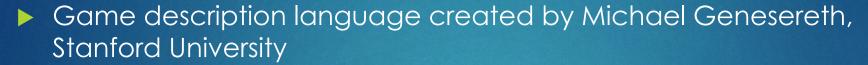
General Game Playing HANNES BERGENFALK

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



- Describes games that are: finite, discrete, deterministic and has complete information. Moves are made simultaneously
- Competition held by AAAI 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

- $\blacktriangleright h \le b_1 \land b_2 \land \dots \land b_n$
- KIF syntax, prefix notation

Example:

(<= (next (cell ?m ?n x)) (does xplayer (mark ?m ?n)) (true (cell ?m ?n b)))</pre>

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 22b))(init (cell 23b))(init (cell 3 1 b)) (init (cell 32b))(init (cell 3 3 b))

- (init (control xplayer))
- $(\leq (\text{goal xplayer 100})(\text{line x}))$
- (<= (goal xplayer 50)(not (line x))(not (line o)) (not open))</pre>
- (<= (goal xplayer 0) (line o))
- $(\leq (\text{goal oplayer 100}) (\text{line o}))$
- $(\leq (\text{goal oplayer 50}) (\text{not (line x}))(\text{not (line o})) (\text{not open}))$
- $(\leq (aoal oplayer 0)(line x))$
- (<= terminal (line x))
- (<= terminal (line o))
- (<= terminal (not open))

- - (<= (legal ?w (mark ?x ?y))(true (cell ?x ?y b))(true (control ?w)))</pre>
 - (<= (legal xplayer noop)(true (control oplayer)))
 - (<= (legal oplayer noop)(true (control xplayer)))

- (<= (next (cell ?m ?n x)) (does xplayer (mark ?m ?n)) (true (cell ?m ?n b)))</pre>
- (<= (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)))</pre>
- (<= (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)))

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



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 "Al" 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:

 $h_1 \le b_1 \land b_2 \land \dots \land b_n$ $h_2 \le b'_1 \land b'_2 \land \dots \land b'_m$

 $h_2 == b_2$

- Backward chaining: evaluate h2 when it is asked for in the evaluation of h1
- Forward chaining: evaluate h2 beforehand so b2 is known when h1 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

