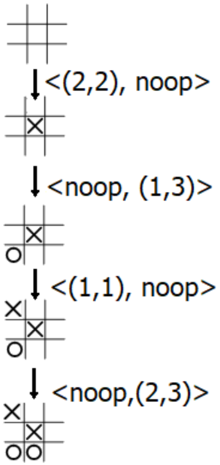
$\theta_r(\delta, j)$: action of agent r at stage j of δ

Path

Sequence of actions

- Run over an ST-model
- No requirement about first and last states
- formulas will be interpreted over a path at some step
- $\delta[j]$: jth state of path δ
- $\theta_r(\delta, j)$ action performed by agent r at state j of path δ

$$\text{eg: } \theta_x(\delta, 3) = a_{1,1}$$



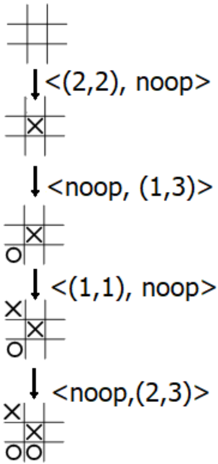
Semantics

W.r.t. M, some path δ and index j

$M, \delta, j \models p$	iff	$p \in \pi(\delta[j])$
$M, \delta, j \models \neg\varphi$	iff	$M, \delta, j \not\models \varphi$
$M, \delta, j \models \varphi_1 \wedge \varphi_2$	iff	$M, \delta, j \models \varphi_1$ and $M, \delta, j \models \varphi_2$
$M, \delta, j \models \mathit{initial}$	iff	$\delta[j] \in I$
$M, \delta, j \models \mathit{terminal}$	iff	$\delta[j] \in T$
$M, \delta, j \models \mathit{wins}(r)$	iff	$\delta[j] \in g(r)$
$M, \delta, j \models \mathit{legal}(r, a)$	iff	$a \in L_r(\delta[j])$
$M, \delta, j \models \mathit{does}(r, a)$	iff	$\theta_r(\delta, j) = a$
$M, \delta, j \models \bigcirc\varphi$	iff	$M, \delta, j+1 \models \varphi$

Tic-Tac-Toe formulas

- $M, \delta, 0 \models \neg p^x_{1,1}$
- $M, \delta, 1 \models p^x_{2,2}$
- $M, \delta, 1 \models \neg wins(x)$
- $M, \delta, 1 \models does(o, a_{1,3})$
- $M, \delta, 2 \models \bigcirc does(o, a_{2,3})$
- $M, \delta, 4 \models \neg \bigcirc wins(x)$



What about proof theory?

Mainly consists of axiom schemas for $\bigcirc$, modus ponens inference rule and general game properties:

Axioms

1. All tautologies of classical propositional logic.
2. $\bigcirc(\varphi \to \psi) \to (\bigcirc\varphi \to \bigcirc\psi)$
3. $\neg \bigcirc \varphi \to \bigcirc\neg\varphi$

Axioms for general game properties

4. $\neg \bigcirc \mathit{initial}$
5. $\mathit{terminal} \to \bigwedge_{a^r \in A^r \setminus \{noop^r\}} \neg \mathit{legal}(a^r) \wedge \mathit{legal}(noop^r)$
6. $\bigvee_{a^r \in A^r} \mathit{does}(r, a)$
7. $\neg(\mathit{does}(r, a) \wedge \mathit{does}(r, b))$ for $a^r \neq b^r$.
8. $\mathit{does}(a^r) \to \mathit{legal}(a^r)$
9. $\varphi \wedge \mathit{terminal} \to \bigcirc\varphi$

GDL for reasoning about games

General game properties

- $\models \bigvee_{r \in N} wins(r) \to terminal$ iff $g(r) \subseteq T$

Bounded time

- $\not\models \bigwedge_{i \in 1..n} \bigcirc^{i} \neg wins(r) \to \bigcirc^{n+1} \neg wins(r)$

[0] $\bigcirc^{n}$: sequence of n $\bigcirc$

GDL for reasoning about games

General game playing w.r.t. some ongoing game

- assessing a "strategy" vs $\langle$game state, move$\rangle$ (M, δ)
- Look ahead via model checking (**PTIME**)
- Winning move (encoded in δ)?

$$M, \delta, 0 \models \bigcirc \mathit{wins}(x)$$

- Prevent opponent x to win?
  - Choose an action a for x and an action b for $-x$ next move

  $\Rightarrow$ Check $M, \delta, 0 \models \bigcirc \mathit{does}(-x, b) \wedge \bigcirc^2 \mathit{wins}(-x)$
  - Choose alternative action a' for x

  $\Rightarrow$ Check $M, \delta', 0 \models \bigcirc \mathit{does}(-x, b) \wedge \bigcirc^2 \neg \mathit{wins}(-x)$
  - Choose other b' and recheck
- No meta-reasoning in GDL (assessment over paths)

"Try to win, if not prevent to loose" cannot be represented

GDL for reasoning about games

Specific game properties

- Set of rules specific to a game
- Identify pattern for general game playing
- Example: Tic-Tac-Toe
  - $\mathit{diagonal}(x) \leftrightarrow \bigwedge_{i \in 1..3} p^x_{i,i} \vee \bigwedge_{i \in 0..2} p^x_{1+i,3-i}$
  - $\mathit{line}(x) \leftrightarrow \mathit{diagonal}(x) \vee \mathit{column}(x) \vee \mathit{row}(x)$
  - Double threat consequence of move a by x: two potential lines
  - Meta-reasoning as two paths are considered (eg: row or column):

    For any next move b by $-x$, pick up x move c and c', build path δ, δ' and check

    $$M, \delta, 0 \models \bigcirc \mathit{row}(x) \text{ or } M, \delta', 0 \models \bigcirc \mathit{column}(x)$$

Exercise

(Simplified) Nim Game

- 2 players sequential game
- 12 sticks
- at each round, each player picks 1, 2 or 3 sticks
- winner of game: the player picking the last stick

Provide the GDL representation

GDL and other logical systems

GDL and Linear Time Logic (LTL)

- GDL close to temporal logic
- Temporal logic: modal logic.
- Temporal operators: *next*, *sometimes*, *always*
- No existential and universal modality in GDL

