Systems with General Intelligence

—

A New Perspective

Michael Thielscher

Outline

PART I

- A Grand AI Challenge
  General game playing
- Defining your own Grand AI Challenge
  Systems with general intelligence

PART II

- A new research agenda
  Combining representations, methods, systems

How Intelligent are AI Systems?

AI systems are able to

- make autonomous decisions
- adapt flexibly to unforeseen situations

Do they, really?

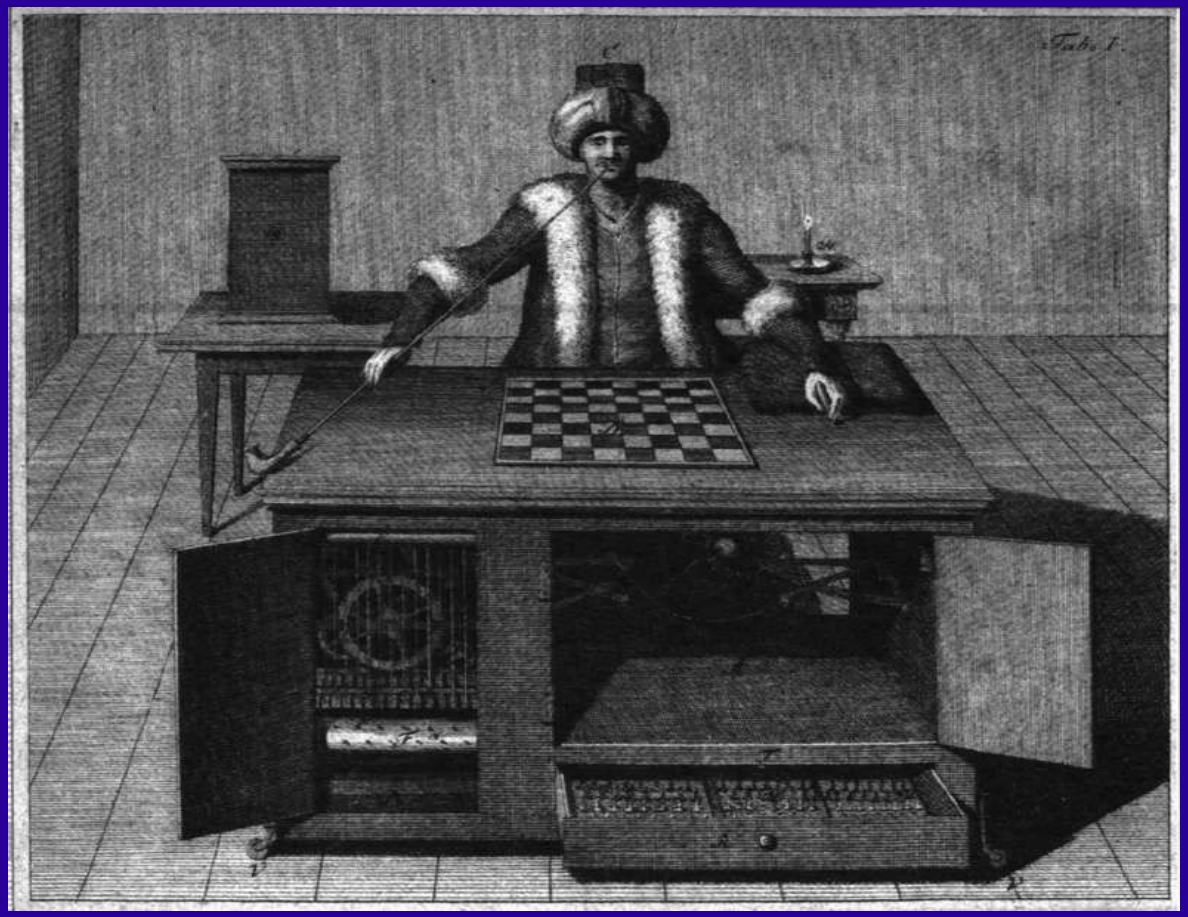
Most existing AI systems are

- designed for a specific and narrow application
- use tailor-made algorithms

The intelligence lies with the programmers—not their systems

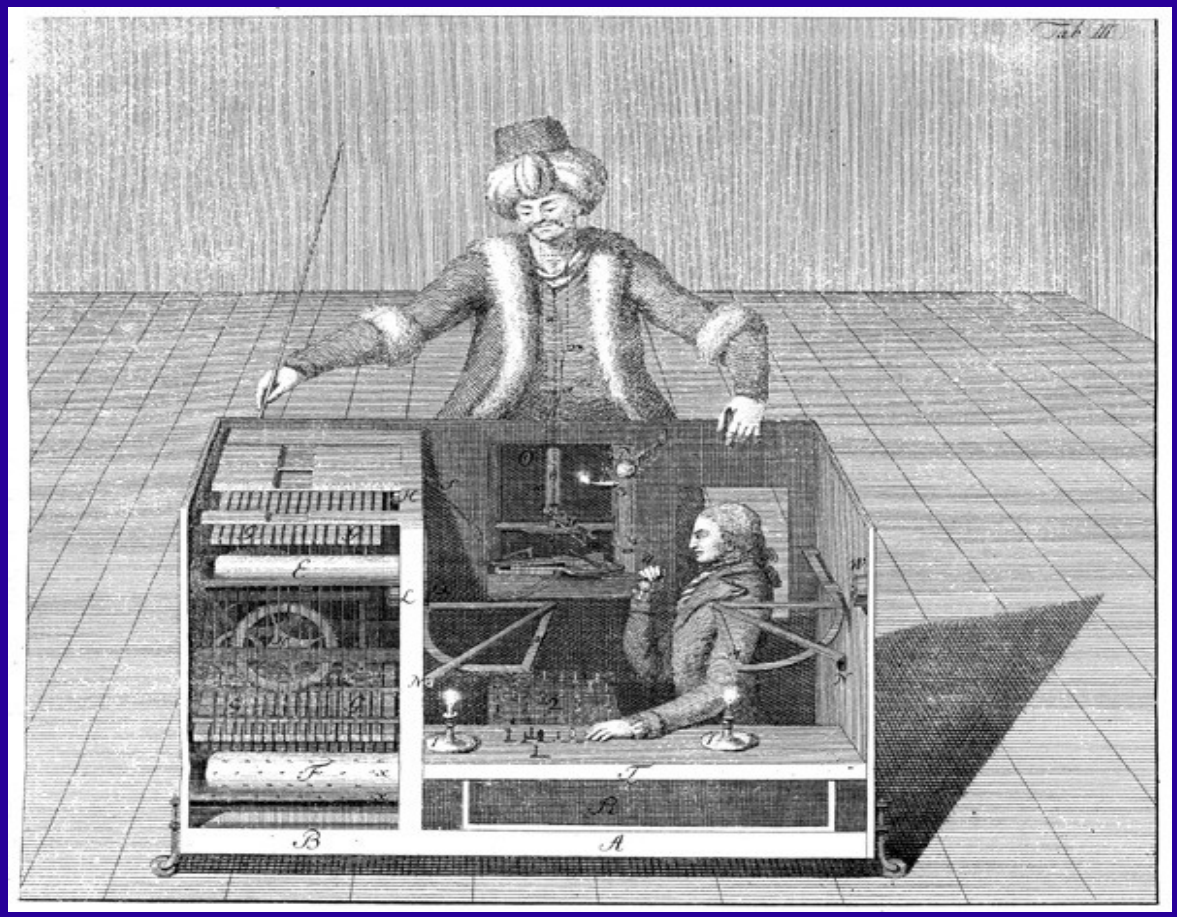
Example: Chess Computers

In the early days, chess playing was considered a key to AI



Turk
(Vienna 1770)

Example: Chess Computers



Secret revealed
(1857)

Example: Chess Computers

Chess computers reach human level

Deep Blue
(New York 1997)

Example: Chess Computers

Deep Blue was a success story. But also a major leap for AI?

