



# General Game Playing

HANNES BERGENFALK

# Why

- ▶ Dream of AI research is strong general intelligence
- ▶ Systems like Deep Blue and Alpha go are amazing but they can only do one thing.
- ▶ General Game Playing systems will still not be general intelligence but it will be one step closer
- ▶ Also games are fun...

# How

- ▶ Game description language created by Michael Genesereth, Stanford University
- ▶ Describes games that are: finite, discrete, deterministic and has complete information. Moves are made simultaneously
- ▶ Competition held by AAI since 2005

# My project

- ▶ Create a program that can parse game rules in the GDL format and reason about those rules using logic
- ▶ If time allows create an AI that can play the games reasonably well
- ▶ Possibly connect it to a general game playing server

# An Overview of GDL

- ▶ A list of implications with certain special keywords
- ▶  $h \leq b_1 \wedge b_2 \wedge \dots \wedge b_n$
- ▶ KIF syntax, prefix notation

▶ Example:

```
(<= (next (cell ?m ?n x)) (does xplayer (mark ?m ?n)) (true (cell ?m ?n b)))
```

# Tic Tac Toe in GDL

```
(role xplayer)
(role oplayer)

(init (cell 1 1 b))
(init (cell 1 2 b))
(init (cell 1 3 b))
(init (cell 2 1 b))
(init (cell 2 2 b))
(init (cell 2 3 b))
(init (cell 3 1 b))
(init (cell 3 2 b))
(init (cell 3 3 b))
(init (control xplayer))

(<= (goal xplayer 100) (line x))
(<= (goal xplayer 50) (not (line x))(not (line o)) (not open))
(<= (goal xplayer 0) (line o))
(<= (goal oplayer 100) (line o))
(<= (goal oplayer 50) (not (line x))(not (line o)) (not open))
(<= (goal oplayer 0) (line x))

(<= terminal (line x))
(<= terminal (line o))
(<= terminal (not open))

(<= (next (cell ?m ?n x)) (does xplayer (mark ?m ?n)) (true (cell ?m ?n b)))
(<= (next (cell ?m ?n o)) (does oplayer (mark ?m ?n)) (true (cell ?m ?n b)))
(<= (next (cell ?m ?n ?w)) (true (cell ?m ?n ?w)) (distinct ?w b))
(<= (next (cell ?m ?n b)) (does ?w (mark ?j ?k)) (true (cell ?m ?n b)) (or (distinct ?m ?j) (distinct ?n ?k)))
(<= (next (control xplayer)) (true (control oplayer)))
(<= (next (control oplayer)) (true (control xplayer)))

(<= (row ?m ?x) (true (cell ?m 1 ?x))(true (cell ?m 2 ?x)) (true (cell ?m 3 ?x)))
(<= (column ?n ?x) (true (cell 1 ?n ?x))(true (cell 2 ?n ?x))(true (cell 3 ?n ?x)))
(<= (diagonal ?x) (true (cell 1 1 ?x))(true (cell 2 2 ?x))(true (cell 3 3 ?x)))
(<= (diagonal ?x) (true (cell 1 3 ?x))(true (cell 2 2 ?x))(true (cell 3 1 ?x)))

(<= (line ?x) (row ?m ?x))
(<= (line ?x) (column ?m ?x))
(<= (line ?x) (diagonal ?x))

(<= open (true (cell ?m ?n b)))

(<= (legal ?w (mark ?x ?y)) (true (cell ?x ?y b))(true (control ?w)))
(<= (legal xplayer noop) (true (control oplayer)))
(<= (legal oplayer noop) (true (control xplayer)))
```

# Method

- ▶ Parse the input file yielding token trees
- ▶ Create specialized data structure representing expressions and implications
- ▶ Implement substitution and unification of variables in said data structure
- ▶ Putting it together into a representation of a game, that can be queried for legal moves, have its state updated, etc...
- ▶ Make use of the GDL keywords

Demo



# Results

- ▶ I can parse GDL and represent a game
- ▶ I can play tic tac toe and games of similar complexity
- ▶ For larger games each turn takes several minutes to process
- ▶ The "AI" in the demo chooses a move completely randomly
- ▶ No server communication
- ▶ One problem I encountered: forward chaining versus backward chaining

# Forward vs backward chaining

Example:  $h_1 \Leftarrow b_1 \wedge b_2 \wedge \dots \wedge b_n$   
 $h_2 \Leftarrow b'_1 \wedge b'_2 \wedge \dots \wedge b'_m$

$$h_2 == b_2$$

- ▶ Backward chaining: evaluate  $h_2$  when it is asked for in the evaluation of  $h_1$
- ▶ Forward chaining: evaluate  $h_2$  beforehand so  $b_2$  is known when  $h_1$  is evaluated
- ▶ I used forward chaining, this was a mistake
- ▶ I am implementing backward chaining, but it does fit neatly into my implementation of substitution
- ▶ Mostly an issue of my time

# Future work

- ▶ Finish implementing backward chaining
- ▶ Make representation and reasoning more efficient in general
- ▶ Implement "better" AI decision making
- ▶ For example: Monte Carlo tree search
- ▶ Implement Server communication

Questions?