  "Statement φ will hold at some stage" cannot be represented

GDL and other logical systems

GDL and Propositional Dynamic Logic (PDL)

- PDL formulas: $[\alpha]\varphi$ s.t. $[\alpha]\varphi =_{def} \neg\langle\alpha\rangle\neg\varphi$
- α limited to atomic program and sequence
- Interpretation over Kripke structure $M = (W, R_\alpha, v)$
- PDL semantics
  - $M, w \models p \iff p \in v(w)$
  - $M, w \models [\alpha]\varphi$ iff for all $w' \in R_\alpha$, $M, w' \models \varphi$

GDL and other logical systems

GDL and Propositional Dynamic Logic (PDL)

- Mapping between GDL and PDL
- First step: map the signature and formulas
- Second step: map the model (interpretations and paths)
- Third step: mapping result

$$M_{GDL}, \delta_{GDL}, j \models_{GDL} \varphi \iff M_{PDL}, w_j \models_{PDL} tr(\varphi)$$

GDL and Propositional Dynamic Logic (PDL)

- Map GDL signature $(N, \mathcal{A}, \Phi)$ and PDL propositional symbols P

$$P = \Phi \cup \{initial, terminal\} \cup \{done(r, a) \mid r \in N \text{ and } a \in \mathcal{A}\} \cup \{wins(r) \mid r \in N\}$$

- Map between GDL and PDL formulas: Function tr
  - $tr(\bigcirc\varphi) = \bigvee_a [a]\varphi$ s.t. a is a joint action
  - $tr(does(r, a)) = [a^r]done(r, a)$
  - $tr(legal(r, a)) = legal(r, a)$
- No more agents and temporal representation

GDL and Propositional Dynamic Logic (PDL)

- Construct a $N_{M,\delta} = (W', R_a, v)$ with respect to (i) joint actions a, (ii) an ST-model $M = (W, I, T, L, U, g, \pi)$ and (iii) path δ:
  1. $W' = W$
  2. For any w:
     - For all $p \in \Phi$, $p \in \pi(w)$ iff $p \in v(w)$
     - $w \in I$ iff $initial \in v(w)$
     - $w \in T$, iff $terminal \in v(w)$
     - $w \in g(r)$ iff $wins(r) \in v(w)$
     - $b \in L_r(w)$ iff $legal(r, b) \in v(w)$
     - $U(w, a) = w'$ iff $wR_a w'$
     - $\theta_r(\delta, w) = a^r$ iff $done(r, a^r) \in v(U(w, a))$
- Interplay

$$M, \delta, j \models_{GDL} \varphi \iff N_{M,\delta}, w_j \models_{PDL} tr(\varphi)$$

GDL and other logical systems

GDL and Linear Time Logic (LTL)

- LTL formulas: $\mathbf{X}\varphi, \mathbf{G}\varphi, \mathbf{F}\varphi$ s.t. $G\varphi =_{def} \neg\mathbf{F}\neg\varphi$
- binary operator: $\varphi\mathbf{U}\psi$ (**U***ntil*)
- Interpretation over paths: infinite sequence δ of propositional interpretations ω
- LTL semantics
  - $\delta \models p \iff p \in \delta[0]$
  - $\delta \models \mathbf{X}\varphi \iff \delta[1..\infty] \models \varphi$
  - $\delta \models \mathbf{F}\varphi \iff$ there exists $i \in 0..\infty$, s.t. $\delta[i..\infty] \models \varphi$

GDL and other logical systems

GDL and Linear Time Logic (LTL)

- Mapping between GDL and LTL
- First step: map the signature
- Second step: map the model (interpretations and paths)
- Third step: equivalence result

$$M, \delta_{GDL}, 0 \models_{GDL} \varphi \iff \delta_{LTL} \models_{LTL} \varphi$$

GDL and Linear Time Logic (LTL)

- Map GDL signature $(N, \mathcal{A}, \Phi)$ and LTL symbols P

$$
\begin{aligned}
P = \Phi \cup \{\mathit{initial}, \mathit{terminal}\} \\
\cup \{\mathit{legal}(r, a) \mid r \in N \text{ and } a \in \mathcal{A}\} \\
\cup \{\mathit{done}(r, a) \mid r \in N \text{ and } a \in \mathcal{A}\} \\
\cup \{\mathit{wins}(r) \mid r \in N\}
\end{aligned}
$$

- Map between GDL and LTL formulas: Function tr

  All symbols are unchanged except $\bigcirc$ and does operators replaced by $\mathbf{X}$ and done operators

$$
tr(\mathit{does}(r, a)) = \mathbf{X}\,\mathit{done}(r, a)
$$

- No more agents and actions!

GDL and Linear Time Logic (LTL)

- Construct an LTL path δ' wrt. an ST model $M = (W, I, T, L, U, g, \pi)$ and path δ st. for any w'_j of δ'
  - $p \in w'_j \iff p \in \pi(\delta[j])$ for any $p \in \Phi$
  - $initial \in w'_j \iff \delta[j] \in I$
  - $terminal \in w'_j \iff \delta[j] \in T$
  - $wins(r) \in w'_j \iff \delta[j] \in g(r)$
  - $legal(r, a) \in w'_j \iff a \in L_r(\delta[j])$
  - $done(r, a) \in w'_j \iff \theta_r(\delta, j-1) = a$ and $j \geqslant 1$
- Interplay

$$M, \delta, j \models_{GDL} \varphi \iff \delta'[j..\infty] \models_{LTL} tr(\varphi)$$

Imperfect Information

Imperfect Information

Example

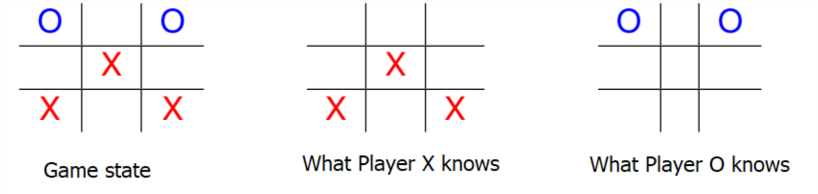


Figure 1: Krieg Tic-Tac-Toe

Each player

- see her own marks only
- know turn-taking and available actions

Main issue

How to describe and reason about games with imperfect information?