No:

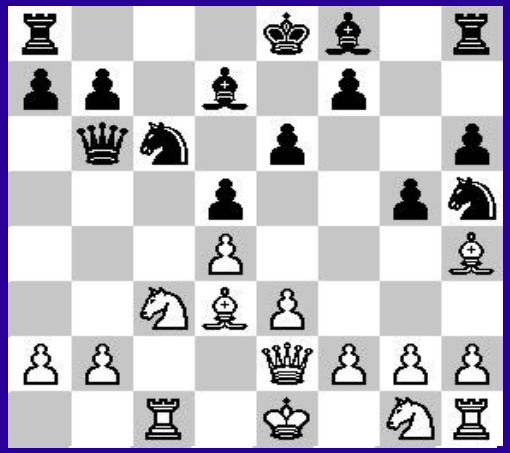
- Chess computers are highly specialised systems
- Deep Blue can't handle anything outside its 64-square world

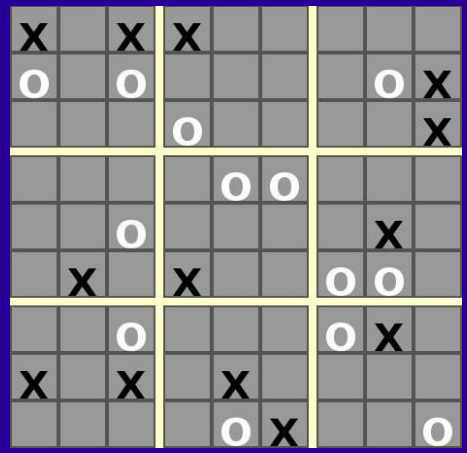
Deep Blue's capabilities were just not general enough

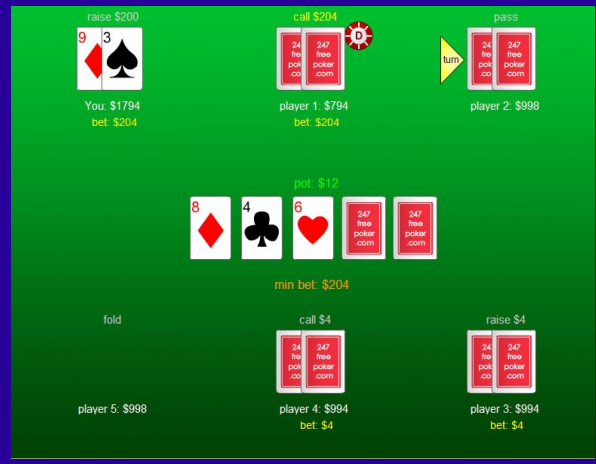
A Grand AI Challenge: General Game Playing

A General Game Player is a system that

- understands description of arbitrary games
- learns to play these games without human intervention

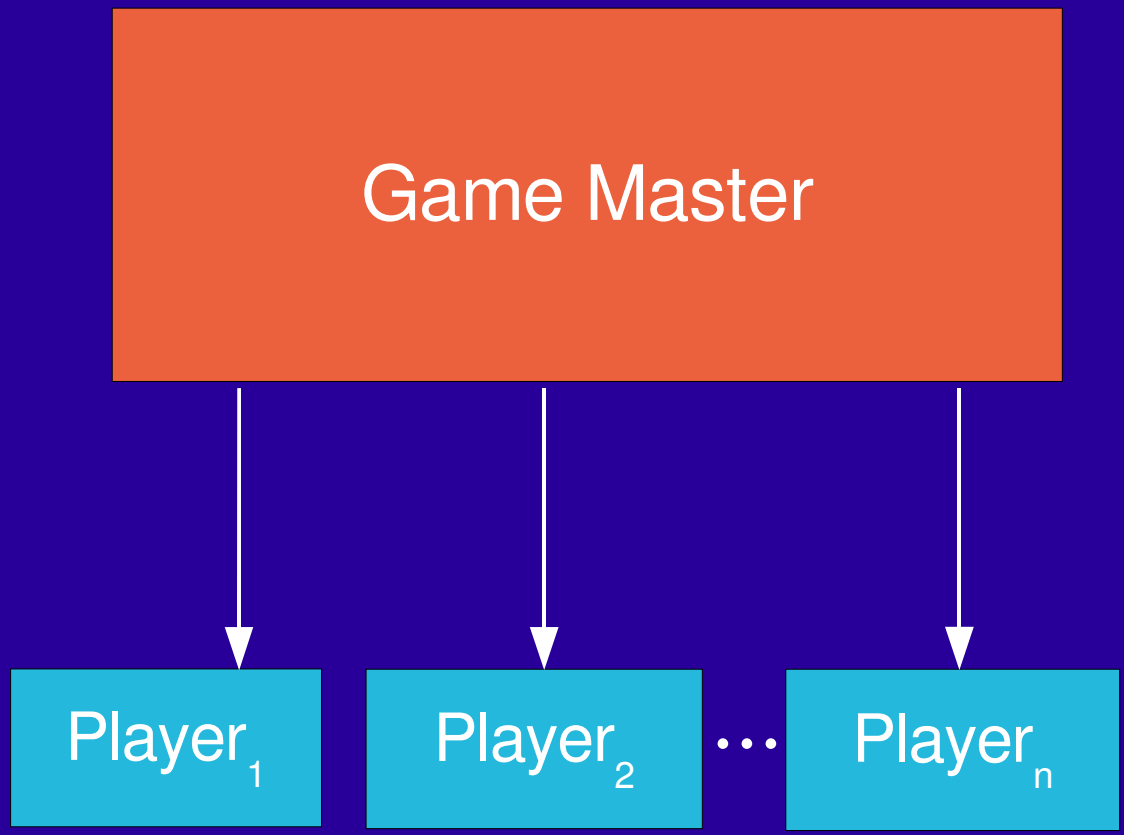






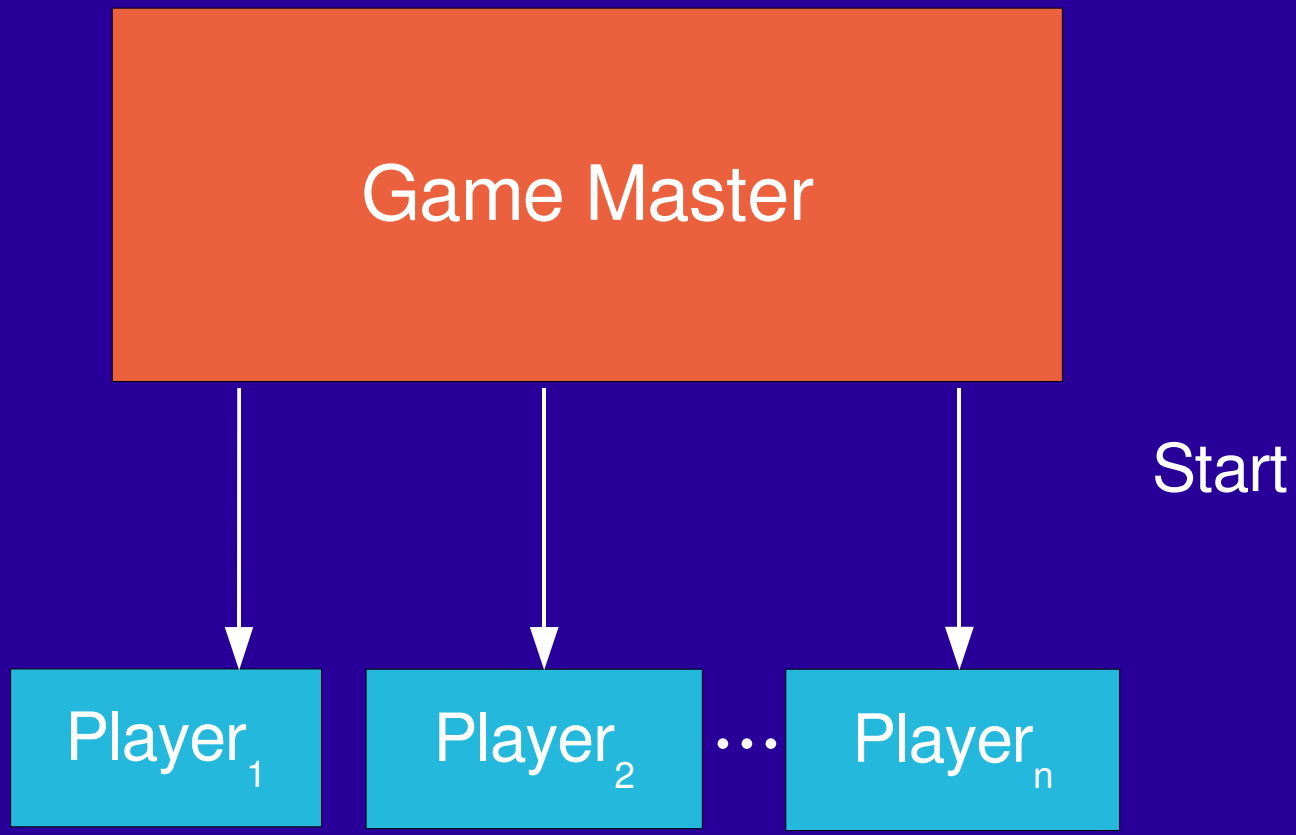
General Game Playing Contest @AAAI since 2005

