Knowledge Representation for Systems with General Intelligence

Michael Thielscher, IJCAI 2021, Montreal
Artificial General Intelligence

Intelligence is the ability to

• Perceive or infer information, and to
• Retain it as knowledge to be applied towards adaptive behaviours within an environment or context

Artificial general intelligence enables systems to

• Perceive or infer information about new tasks
• Adapt to these tasks without human intervention

Wikipedia
Systems with General Intelligence

A system with general intelligence can

• Take representations of new problems as input
• Solve these problems without human intervention

Disclaimer
This isn’t just Newell and Simon’s General Problem Solver:
• They only used symbols to represent problems
• They only used symbol manipulation to solve them
A system with general intelligence can
• Take representations of new problems as input
• Solve these problems without human intervention

Claim
Systems with general intelligence require to integrate different AI methods
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

A general game playing system can

• Understand descriptions of new games
• Learn to play these new games effectively without human intervention
Define:

1. A **problem class** $C$
2. How to **present** problems $p \in C$ to a system

**General Game Playing (2005, 1\textsuperscript{st} AAAI Competition)**

Finite, $n$-player games with
- Synchronous moves
- Complete state information
Defining General Problem-Solving Tasks

Define:

1. A problem class $C$
2. How to present problems $p \in C$ to a system
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

Knowledge Representation Formalisms for Dynamic Domains

J. McCarthy’s Situation Calculus (1968)
STRIPS (1972)
↩ PDDL (1998)
Game Description Language GDL (2005)
Video Game Description Language (2014)
Ludii (2016)
Game Descriptions

$$(\text{init} \ (\text{cell} \ a \ 1 \ \text{white-rook}))$$

$$\ldots$$

$$(\text{init} \ (\text{cell} \ h \ 8 \ \text{black-rook}))$$

$$(\leq\ (\text{legal} \ \text{white} \ (\text{castle} \ ?\text{direction})))$$

$$\quad (\text{true} \ (\text{control} \ \text{white}))$$

$$\quad (\text{true} \ (\text{cell} \ e \ 1 \ \text{white-king}))$$

$$\ldots$$

$$(\leq\ (\text{next} \ (\text{cell} \ c \ 1 \ \text{white-king})))$$

$$\quad (\text{does} \ \text{white} \ (\text{castle} \ \text{c-side})))$$

$$\leq\ (\text{goal} \ (?\text{player} \ 100))$$

$$\quad (\text{checkmate})$$

$$\quad (\text{not} \ (\text{true} \ (\text{control} \ ?\text{player})))$$
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

Inferring New Knowledge

Answer Set Program

Solver for single-player games

M.T., ICLP’09

(init (cell a 1 white-rook))
...
(<= (legal (white castle ?direction))
    ... )
...
(<= (goal (?player 100))
    (checkmate)
    (not (true (control ?player))))
Inferring New Knowledge

Answer Set Program
Solver for single-player games

Stochastic Constraint Program
General search-based player

Koriche, Lagrue, Piette, & Tabary, *Constraints* 2016
Inferring New Knowledge

(init (cell a 1 white-rook))
  ...
(<= (legal (white castle ?direction))
    ... )
  ...
(<= (goal (?player 100))
    (checkmate)
    (not (true (control ?player))))

Answer Set Program
Solver for single-player games

Stochastic Constraint Program
General search-based player

Quantified Boolean Formulas
Solver for two-player games
Learning to Play New Games

```
(init (cell a 1 white-rook))
...
(<= (legal (white castle ?direction))
    ...)
...
(<= (goal (?player 100))
    (checkmate)
    (not (true (control ?player))))
```

Propositional Network

Schkufza, Love, & Genesereth, AI’08
Learning to Play New Games

```
(init (cell a 1 white-rook))
...
(<= (legal (white castle ?direction))
   ...
   ...
(<= (goal (?player 100))
   (checkmate)
   (not (true (control ?player))))
```

Propositional Network

Neural Network

Goldwaser & M.T., AAAI’20

Feature extraction
The AlphaZero Method

- **Self-play** using Monte Carlo Tree Search (MCTS) at each state, guided by Neural Network

- **Train** Neural Network with self-play result

Silver et al., Science 2018
Generalising AlphaZero to Full GDL

**AlphaZero**
- Two-player, zero-sum, symmetric, turn-based, board games
- Hand-crafted network architecture for each game

**GGPZero**
- Arbitrary GDL games
- Non-zero-sum games: $E(\text{Reward})$
- Asymmetric multiplayer games: $n$ output layers
Can we learn from General Game-Playing Systems?

- “Maybe our conception of chess has been too limited.”
  (Demis Hassabis)
- “Deeper understanding is very difficult to make explicit.”
  (Judit Polgár)

GGP: KR + Inference + Search + Learning

- Knowledge representation to describe new games
- Reasoning to understand the rules
- Search enhanced by Learning to play well
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

General Problem-Solving Robot (2016)

Goals

- High-level planning, execution monitoring
- Parallel actions, execution constraints
- Handle human interference

General blocksworld problem-solving robot

2x speed
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

Control Hierarchy

Action update

Right-arm Control

Symbolic Control

Physics Simulator
Gazebo

Left-arm Control

Clark, Hengst, …, & M.T., IJCAI'16
Control Hierarchy

Sensing update

Symbolic Control

Physics Simulator Gazebo

Right-arm Control

Left-arm Control

Clark, Hengst, …, & M.T., IJCAI’16
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