GDL-II - Representing Imperfect Information Games

GDL-II: extension of GDL - Server side [Thi10]

- `sees` specify what a player perceives at the next state

  (sees ?player (holds ?player ?card))

  `sees` behaviour similar to `next`: only in head of clauses.

- *random* random player

  (role random)

  Perform action with parameters randomly set

  (does random (deal ?player ?card))

Krieg Tic-Tac-Toe GDL representation (1/3)

Simultaneous move: possible tie-break

```
;;; additional random player for tie break
    (role black)
    (role white)
    (role random)

;;; random player can only solve tie break
    (legal random (tiebreak white))
    (legal random (tiebreak black))

;;; "tried" predicate: "try to mark"
    (<= next (tried ?r ?m ?n)
        (does ?r (mark ?m ?n)))

   (<= next (tried ?r ?m ?n)
       (true (tried ?r ?m ?n)))
```

Krieg Tic-Tac-Toe GDL representation (2/3)

Solving tie-break

```
;;; possible tie-break
    (<= next (cell ?r ?m ?n)
        (true (cell ?m ?n b))
        (does white (mark ?m ?n)))
        (does black (mark ?m ?n)))
        (does random (tiebreak ?r)))
```

Krieg Tic-Tac-Toe GDL representation (3/3)

Only seeing own moves

```
;;; success when moves differ
    (<= sees ?r1 (cell ?m1 ?n1 ?r1)
        (true (cell ?m1 ?n1 b))
        (does ?r1 (mark ?m1 ?n1))
        (does ?r2 (mark ?m2 ?n2))
        (distinct ?m1 ?m2))

 ...
;;; successful tie break
    (<= sees black (cell ?m ?n black)
        (true (cell ?m ?n b))
        (does black (mark ?m ?n)))
        (does random (tiebreak black)))
 ...
```

GDL-II Semantics

Mapping Game G to State-Transition model

- Σ set of all states S of ground atoms f
- $S^{\texttt{true}} = \{\texttt{true}(f_1), \cdots, \texttt{true}(f_n)\}$
  - S: set of ground atoms $f_1 \cdots f_n$
  - $S^{\texttt{true}}$: extension of S with `true` predicate
- $M^{\texttt{does}} = \{\texttt{does}(a_1, ,a_1), \cdots, \texttt{does}(r,a_r)\}$
  - $M^{\texttt{does}}$: joint move derivable from $G \cup S^{\texttt{true}}$
- Model $M = (\Sigma, N, w_0, t, l, u, \mathcal{I}, g)$
  - $N = \{r \mid G \text{ satisfies } \texttt{role}(r)\ \}$
  - $w_0 = \{f \mid G \text{ satisfies } \texttt{init}(f)\ \}$
  - $u(M, S) = \{f \mid G \cup S^{\texttt{true}} \cup M^{\texttt{does}} \text{ satisfies } \texttt{next}(f)\ \}$ for all M and S
  - $\mathcal{I} = \{(r, M, S, p) \mid G \cup S^{\texttt{true}} \cup M^{\texttt{does}} \text{ satisfies } \texttt{sees}(r, p)\ \}$ for all $r \neq \texttt{random}$, M and S

Krieg Tic-Tac-Toe State-Transition model (1/3)

Building up model $M = (N, w_0, t, l, u, \mathcal{I}, g)$

$\{black, white\} \subseteq N$

```
(role black)
(role white)
(role random)
```

$\{cell(1, 1, b), ..., cell(3, 3, b)\} \in w_0$ as

```
(init (cell 1 1 b))
...
(init (cell 3 3 b))
```

Krieg Tic-Tac-Toe State-Transition model (2/3)

Building up model $M = (N, w_0, t, l, u, \mathcal{I}, g)$

$u(\langle (1,1)^x, (3,3)^o \rangle, w_0) = \{cell(1,1,x), ..., \; cell(3,3,o)\}$ as

$G \cup w_0^{\texttt{true}} \cup \langle (1,1)^x, (3,3)^o \rangle^{\texttt{does}}$ satisfies `(next (cell 1 1 x))`

and

$G \cup w_0^{\texttt{true}} \cup \langle (1,1)^x, (3,3)^o \rangle^{\texttt{does}}$ satisfies `(next (cell 3 3 o))`

Remind that

```
(<= next (cell ?r ?m ?n)
        (true (cell ?m ?n b))
        (does white (mark ?m ?n)))
        (does black (mark ?m ?n)))
        (does random (tiebreak ?r)))
```

Krieg Tic-Tac-Toe State-Transition model (3/3)

Building up model $M = (N, w_0, t, l, u, \mathcal{I}, g)$

$(x, \langle(1,1)^x, (3,3)^o\rangle, w_1, \mathit{cell}(1,1,x)) \in \mathcal{I}$ as

$G \cup w_0^{\texttt{true}} \cup \langle(1,1)^x, (3,3)^o\rangle^{\texttt{does}}$ satisfies `(sees x (cell 1 1 x))`

Remind that

```
(<= sees ?r1 (cell ?m1 ?n1 ?r1)
    (true (cell ?m1 ?n1 b))
    (does ?r1 (mark ?m1 ?n1))
    (does ?r2 (mark ?m2 ?n2))
    (distinct ?m1 ?m2))
```

Notice that $(o, \langle(1,1)^x, (3,3)^o\rangle, w_1, \mathit{cell}(1,1,x)) \notin \mathcal{I}$ as

$G \cup w_0^{\texttt{true}} \cup \langle(1,1)^x, (3,3)^o\rangle^{\texttt{does}}$ does not satisfies `(sees o (cell 1 1 x))`

Epistemic extension of GDL logic

Server side vs Player side

- GDL-II: how to represent incomplete or uncertain information
- GDL-II: server perspective (what is the information flow?)
- Player perspective
  - How to handle certain and uncertain information?
  - How to handle other players' "knowledge"?