How it Works

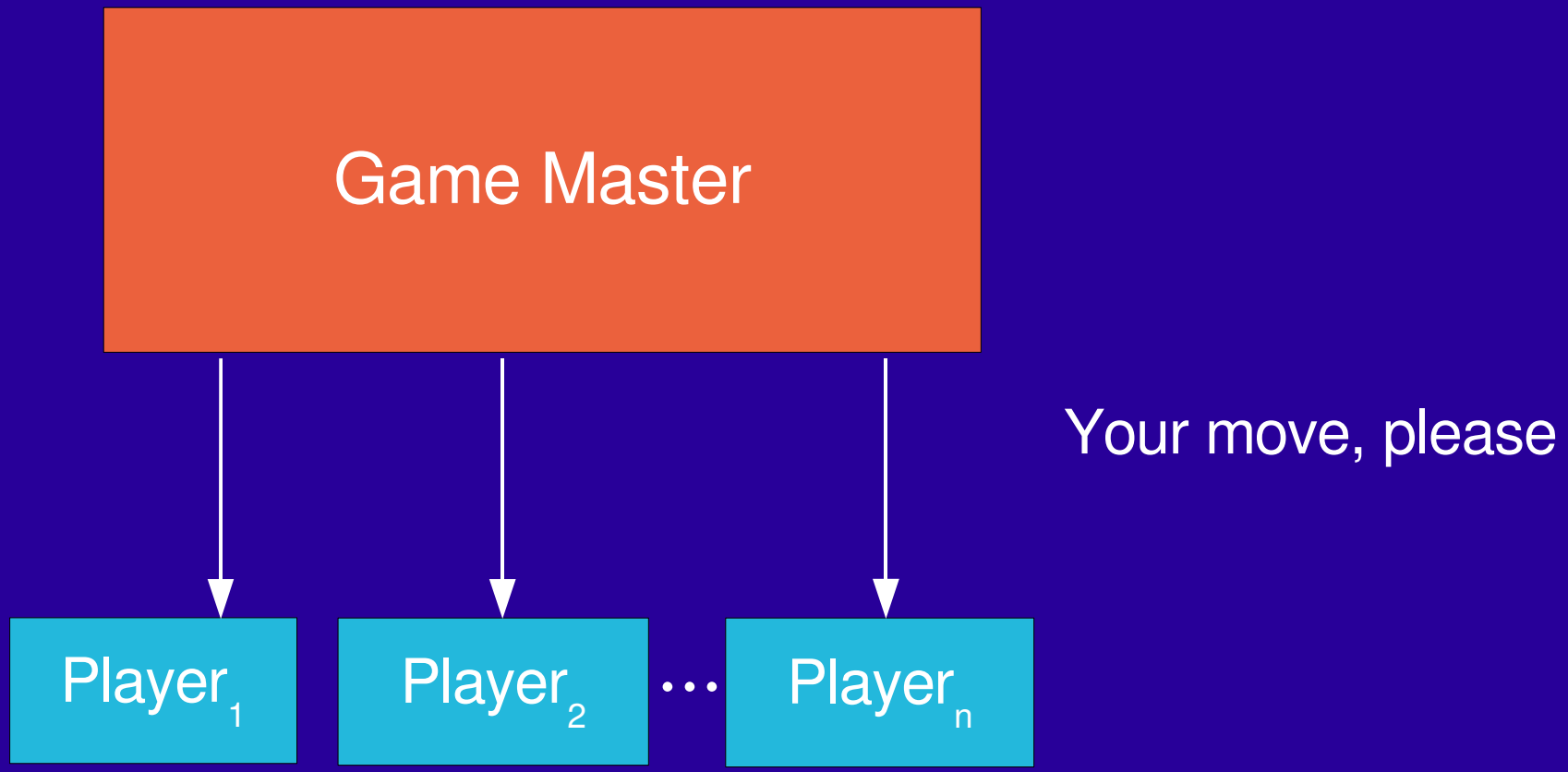


Game description
Time to think: 1,800 sec
Time per move: 45 sec
Your role

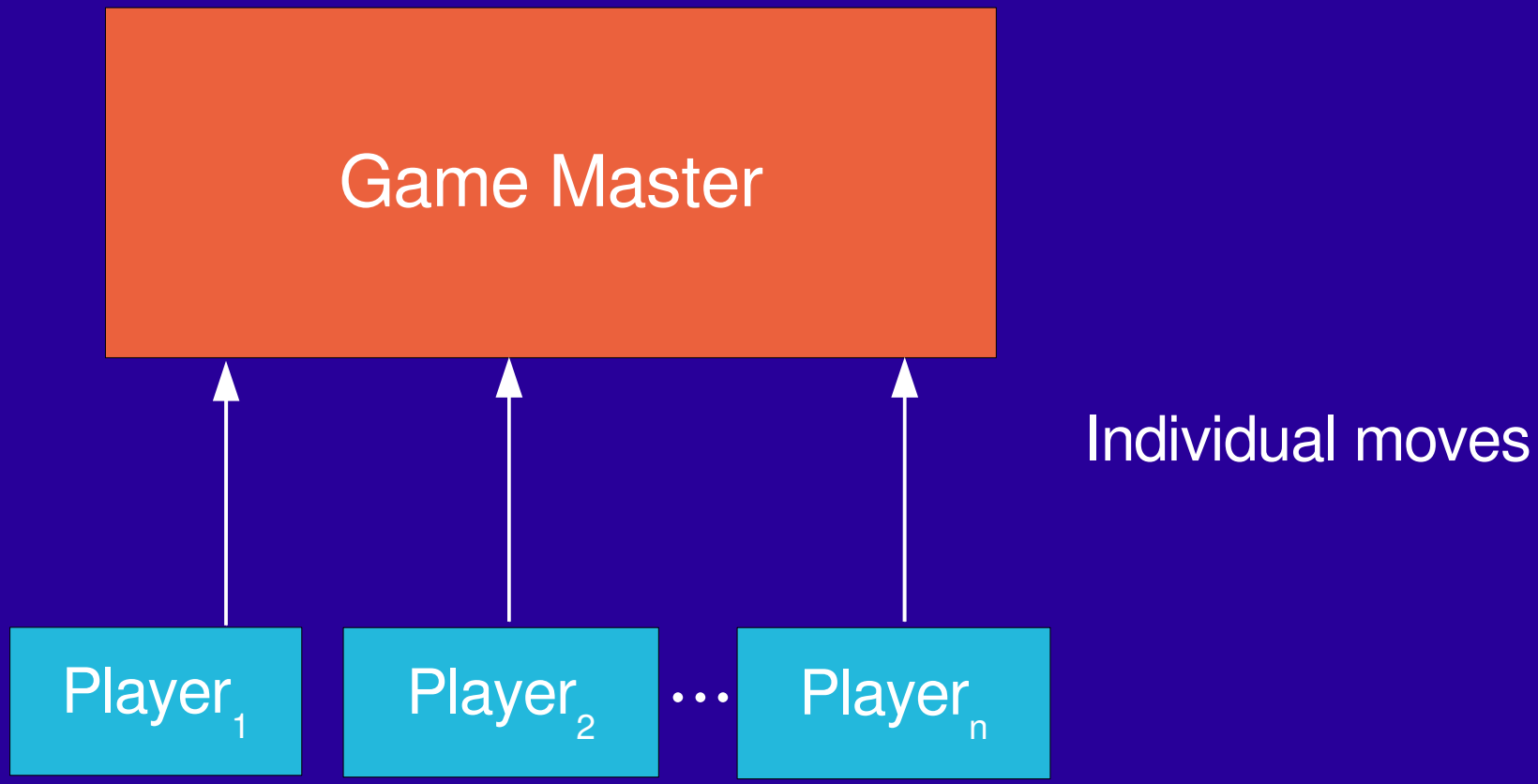
How it Works



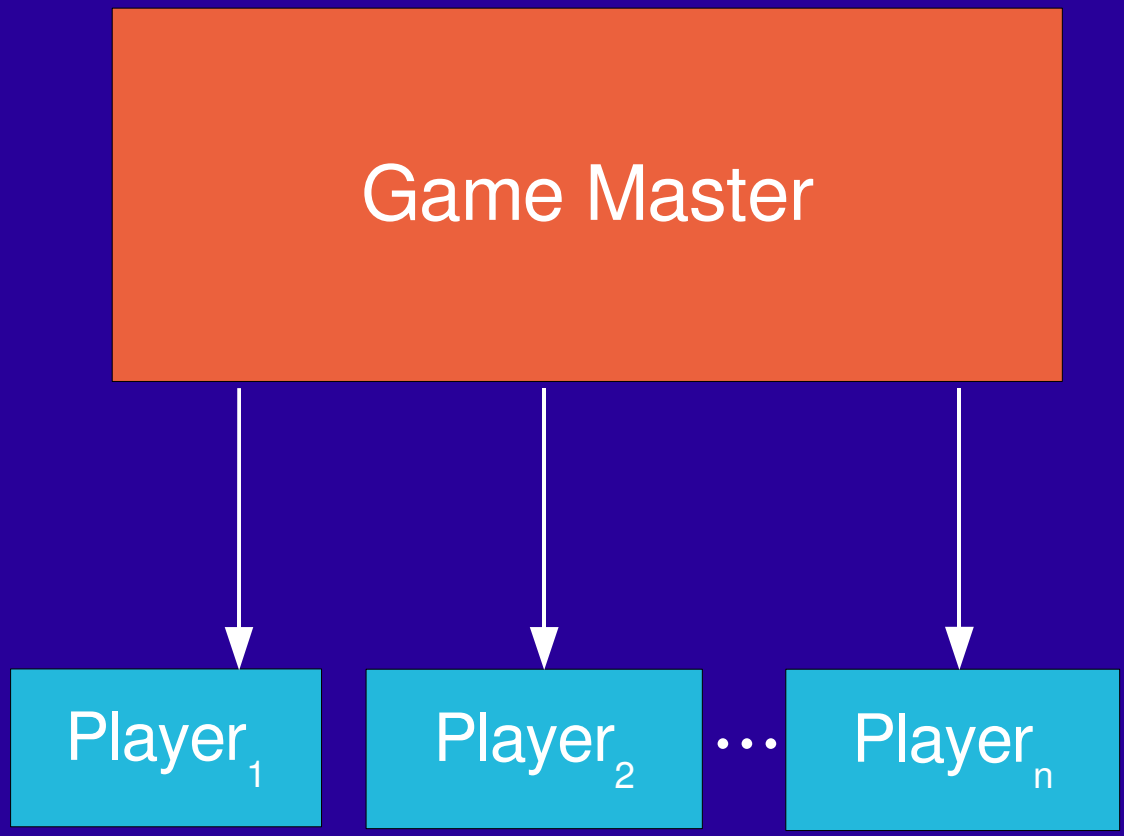
How it Works



How it Works

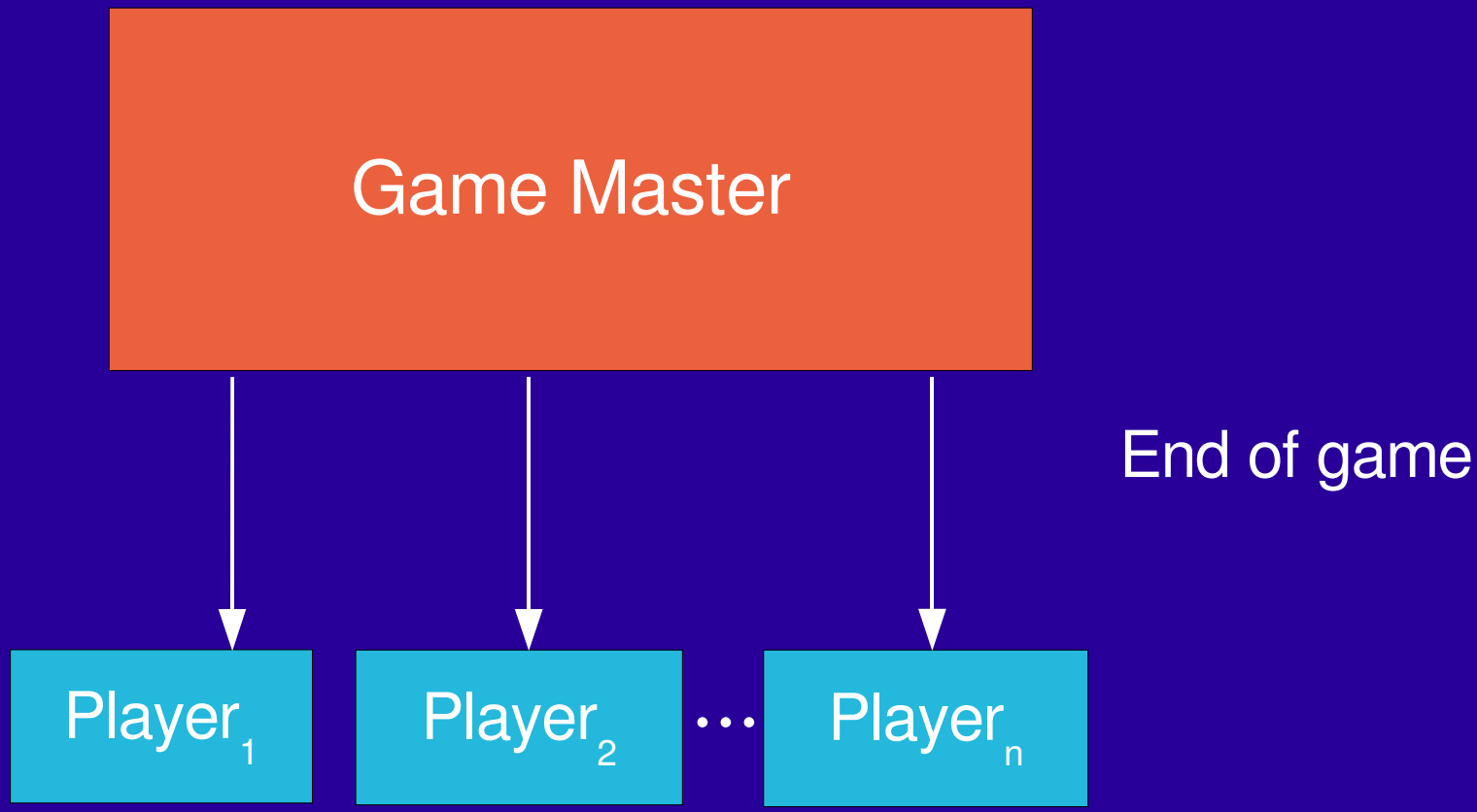