Describing Epistemic Planning Problems

- Robot $R$ grabs a block (b) to check if it’s labelled “A”
- Robot $S$ may observe this

\[
\begin{align*}
\text{pre}: & \quad \text{label}(b,A) \quad \text{pre}: \quad \text{not label}(b,A) \\
\text{eff}: & \quad \text{has}(R,b) \quad \text{eff}: \quad \text{has}(R,b)
\end{align*}
\]
Describing Epistemic Planning Problems

- Robot R grabs a block (b) to check if it’s labelled “A”
- Robot S may observe this

\[
\begin{align*}
\text{pre:} & \quad \text{label}(b,A) \\
\text{pre:} & \quad \text{not label}(b,A) \\
\text{effect:} & \quad \text{has}(R,b) \\
\text{effect:} & \quad \text{has}(R,b)
\end{align*}
\]
Describing Epistemic Planning Problems

- Robot \( R \) grabs a block (b) to check if it’s labelled “A”
- Robot \( S \) may observe this

\[
\begin{align*}
\text{pre: } & \text{ label}(b,A) \\
\text{eff: } & \text{ has}(R,b) \\
\text{pre: } & \text{ not label}(b,A) \\
\text{eff: } & \text{ has}(R,b)
\end{align*}
\]
Describing Epistemic Planning Problems

- Robot $R$ grabs a block (b) to check if it’s labelled “A”
- Robot $S$ may observe this

\[
\text{pre: } \text{label}(b, A) \\
\text{looking}(S) \\
\text{eff: } \text{has}(R, b) \\
\]

\[
\text{pre: } \text{not label}(b, A) \\
\text{looking}(S) \\
\text{eff: } \text{has}(R, b) \\
\]

\[
\text{pre: } \text{label}(b, A) \\
\text{not looking}(S) \\
\text{eff: } \text{has}(R, b) \\
\]

\[
\text{pre: } \text{not label}(b, A) \\
\text{not looking}(S) \\
\text{eff: } \text{has}(R, b) \\
\]
A New Event Model Description Language

<table>
<thead>
<tr>
<th>Event</th>
<th>Causes</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grab(x,b)</td>
<td>causes has(x,b)</td>
<td></td>
</tr>
<tr>
<td>Grab(x,b)</td>
<td>causes obs(x,label(A)) if label(b,A)</td>
<td></td>
</tr>
<tr>
<td>Grab(x,b)</td>
<td>causes obs(y,grab(x,b)) if looking(y)</td>
<td></td>
</tr>
<tr>
<td>Signal(x,y)</td>
<td>causes looking(y)</td>
<td></td>
</tr>
<tr>
<td>Distract(x,y)</td>
<td>causes not looking(y)</td>
<td></td>
</tr>
</tbody>
</table>

It follows that, after Signal(R,S), Distract(R,T), Grab(R,b),

- R knows whether block b is labelled “A”
- S knows that R knows whether block b is labelled “A”
- T doesn’t know that S knows that R knows …
A New Event Model Description Language

<table>
<thead>
<tr>
<th>Event</th>
<th>Cause 1</th>
<th>Cause 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grab(x,b)</td>
<td>causes has(x,b)</td>
<td></td>
</tr>
<tr>
<td>Grab(x,b)</td>
<td>causes obs(x,label(A)) if label(b,A)</td>
<td></td>
</tr>
<tr>
<td>Grab(x,b)</td>
<td>causes obs(y,grab(x,b)) if looking(y)</td>
<td></td>
</tr>
<tr>
<td>Signal(x,y)</td>
<td>causes looking(y)</td>
<td></td>
</tr>
<tr>
<td>Distract(x,y)</td>
<td>causes not looking(y)</td>
<td></td>
</tr>
</tbody>
</table>

Theorem. Rajaratnam & M.T., KR’21

- Every event model can be described in the language (DER)
- The DER description can be exponentially smaller
- The canonical DER description is always polynomial in size
General Epistemic BW Problem-Solving Robot (Work in Progress)

Goals

• High-level epistemic planning, execution monitoring
• Parallel actions, execution constraints
• Handle human/robot collaboration
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

Hierarchical Control for Virtual Agents

Animated characters
- Act and react to human gestures and movements

Hierarchical control combines
- High-level BDI-based agent program (Jason)
- Low-level control engine for characters (Unity)
Hierarchical Control for Virtual Agents

Intraspace
UNSW / Academy of Fine Arts Vienna,
Christina Jauernek
Outline

1. General game-playing programs
   • KR for describing new games
   • + Inference + Search + Deep Learning

2. General collaborative problem-solving robots
   • Hybrid robot control
   • + Epistemic planning for collaboration

3. Agent programming for interactive artworks

The Escape Room Robot Challenge

Goal
• Test the ability of robots to solve new problems in new environments

Tasks
• Scan and map room
• Identify physical objects
• Understand and solve symbolic puzzles
• Relate solutions to objects and actions
Summary: KR to Leverage AI Systems

Goal
• Leverage our AI systems by enhancing their ability to understand and adapt to new problems

KR helps systems to
• Understand and reason about problem descriptions

Examples
• KR + Search + Deep RL to build a general game player
• KR + Hybrid control architectures for robots, virtual agents
Thanks

Dennis Del Favero
Adrian Goldwaser
Som Guan
Bernhard Hengst
Maurice Pagnucco
David Rajaratnam
Ji Ruan
Claude Sammut

Keith Clark
Thorsten Engesser
Christian Freude
Alvaro Gunawan
Christina Jauernik
Robert Mattmüller
Ajit Narayanan
Bernhard Nebel
Peter Robinson
Wolfgang Tschapeller