Epistemic extension: Syntax (1/2)

Extending GDL with epistemic operators [JZPZ16]

- $\mathsf{K}_r\varphi$: "agent r knows φ"
- $\mathsf{C}\varphi$: as "φ is common knowledge among all the agents in N"

Definition (Syntax)

$$\varphi \ ::= \ p \mid initial \mid terminal \mid legal(r,a) \mid wins(r) \mid does(r,a) \mid$$

$$\neg\varphi \mid \varphi \wedge \psi \mid \bigcirc\varphi \mid \mathsf{K}_r\varphi \mid \mathsf{C}\varphi$$

$\mathsf{E}\varphi =_{def} \bigwedge_{r\in N} \mathsf{K}_r\varphi$

Epistemic extension: Syntax (2/2)

Sequential Krieg-Tic-Tac-Toe - Epistemic rules

Additional symbol:

tried$(r, a_{i,j})$ *represents the fact that player r has tried to mark cell (i, j) but failed*

1. $tried(r, a_{i,j}) \to p^{-r}_{i,j}$
2. $does(r, a_{i,j}) \to \mathsf{K}_r(does(r, a_{i,j}))$
3. $initial \to \mathsf{E}initial$
4. $(turn(r) \to \mathsf{E}turn(r)) \wedge (\neg turn(r) \to \mathsf{E}\neg turn(r))$
5. $(p^r_{i,j} \to \mathsf{K}_r p^r_{i,j}) \wedge (\neg p^r_{i,j} \to \mathsf{K}_r \neg p^r_{i,j})$
6. $(tried(r, a_{i,j}) \to \mathsf{K}_r tried(r, a_{i,j})) \wedge (\neg tried(r, a_{i,j}) \to \mathsf{K}_r \neg tried(r, a_{i,j}))$

Epistemic extension: Semantics (1/2)

Epistemic state transition (EST) model M is a tuple

$$(W, I, T, \{R_r\}_{r\in N}, \{L_r\}_{r\in N}, U, g, \pi)$$

- W is a non-empty set of *possible states*.
- $I \subseteq W$, representing a set of *initial* states.
- $T \subseteq W \backslash I$, representing a set of *terminal* states.
- $R_r \subseteq W \times W$ is an equivalence relation for agent r, indicating the states that are indistinguishable for r.
- $L_r \subseteq W \times A^r$ is a *legality* relation for agent r,
- $U : W \times \prod_{r\in N} A^r \hookrightarrow W\backslash I$ is a partial *update* function
- $g : N \to 2^W$ is a *goal* function, specifying the winning states for each agent.
- $\pi : W \to 2^\Phi$ is a standard valuation function.

Epistemic extension: Semantics (2/2)

Imperfect Recall

$$\delta \approx_r \delta' \quad \text{iff} \quad \delta[0]\, R_r\, \delta'[0]$$

Satisfaction with respect to some EST M and path δ

$$\begin{array}{lll} M, \delta \models \mathsf{K}_r\varphi & \text{iff} & \text{for any } \delta' \in \mathcal{P}, \text{ if } \delta \approx_r \delta', \text{ then } M, \delta' \models \varphi \\ M, \delta \models \mathsf{C}\varphi & \text{iff} & \text{for any } \delta' \in \mathcal{P}, \text{ if } \delta \approx_N \delta', \text{ then } M, \delta' \models \varphi \end{array}$$

where $\approx_N$ is the transitive closure of $\bigcup_{r \in N} \approx_r$ and $\mathcal{P}$ is the set of all paths in M.

Epistemic extension: Axiomatics (1/3)

Mainly consists of axiom schemas and inference rules for $\bigcirc, \mathsf{K}_r, \mathsf{C}$ and general game properties [JPZ17]

Axioms

1. All tautologies of classical propositional logic.

Axioms for general game properties

2. $\neg \bigcirc \, \mathit{initial}$
3. $\mathit{terminal} \rightarrow \bigwedge_{a^r \in A^r \setminus \{noop^r\}} \neg \mathit{legal}(a^r) \wedge \mathit{legal}(noop^r)$
4. $\bigvee_{a^r \in A^r} \mathit{does}(a^r)$
5. $\neg(\mathit{does}(a^r) \wedge \mathit{does}(b^r))$ for $a^r \neq b^r$.
6. $\mathit{does}(a^r) \rightarrow \mathit{legal}(a^r)$
7. $\varphi \wedge \mathit{terminal} \rightarrow \bigcirc\varphi$

Epistemic extension: Axiomatics (2/3)

Axioms for $\bigcirc, \mathsf{K}_r, \mathsf{C}$

8. $\bigcirc(\varphi \to \psi) \to (\bigcirc\varphi \to \bigcirc\psi)$
9. $\neg \bigcirc \varphi \leftrightarrow \bigcirc \neg\varphi$
10. $\mathsf{K}_r(\varphi \to \psi) \to (\mathsf{K}_r\varphi \to \mathsf{K}_r\psi)$
11. $\mathsf{K}_r\varphi \to \varphi$
12. $\mathsf{K}_r\varphi \to \mathsf{K}_r\mathsf{K}_r\varphi$
13. $\neg\mathsf{K}_r\varphi \to \mathsf{K}_r\neg\mathsf{K}_r\varphi$
14. $\mathsf{E}\varphi \leftrightarrow \bigwedge_{r=1}^{m} \mathsf{K}_r\varphi$
15. $\mathsf{C}\varphi \to \mathsf{E}(\varphi \wedge \mathsf{C}\varphi)$

Inference Rules

- **(R1)** From φ, $\varphi \to \psi$ infer ψ.
- **(R2)** From φ infer $\bigcirc\varphi$.
- **(R3)** From φ infer $\mathsf{K}_r\varphi$.
- **(R4)** From $\varphi \to \mathsf{E}(\varphi \wedge \psi)$ infer $\varphi \to \mathsf{C}\psi$.

Epistemic extension: Axiomatics (3/3)

Derivation about Krieg-Tic-Tac-Toe (full description: Σ_{KT}).