How it Works



Individual information about state/moves

How it Works



Game Descriptions

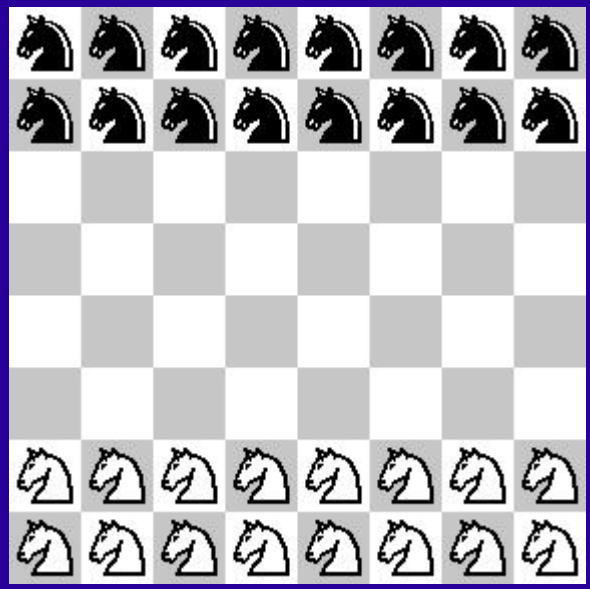
Games are described by logic programs using a few pre-defined keywords

```
role(jane).
role(rick).
role(random).
card(♣7).  card(♣8).  ...  card(♣ace).

init(dealingRound).
```