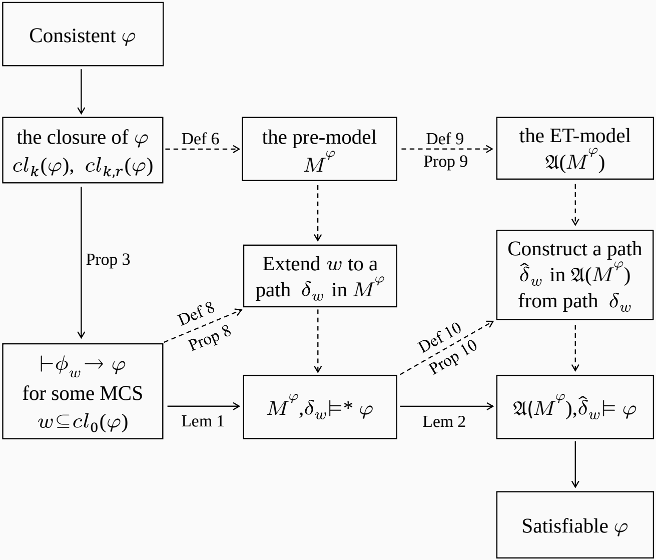
Proposition

For any $r \in N_{KT}$ *and* $a^r_{i,j} \in A^r_{KT}$,

1. $\vdash_{\Sigma_{KT}} initial \to \mathsf{C} initial$
2. $\vdash_{\Sigma_{KT}} legal(a^r_{i,j}) \to \mathsf{K}_r(legal(a^r_{i,j}))$
3. $\vdash_{\Sigma_{KT}} does(a^r_{i,j}) \to \bigcirc \mathsf{K}_r(p^r_{i,j} \vee tried(a^r_{i,j}))$
4. $\vdash_{\Sigma_{KT}} \mathsf{K}_r tried(a^r_{i,j}) \to \mathsf{K}_r p^{-r}_{i,j}$

Completeness... in one slide

Overall picture



EGDL for reasoning about games

General game playing w.r.t. some ongoing game

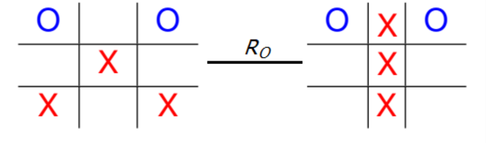


Figure 2: Player o Knowledge

Player o cannot distinguish between the two states

$terminal \rightarrow Cterminal$ is not valid

EGDL for reasoning about games

General game playing w.r.t. some ongoing game

- assessing a “strategy” vs $\langle$game state, move$\rangle$ (M, δ)
- Look ahead via model checking (Δ_2^p)
- Winning situation (encoded in δ)?

$$M, \delta \models K_r \bigcirc \mathit{wins}(x)$$

- Prevent opponent of r to win?

$$\text{Check } M, \delta \models \mathit{does}(r, a) \wedge K_r \bigcirc \neg \mathit{wins}(-r)$$

- Opponent of r may win (wrt. to some r move)?

$$\text{Check } M, \delta \models \neg K_r \neg \bigcirc (\mathit{does}(-r, a) \rightarrow \bigcirc \mathit{wins}(-r))$$

- Still no reasoning over paths in EGDL

Specific game properties

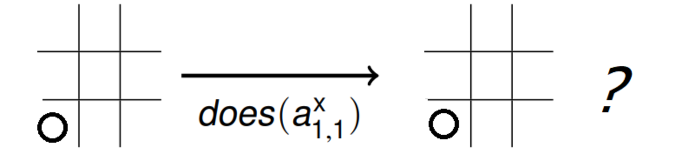


Figure 3: Player x move

Player x knows that

$$does(x, a_{i,j}) \rightarrow \bigcirc K_x(p^x_{i,j} \vee tried(x, a_{i,j}))$$

Hence

$$K_x tried(x, a_{1,1})$$

Specific game properties

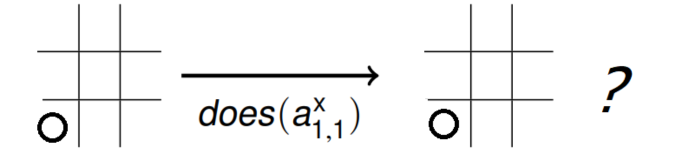


Figure 4: Player x move

Player x knows that

$$K_x tried(x, a_{i,j}) \rightarrow K_x p^o_{i,j}$$

Hence

$$K_x p^o_{1,1}$$

Exercise

Guessing a number

- 2 players game
- Player 1 choose a number $n \in [1, 10]$ (initial state)
- Player 2 has to guess n
- After each round, Player 1 informs Player 2 whether its proposal is too low or too high.
- Player 2 wins if it guesses n in 3 rounds.

Provide the EGDL representation

GDL-III - Representing epistemic games

GDL-III: extension of GDL-II - Server side [Thi17]

- (`knows ?r ?p`) specify that player ?*r* knows information *p*?

  (knows ?player (holds ?player2 ?card))

- (`knows ?p`) specify that ?*p* is common knowledge
- `knows`: introspection operator

  Goals can be epistemic: russian card, muddy children...

- Complex reification mechanism for handling nested knowledge operator

GDL-III Semantics

Knowledge State-Transition System

*fluent state **and** epistemic state*

- Knowledge state: (S, K)
  - S state: defined with respect to `true` predicate (just like for GDL-II)
  - K state: defined with respect to `knows` predicate

$$K = \{(\texttt{knows r f}) \cdots\}$$

- Satisfaction is defined with respect to G, $S^{\texttt{true}}$, K
- Resulting state S' obtained by doing move M

  given by `next` *predicate satisfied by* G, $S^{\texttt{true}}$, K *and* $M^{\texttt{does}}$

$$u(M, S, K) = \{f \mid G \cup S^{\texttt{true}} \cup M^{\texttt{does}} \cup K \text{ satisfies } \texttt{next}(f)\ \}$$

GDL-III Semantics

Knowledge State-Transition System

- Knowledge state: (S, K)
- Resulting state K' obtained by doing move M
  - Need to consider a sequence of moves
  - No move (empty sequence ϵ). Smallest set s.t.

$$K_\epsilon = \{(\texttt{knows}\, r\, f) \mid G \cup w_0^{\texttt{true}} \cup K_\epsilon \text{ satisfies } f\,\}$$

  - Sequence of moves δ and knowledge

$$K_{\delta M} = \{(\texttt{knows}\, r\, f) \mid G \cup s_{\delta' M'}^{\texttt{true}} \cup K_{\delta M} \text{ satisfies } f \quad \text{for all } \delta M \sim_r \delta' M'\,\}$$

- $\delta M \sim_r \delta' M'$: the two sequences cannot be distinguished (same moves and percepts)

Knowledge State-Transition System - a short example

- Muddy children with 2 kids: *alice* and *bob*
- Initialize the foreheads (one of them is muddy)

```
(<= (legal random (muddy ?a ?b ... ?k))
     (true (round 0))
     (yesno ?a) (yesno ?b) ... (yesno ?k)
     (or (yes ?a) ... (yes ?k))))
```

- No initial knowledge (about foreheads)

$$K_\epsilon = K_{01} = K_{10} = K_{11} = \emptyset$$

GDL-III Semantics

Knowledge State-Transition System - a short example

- $K_{01} = K_{10} = K_{11} = \emptyset$
- Only possible move for *alice* and *bob*: $\langle sayNo, sayNo \rangle$

```
(<= (legal ?c (say No)) (not (knows ?c (isMuddy ?c))))

(<= (sees ?c (said ?d ?x)) (does ?d (say ?x)))
```

- After the first (joint) move

$$01.\langle sayNo, sayNo \rangle \sim_{alice} 11.\langle sayNo, sayNo \rangle \sim_{bob} 10.\langle sayNo, sayNo \rangle$$

Knowledge State-Transition System - a short example

- *alice* sees *bob*'s forehead (and vice-versa)

```
(<= (sees ?c (has ?d mud)) (true (has ?d mud)) (distinct ?c ?d))
(<= (isMuddy ?c) (true (has ?c mud)))
```

- Suppose only *bob* is muddy then according to the rules:

```
(<= (sees ?c (said ?d ?x)) (does ?d (say ?x)))
```

bob **knows** as *alice* says No

$K_{01.\langle sayNo, sayNo\rangle} =$

$\{$`(knows alice (isMuddy bob)), (knows bob (isMuddy bob))`$\}$

Imperfect Information: open

Pending questions:

- Benefit of the epistemic dimension?
  - Imperfect information
  - Epistemic game
- Different types of imperfect information?
  - Perfect recall vs Memoryless
  - Nested operator?
- How to implement?

  Computational Complexity - Implementation with an epistemic reasoner?

Reasoning for winning?

From Game Theory to Logic

- Key question in GT: can the player win?
- What is *best response*?
- What about rational behaviour and equilibrium?

van Benthem (2012)

> *Much of game theory is about the question whether strategic equilibria exist. But there are hardly any explicit languages for defining, comparing, or combining strategies.*

Focus on the representation of strategies

Extend GDL and build a player on that extension

- Connecting action and output: how to play?
  - Quantification over possible runs is compulsory

    Overall assessment of the game: what happened if, instead of playing a, b is played?
  - Priority over eligible actions

    if action a leads to win while action b leads to loose, action a should be chosen (if rational)
- Question: how to represent predefined library of strategies?

Focus on the existence of strategies

Combining GDL and Alternating-time Temporal Logic (ATL)

- No previous background expertise

  Assess on the fly, whether it is possible to win?

- Alternative: Using pre-existing logic for reasoning about games (ATL)
- Question: how to represent agent abilities and rationality?

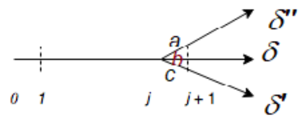
GDL-based Strategy Representation (1/5)

"Priority" operator: $\phi \triangledown \psi$ [JZP14]

ϕ should hold; if not then ψ hold

$$M, \delta, j \models \phi \textit{ or } (\texttt{Paths}(\phi, \delta[0,j]) = \emptyset \textit{ and } M, \delta, j \models \psi)$$

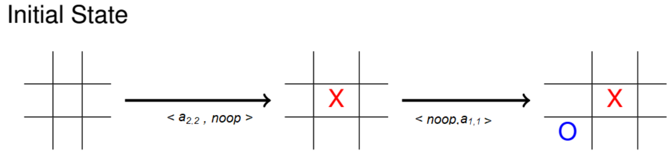
where $\texttt{Paths}(\phi, \delta[0,j])$ is the set of paths where ϕ holds at j and sharing initial segment $\delta[0,j]$:



$\texttt{Paths}(does(r,a), \delta[0,j]) = \{\delta''\}$ and $\texttt{Paths}(does(r,b), \delta[0,j]) = \{\delta\}$

GDL-based Strategy Representation (2/5)

Suppose M and δ:



- $M, \delta, 0 \models does(x, a_{2,2})$
- $M, \delta, 0 \not\models does(x, a_{1,1})$
- $M, \delta, 0 \models does(x, a_{2,2}) \triangledown does(x, a_{1,1})$
- $M, \delta, 1 \models does(o, a_{1,1})$
- $M, \delta, 1 \not\models does(o, a_{2,2})$
- $M, \delta, 1 \models does(o, a_{2,2}) \triangledown does(o, a_{1,1})$

GDL-based Strategy Representation (4/5)

Strategy rule

- syntax: $\phi := \varphi_1 \triangledown \varphi_2 \triangledown \cdots \triangledown \varphi_n$
- Non-ambiguous: at any state, ϕ must "elicit" only one action:
- Could be extended to perfect recall: consider history rather than state.
- Strategy for Player x (1st player)

  $combined^x := fill\_centre^x \triangledown check^x \triangledown block^x \triangledown fill\_corner^x \triangledown fill\_any^x$

  and

  $$\phi^x := (turn(x) \rightarrow combined^x) \wedge (\neg turn(x) \rightarrow noop^x)$$

- Strategy rule ϕ^x is a no loosing strategy for x

  No way to express the output in the GDL with priority

Example: strategy for Tic-Tac-Toe

- Fill the center:

  $$fill\_center^r = does(a^r_{2,2})$$

- Check if I can win:

  $$check^r = \bigvee_{i,j=1}^{3} (does(a^r_{i,j}) \wedge \bigcirc wins(r))$$

- Prevent immediate loss:

  $$block^r = \bigvee_{i,j=1}^{3} (\bigcirc(does(a^{-r}_{i,j}) \wedge \bigcirc wins(\text{-}r)) \wedge does(a^r_{i,j}))$$