Game Descriptions (Cont'd)

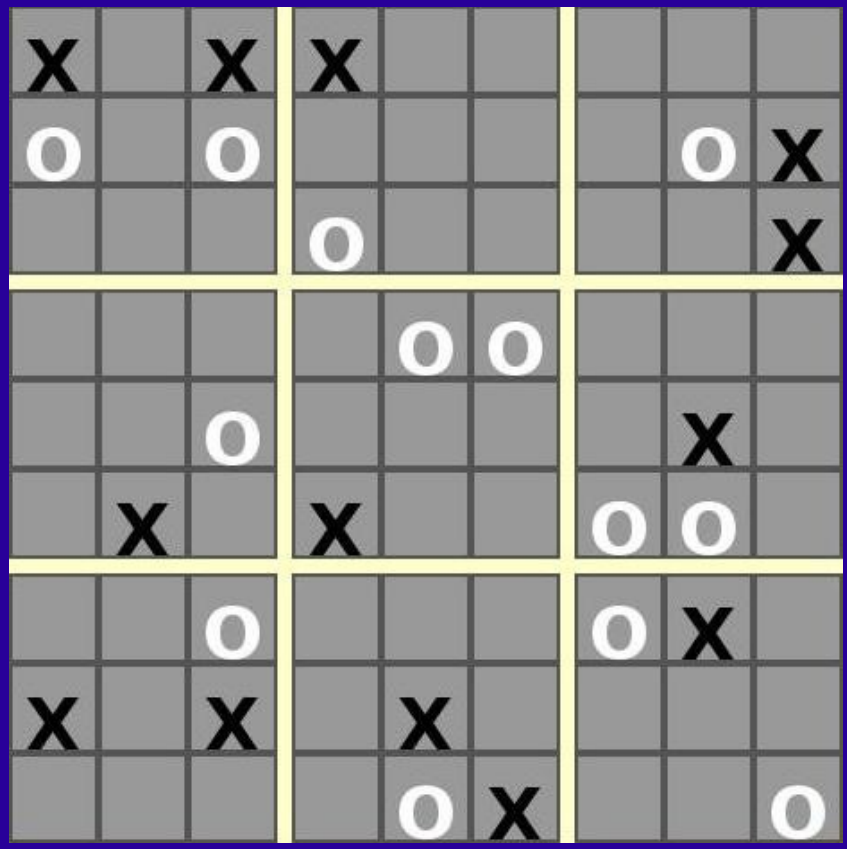
```
legal(random,deal(C,D)) <= true(dealingRound),
                           card(C), card(D),
                           distinct(C,D).

sees(jane,yourCard(C)) <= does(random,deal(C,D)).
sees(rick,yourCard(D)) <= does(random,deal(C,D)).

legal(jane,...) <= ...
legal(rick,...) <= ...

terminal <= ...
goal(P,N) <= ...
```

Example 1



AAAI 2007

Example 2



AAAI 2010

A Vibrant Reserch Area

History

1968 J. Pitrat: “Realization of a General Game Playing Program”

2005 First GGP Competition @AAAI

2009 First GGP Workshop @IJCAI

2010 First Technical Paper Session on GGP @AAAI

Research centers

Dresden, Edmonton, Paris, Potsdam, Reykjavik, Stanford, Sydney, ...

Online repositories

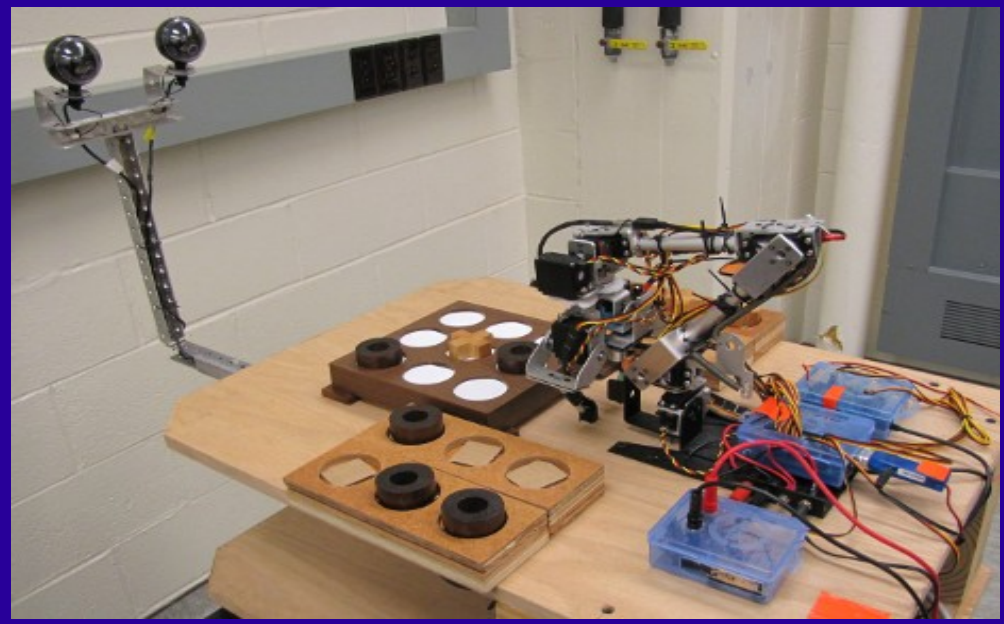
- games.stanford.edu (description language, competition)
- general-game-playing.de (game server, basic players, literature)

Two Questions

- Can a general game player beat Deep Blue in chess?
  - No (but may change in the future)
  - Focus is on general players, not savants
  - There is a market for a chess computer that is weaker but can adapt to any chess variant without being re-programmed
- Isn't a general game player still a very special system?
  - Yes, but will change in the future

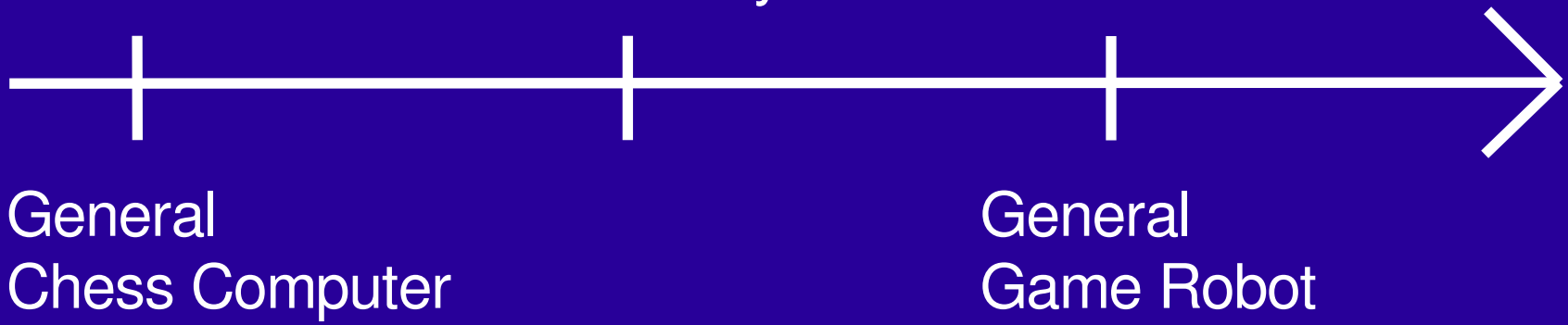
Some Ideas for General General Game Playing

- Natural Language
  - Systems understand game rules in (controlled) English
- Vision
  - Camera system identifies new boards and pieces
- Robotics
  - Robotic manipulation of game hardware



(Purdue University 2010)

A Continuous Scale



From General Game Playing to General X

The idea behind General Game Playing can be applied to other areas, bringing today's AI systems to a new level of generality

Systems with general intelligence

- understand descriptions of new environments and tasks
- adapt to these environments/tasks without human intervention

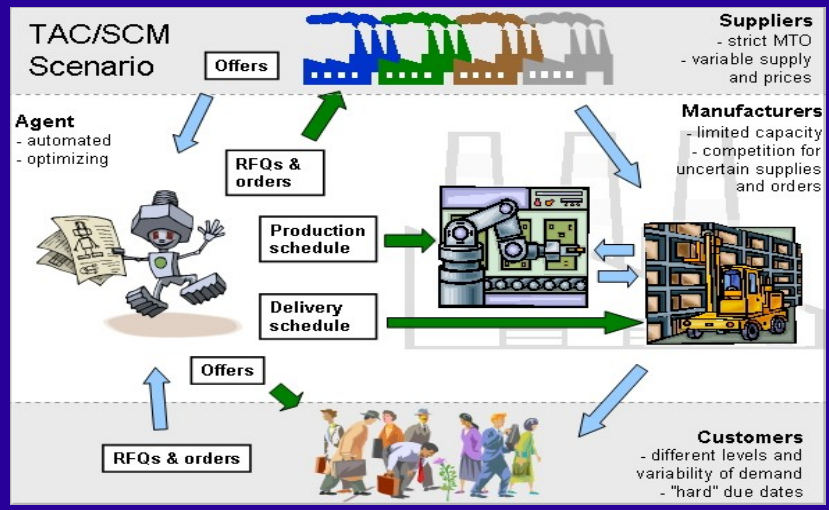
How to create your own General AI Challenge:

- Define a broad—but sufficiently restricted—problem class X
- Design a suitable communication/description language for X

Two Random Ideas

General Trading Agents

- understand new trading scenarios
- trade without human intervention



General Robots

- understand new tasks
- adapt without human intervention

Part II:

Addressing a General AI Challenge

A Brief History of AI

“Silver bullets” have been proposed throughout the history, eg

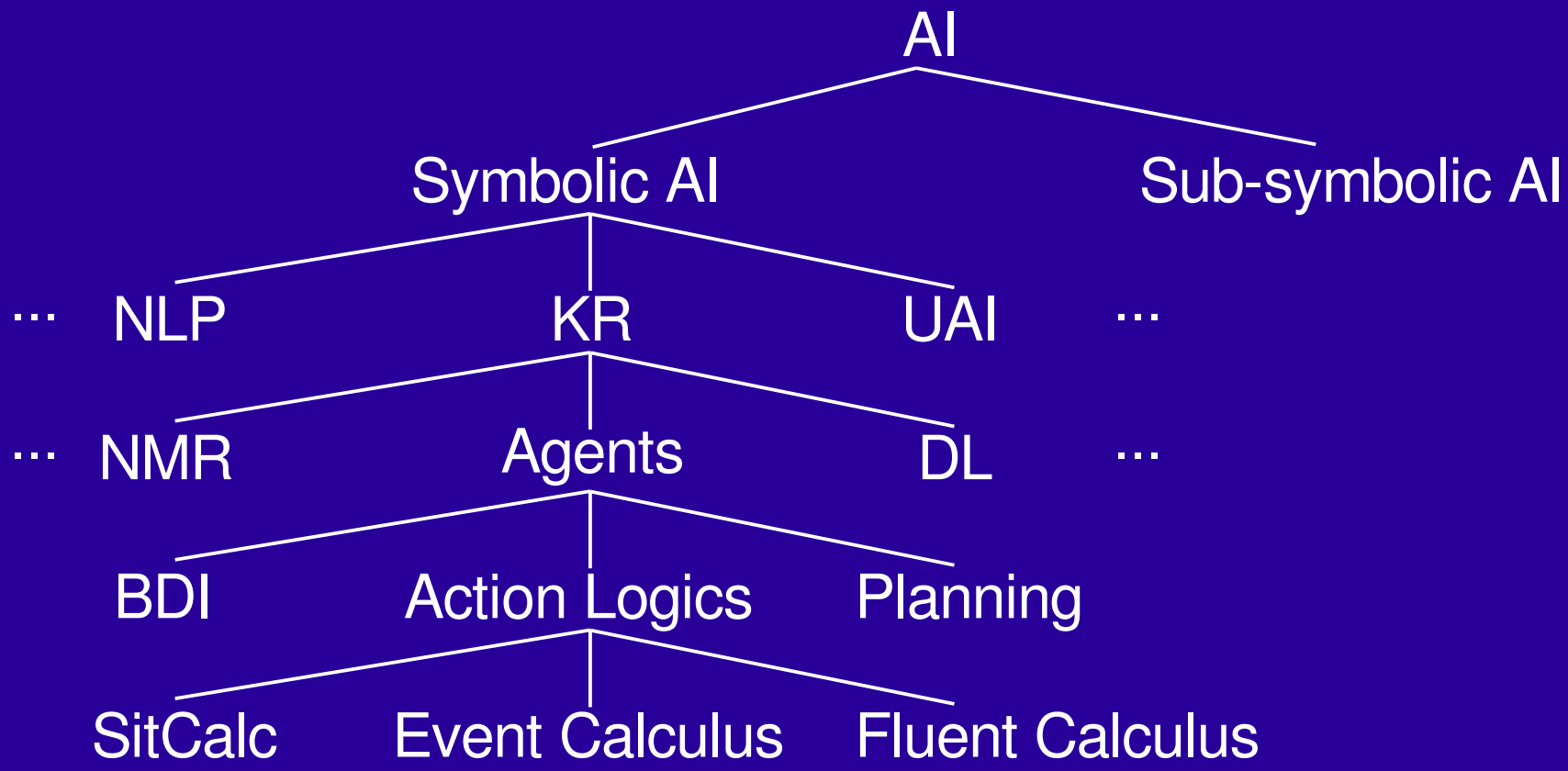
- GOFAI (1960's)
- Sub-symbolic AI (1980's)
- Bayesian AI (1990's)

but:

- different problems may require different representations
- different tasks may require different computations

AI Today

Individual theories cater for individual aspects of intelligence



Specialization: Pro

Focusing on a single, narrow AI problem allows to

- use a tailor-made representation
- gain a deeper understanding of the fundamental and computational issues related to this particular aspect of AI

Today, there exist a variety of

- well-understood approaches—for many individual aspects of AI
- highly optimized algorithmic solutions—to many specific problems

Specialization: Cons

- There is a danger to fiddle with minor details
- AI Challenges require to address a range of aspects together
  - Challenge 1: combine different representations
  - Challenge 2: integrate different implementations

Systems with General Intelligence

Programs or robots with general intelligence (GI) must exhibit many facets of intelligence

→ need to integrate successful AI methods

Top-Down

Take well-defined GI challenge

- identify sub-tasks
- choose methods to combine
- build integrated system

Bottom-Up

Choose and combine

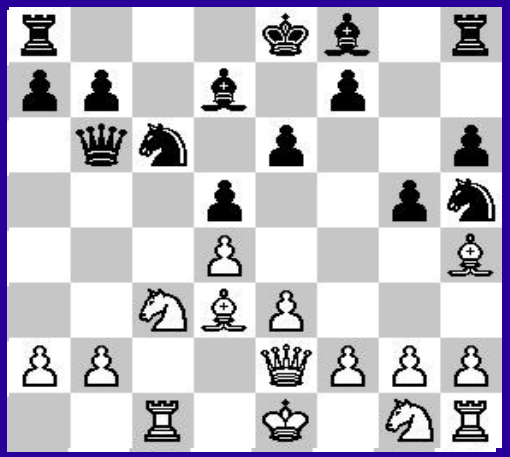
- representation formalisms
- algorithmic solutions
- implementations

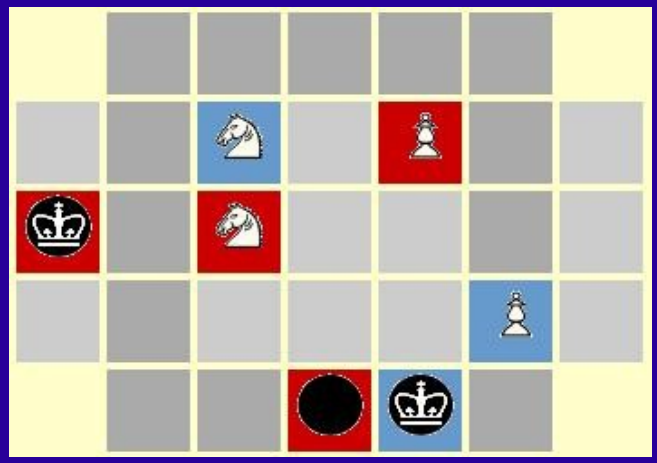
Top-Down Combinations (Example)