- Fill an available corner:

  $$fill\_corner^r = \bigvee_{i,j \in \{1,3\}} does(a^r_{i,j})$$

- Fill anywhere available:

  $$fill\_any^r = \bigvee_{i,j=1}^{3} does(a^r_{i,j})$$

- Combined actions:

  $$combined^r = fill\_centre^r \triangledown check^r \triangledown block^r \triangledown fill\_corner^r \triangledown fill\_any^r$$

GDL-based Strategy Representation

A modal reading of the priority operator (1/2) [ZT15]

- Basic GDL + look ahead operator: $\lfloor a \rfloor \varphi$

  If action a were chosen then φ would be true (but a is not executed)

- *does* operator restricted to joint action: does(a)
- New semantics relative to a state and a joint action: $w, a \models \varphi$
  - $w, a \models p$ iff $p \in \pi(w)$
  - $w, a \models does(b)$ iff $a = b$
  - $w, a \models \lfloor b \rfloor \varphi$ iff $w, b \models \varphi$

A modal reading of the priority operator (2/2)

- Prioritised disjunction operator

$$\varphi \nabla \psi =_{def} \varphi \vee (\psi \wedge \bigwedge_{c} \lfloor c \rfloor \neg\varphi)$$

- In terms of semantics

  For any M, w and a: $w, a \models \varphi \nabla \psi$ iff either $w, a \models \varphi$ or $w, a \models \psi$ but $w, c \models \neg\varphi$ for all c

ATL-Reasoning about strategies

ATL for reasoning about GDL game description

- Use GDL game description as underlying semantic for ATL reasoning
- ATL: reasoning about cooperation

$$\langle\langle C\rangle\rangle\varphi \text{ Coalition } C \text{ can achieve } \varphi$$

- GDL + ATL:
  - check properties of game (playability)
  - check strategic properties

ATL Reasoning about strategies

Alternating-time Temporal Logic - Syntax [AHK02, RvdHW09]

- Coalition operator $\langle\langle C \rangle\rangle$
- Temporal operator $\bigcirc$ (next), $\Box$ (always), $\Diamond$ (sometimes), $\mathcal{U}$ (until)

$$\varphi \ ::= \ p \mid \varphi \vee \varphi \mid \langle\langle C \rangle\rangle \bigcirc \varphi \mid \langle\langle C \rangle\rangle \Box \varphi \mid \langle\langle C \rangle\rangle \Diamond \varphi \mid \langle\langle C \rangle\rangle \varphi \mathcal{U} \varphi$$

- coalition and temporal operators always together

$$\langle\langle x \rangle\rangle \Diamond wins(x) \vee \langle\langle x \rangle\rangle \Diamond \neg wins(-x)$$

Alternating-time Temporal Logic - Semantics

- based on Concurrent Game Structure (or Transition systems)

$$\mathcal{A} = (\mathcal{Q}, q_0, N, \Pi, \pi, \mathit{legal}, \mathit{update})$$

  where

  - $\mathcal{Q}$: set of states
  - q_0: initial state
  - N: set of agents
  - Π: propositions
  - π: valuation function
  - *legal*: possible move function for each agent
  - *update*: deterministic joint move transition function

- Truth condition relative to a state q

$$\mathcal{A}, q \models_{ATL} \varphi$$

ATL Reasoning about strategies

Alternating-time Temporal Logic - Semantics

- λ: sequence of states
- Additional component: strategy function $f_a(\lambda) \in \mathit{legal}(a, q)$ where q is the last state of λ

$$F_A = \{f_a | a \in A\}$$

- Output of a strategy: set of possible sequences $\lambda = q\, q'\, q'' \ldots$

$$\mathit{out}(q, F_A) = \{\lambda | \lambda[0] = q \text{ and}$$
$$\exists m \text{ s.t. } \forall a \in A, m_a \in f_a(\lambda[0..i]) \text{ and } (\lambda[i{+}1] = \mathit{update}(\lambda[i], m)\}$$

ATL Reasoning about strategies

Alternating-time Temporal Logic - Semantics

- $\mathcal{A} = (\mathcal{Q}, q_0, N, \Pi, \pi, legal, update)$
- Truth conditions
  - $\mathcal{A}, q \models_{ATL} p$ iff $p \in \pi(q)$
  - $\mathcal{A}, q \models_{ATL} \langle\langle C \rangle\rangle \bigcirc \varphi$ iff there exists F_C such that:

    $$\mathcal{A}, \lambda[1] \models_{ATL} \varphi \text{ for all } \lambda \in out(q, F_C)$$

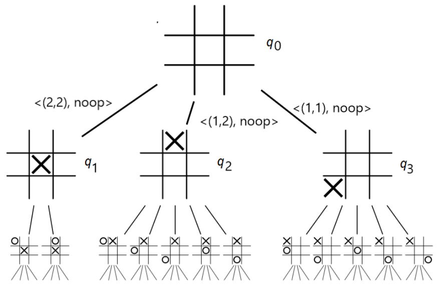
  - $\mathcal{A}, q \models_{ATL} \langle\langle C \rangle\rangle \Box \varphi$ iff there exists F_C such that:

    $$\mathcal{A}, \lambda[i] \models_{ATL} \varphi \text{ for all } \lambda \in out(q, F_C) \text{ and } i \geqslant 0$$

ATL Reasoning about strategies

$\mathcal{A} = (\mathcal{Q}, q_0, N, \Pi, \pi, \mathit{legal}, \mathit{update})$

- Assume
  $f_x([q_0 q_1]) = \mathit{noop}$
- $\mathcal{A}, q_1 \models_{ATL}$
  $\langle\langle x \rangle\rangle \Box (p^x_{1,1} \vee p^x_{3,3})$
- Assume
  $f_o([q_0 q_2]) = \{(2,2)\}$
- $\mathcal{A}, q_2 \models_{ATL}$
  $\langle\langle o \rangle\rangle \bigcirc p^o_{2,2}$



ATL Reasoning about strategies

ATL for checking GDL specification

- Translation/embedding of GDL theory to ATL
- Model checking is EXPTIME
- Checking soundness

$$\langle\langle\rangle\rangle\Box((\mathit{terminal} \wedge \varphi) \rightarrow \langle\langle\rangle\rangle\Box(\mathit{terminal} \wedge \varphi))$$

- Winnable

$$\bigvee_i \langle\langle i\rangle\rangle\Diamond \mathit{wins}(i)$$

- Sequential

$$\langle\langle\rangle\rangle\Box(\langle\langle N\rangle\rangle \bigcirc \varphi \rightarrow \bigvee_i \langle\langle i\rangle\rangle \bigcirc \varphi)$$

ATL for checking GDL specification

- Tic-Tac-Toe properties (CGS encoding)
- no-losing strategies for x

$$\langle\langle x \rangle\rangle \Box (\mathit{terminal} \rightarrow \neg \mathit{wins}(o))$$

- No explicit representation of actions (hidden in the semantics)

GDL-based Strategy Representation - Going further

Pending questions:

- How to design strategies?

  Connection with Machine Learning and Planning

- Generalize strategies?

  Are they any common points (General Strategic Reasoning)

- How to implement?

  Complexity of strategic reasoning and complexity of the game

Still a lot to do! - Example: Equivalent games

Equivalent games (1/3)

Number Scrabble:

1. $initial \leftrightarrow turn(\mathrm{b}) \wedge \neg turn(\mathrm{w}) \wedge \bigwedge_{i=1}^{9} \neg(s(\mathrm{b}, i) \vee s(\mathrm{w}, i))$
2. $wins(r) \leftrightarrow \big(\bigvee_{i=2}^{3}(s(r,i) \wedge s(r,4) \wedge s(r,11-i)) \vee$
   $\bigvee_{i=1}^{2}(s(r,i) \wedge s(r,6) \wedge s(r,9-i)) \vee$
   $\bigvee_{i=1}^{4}(s(r,5-i) \wedge s(r,5) \wedge s(r,5+i))\big)$
3. $terminal \leftrightarrow wins(\mathrm{b}) \vee wins(\mathrm{w}) \vee \bigwedge_{i=1}^{9}(s(\mathrm{b}, i) \vee s(\mathrm{w}, i))$
4. $legal(r, pick(n)) \leftrightarrow \neg(s(\mathrm{b}, n) \vee s(\mathrm{w}, n)) \wedge turn(r) \wedge \neg terminal$
5. $legal(r, noop) \leftrightarrow turn(-r) \vee terminal$
6. $\bigcirc s(r,n) \leftrightarrow s(r,n) \vee (\neg(s(\mathrm{b}, n) \vee s(\mathrm{w}, n)) \wedge does(r, pick(n)))$
7. $turn(r) \wedge \neg terminal \rightarrow \bigcirc \neg turn(r) \wedge \bigcirc turn(-r)$

Equivalent games (2/3)

Equivalence

Semantics 2 models (State-Transition) with a bisimulation between them

Syntax Set of rules are equivalent

Number Scrabble and Tic-Tac-Toe are equivalent

Equivalent Games (3/3)

Pending questions:

- Loose equivalence

  A game is "close " to a second one? Restricted equivalence to a sub-part of the game?

- Connecting equivalence and strategic reasoning

  "ready-to-go" strategies

- How to implement

  Complexity for deciding whether two games are equivalent. Available heuristics?

Perspectives

A lot of questions!

On GDL:

- Connecting action and strategy
- Imperfect Information
- Games comparison

Still on GDL

- Connection to planning
- Construction of a General Player?

    Is it realistic to reason with GDL formulas?

Main references i

Rajeev Alur, Thomas A Henzinger, and Orna Kupferman.
Alternating-time temporal logic.
Journal of the ACM, 49(5):672–713, 2002.

Michael Genesereth and Michael Thielscher.
***General game playing*, volume 8 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*.**
Morgan & Claypool Publishers, 2014.

Guifei Jiang.
***Logics for strategic reasoning and collective decision-making*.**
PhD thesis, Western Sydney & University Université Toulouse 1, 2016.

Main references ii

Guifei Jiang, Laurent Perrussel, and Dongmo Zhang.
On axiomatization of epistemic GDL.
In *LORI*, volume 10455 of *Lecture Notes in Computer Science*, pages 598–613. Springer, 2017.

Guifei Jiang, Dongmo Zhang, and Laurent Perrussel.
GDL meets ATL: A logic for game description and strategic reasoning.
In *Proceedings of the 13th Pacific Rim International Conference on Artificial Intelligence (PRICAI'14)*, pages 733–746. Springer, 2014.

Guifei Jiang, Dongmo Zhang, Laurent Perrussel, and Heng Zhang.
Epistemic GDL: A logic for representing and reasoning about imperfect information games.
In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, pages 1138–1144, 2016.

Ji Ruan, Wiebe van der Hoek, and Michael Wooldridge.
Verification of games in the game description language.
J. Log. Comput., 19(6):1127–1156, 2009.

Main references iv

Michael Thielscher.
A general game description language for incomplete information games.
In *AAAI*. AAAI Press, 2010.

Michael Thielscher.
GDL-III: A description language for epistemic general game playing.
In *IJCAI*, pages 1276–1282. ijcai.org, 2017.

Johan van Benthem.
In praise of strategies.
In Jan van Eijck and Rineke Verbrugge, editors, *Games, Actions and Social Software: Multidisciplinary Aspects*, pages 96–116. Springer Berlin Heidelberg, 2012.

Dongmo Zhang and Michael Thielscher.
A logic for reasoning about game strategies.
In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*, pages 1671–1677, 2015.

Want to work on it

PhD position: General Auction Player

Starting date: Feb 19 (negotiable) - 3 years.

Keywords: reasoning about auctions, GDL, strategic reasoning, Auction mechanism.

Post-Graduate - Master thesis: Strategic player

Starting date: Feb 19 - 6 months - pre-PhD.

Keywords: strategic reasoning, GDL, ATL, Model-checking based player.

More details: `laurent.perrussel@irit.fr`