—

FLUXPLAYER

General Game Playing Systems





A General Game Player requires methods from

- Knowledge Representation and Reasoning
- Planning and Search
- Computer Game Playing
- Learning

FLUXPLAYER

Our General Game Player FLUXPLAYER combines

- Reasoning about Actions ("FLUX", to understand the game rules)
- Planning and Search
- Automated Theorem Proving (to generate knowledge about a game)
- Fuzzy Logic (to evaluate intermediate positions)
- Neural Nets (to improve parameter settings of evaluation functions)

FLUXPLAYER's performance in all previous GGP Championships

- AAAI: 2005 Semifinal, 2006 Winner, 2007 Second, 2008 Semifinal
- IJCAI: 2009 Second

FLUXPLAYER

Two examples of research output from this Grand Challenge

- Answer Set Programming for verification of dynamic systems

  (Schiffel & T, IJCAI 2009; T & Voigt, AAAI 2010)

- Combining Neural Networks with Symbolic Logic

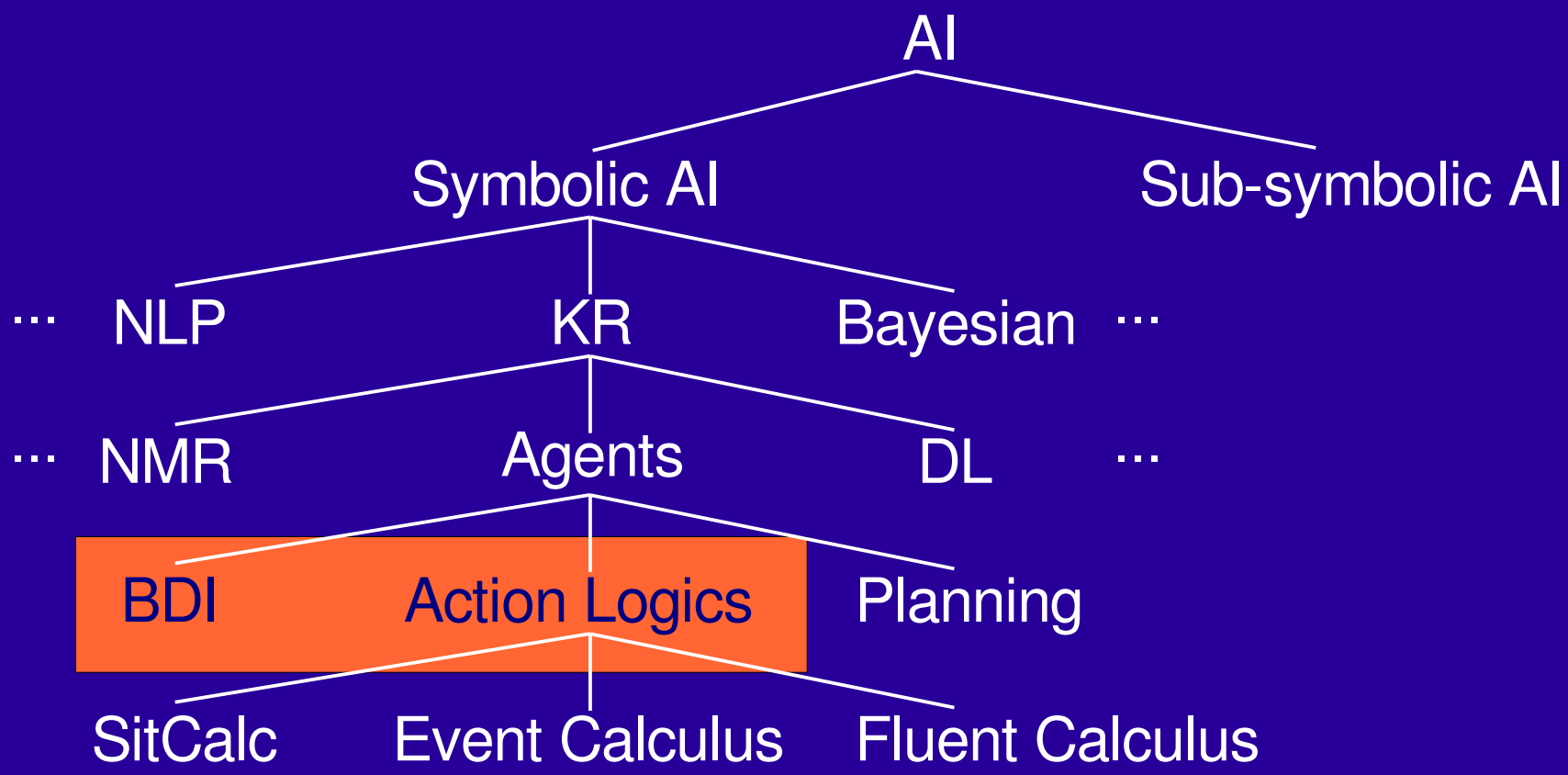
  (Michulke & T, ECML 2009)

Bottom-Up Combination: Example

—

BDI-Based Agent Programs
&
Action Logics

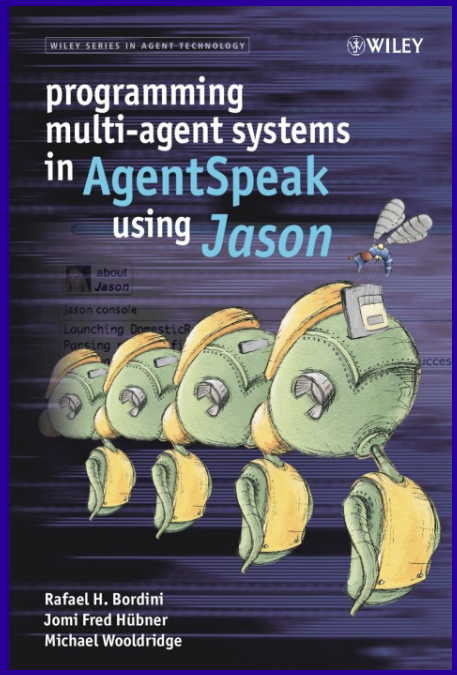
Combining Formalisms



Two Distinct Areas with a Similar Goal

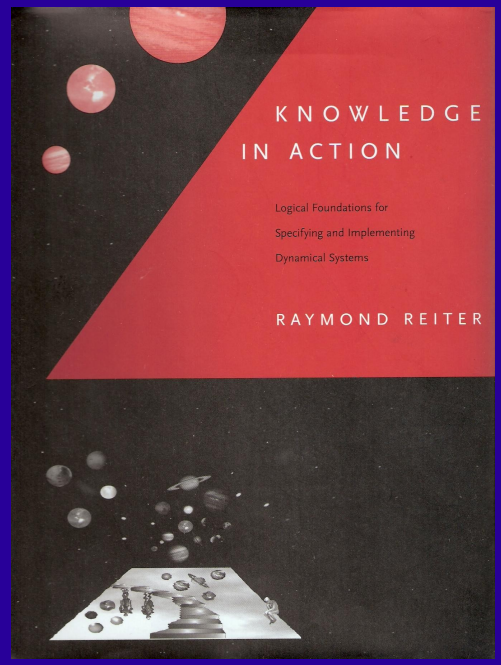
BDI-based Programming

- since early 1990's
- to build cognitive agents



Action Logics

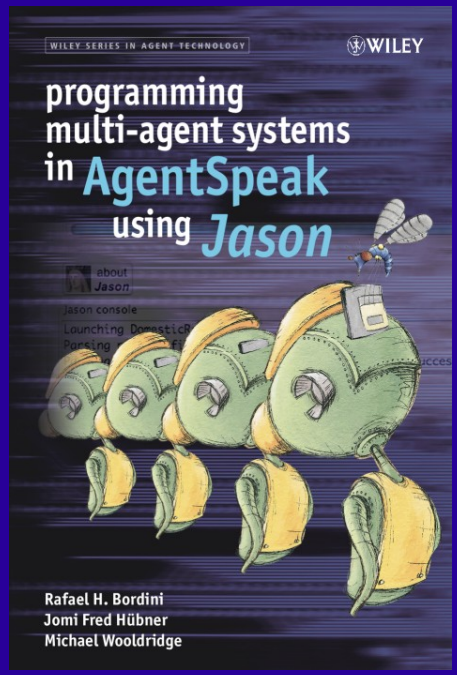
- since late 1960's
- theory of cognitive agents



Similar Goal—Different Strengths

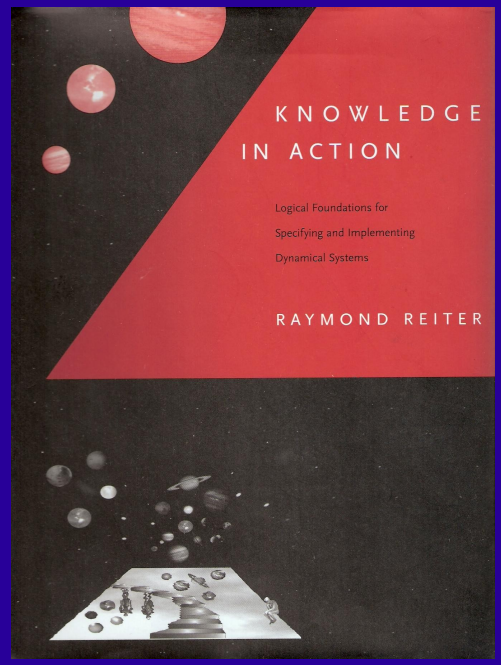
BDI-based Programming

+ practical programming
– simplistic action model



Action Logics

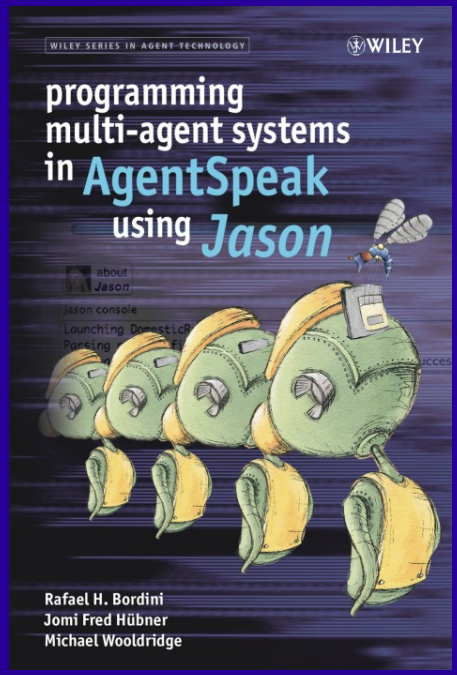
+ rich action model
– barely used in practice



Why Combine the Two?

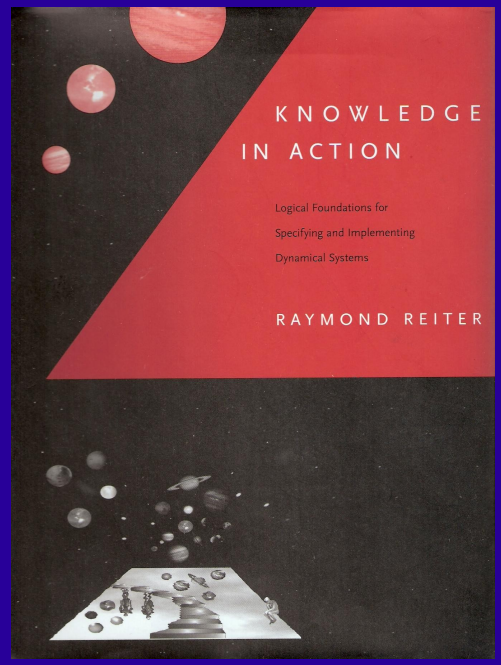
BDI-based Programming

+ practical programming
– simplistic action model



Action Logics

+ rich action model
– barely used in practice



Need to Align Representations

Main issue: two methods based on different representations

- Agent programs are collections of reactive behaviors

```
+!capture(X) : ¬have(X) | !nextto(X); get(X); !at(home)
```

- Action knowledge is given in form of logical formulas

```
poss(get(X),S) ≡ holds(nextto(X),S)

holds(have(X),do(A,S)) ⊂ A = get(X) ∨ holds(have(X),S)
```

- Reactive programs come with operational semantics, based on the (Beliefs, Desires, Intentions)-model of agents
- Action theories have declarative semantics, based on logic

Solution

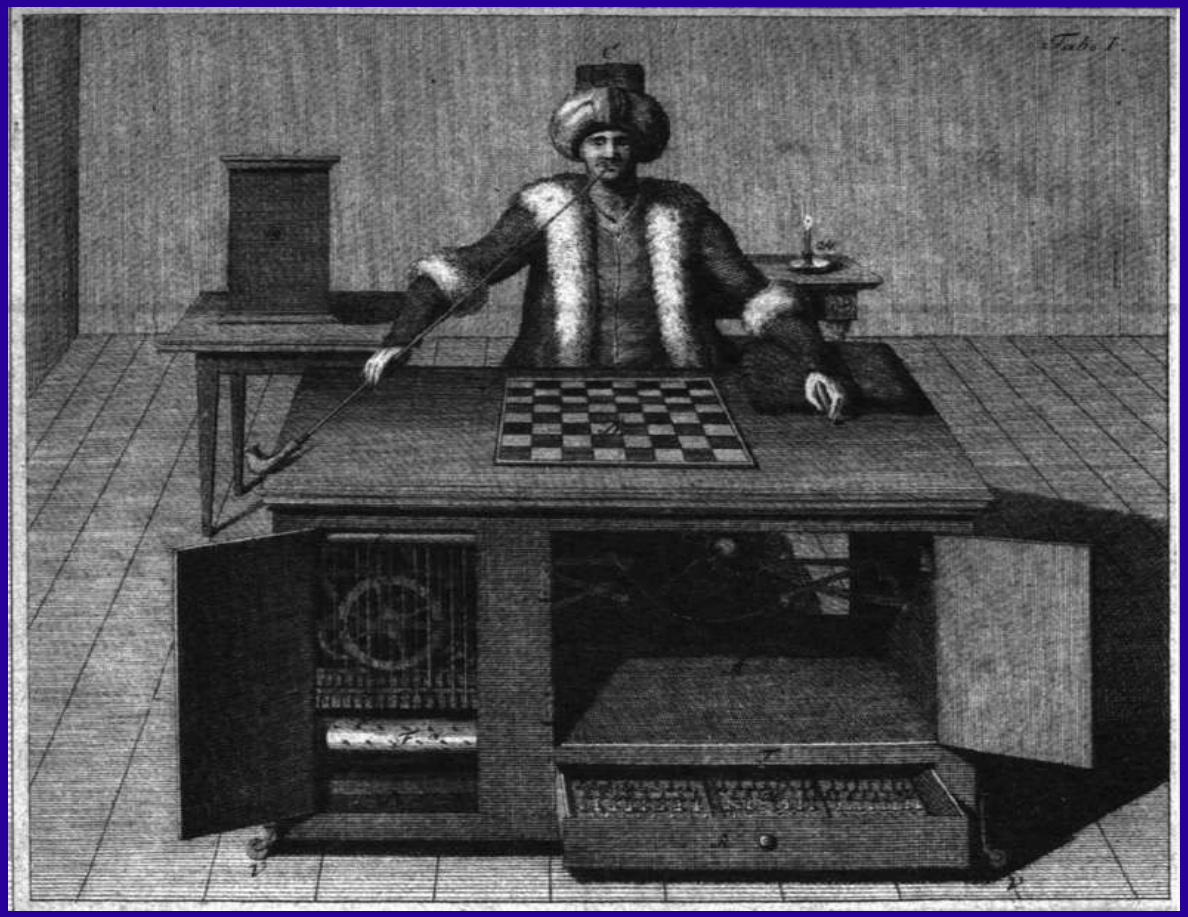
A bridging language helps aligning the two representations

- Agent Logic Programs
  - extend logic programs (Prolog) by actions
  - come with an operational semantics
  - and with a declarative semantics
- Resulting integration
  - provides declarative semantics for BDI-based languages
  - provides formal underpinnings for combining implementations
  - is correct—provided 8(!) assumptions and conditions are met

(MT, KR 2010)

Conclusion

First Demonstration of AI



Turk
(Vienna 1770)

Future Demonstrations of AI

Systems with general intelligence

- understand descriptions of radically new environments/tasks
- adapt to these environments/tasks without human intervention

When built, these systems

- provide impressive demonstrations of AI's potential
- lift a specific AI field to a new level

To do so,

- the technology is out there
- but combining AI methods can be a challenge of its own