The Art and Science of Action Programming Languages

Michael Thielscher

Chapter 1

Introduction

Introduction

This tutorial is concerned with knowledge representation and reasoning techniques for systems that act autonomously in a complex environment.

Introduction

The aim is to endow these systems with knowledge of how their world functions, in particular knowledge of their own abilities to act.

Reasoning about this knowledge allows to

- make autonomous decisions
- exhibit goal-oriented behavior

Introduction

Foundations

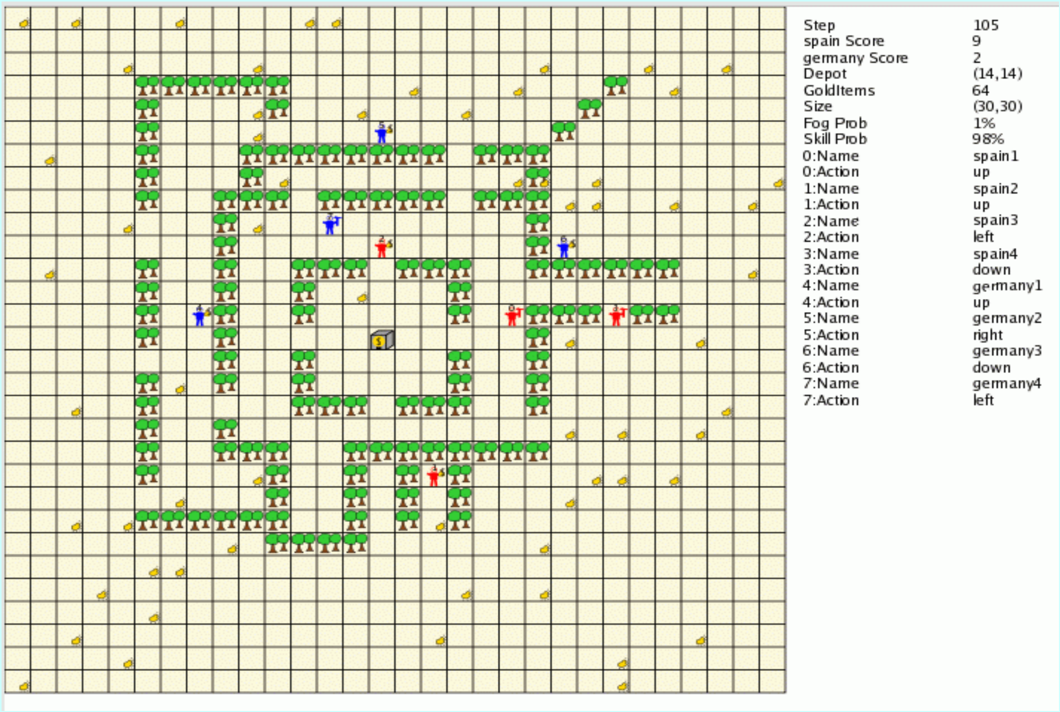
- Symbolic representation
- Logical reasoning

Advantages

- High degree of abstraction
- High-level action programming languages
- Large amount of diverse knowledge
- Uncertainty via disjunction / existential quantification

Application I: Multiagent Systems

Multiagent systems consist of autonomous, intelligent agents



CLIMA-06 Contest: Two competing teams of agents

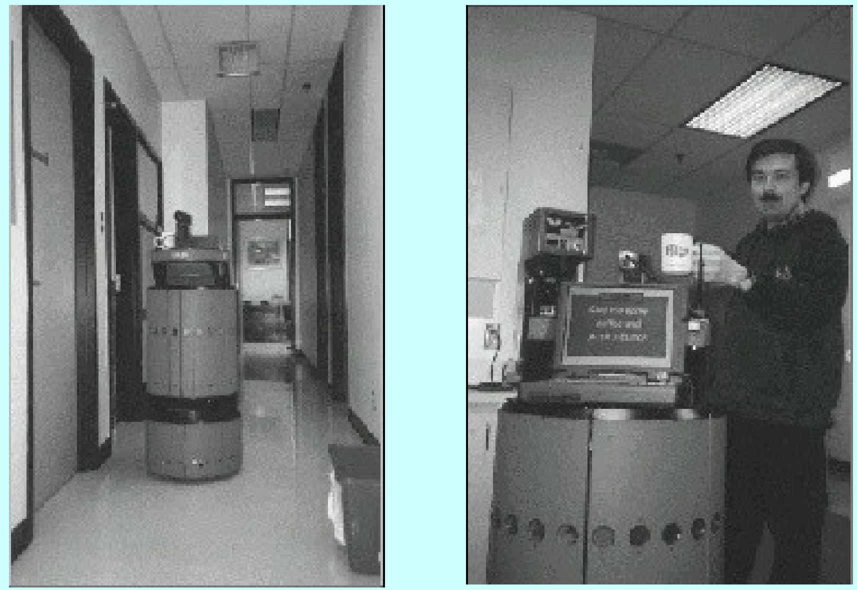
Reasoning Tasks

- Verifying applicability of actions
  Can I go forward now?
- Prediction
  Where will the gold be after moving forward?
- Planning / Goal-oriented behavior
  Which part of the environment shall I explore?

Application II: Cognitive Robots

Autonomous robots making high-level decisions

Museum Guide RHINO



Coffee Delivery Robot

Reasoning Tasks

- Verifying applicability of actions

  Can I deliver coffee now?

- Prediction

  Where will the coffee be after picking it up?

- Planning / Goal-oriented behavior

  In which order shall I satisfy the requests?

- Explanation

  I can't deliver coffee now, what went wrong?

Application III: General Game Playing

A General Game Player is a system that

- understands formal descriptions of arbitrary games
  Solitaire, 4-dimensional Chess,
  n-player Monopoly, Texas Hold'Em, ...
- plays these games without human intervention

Reasoning Tasks

- Verifying applicability of actions (aka moves)
  Can I move my king now?
- Prediction
  Will I still be able to castle afterwards?
- Planning / Goal-oriented behavior
  How can I win the game?

The Game Playing Metaphor

Agent in static world

- Single-Player Game

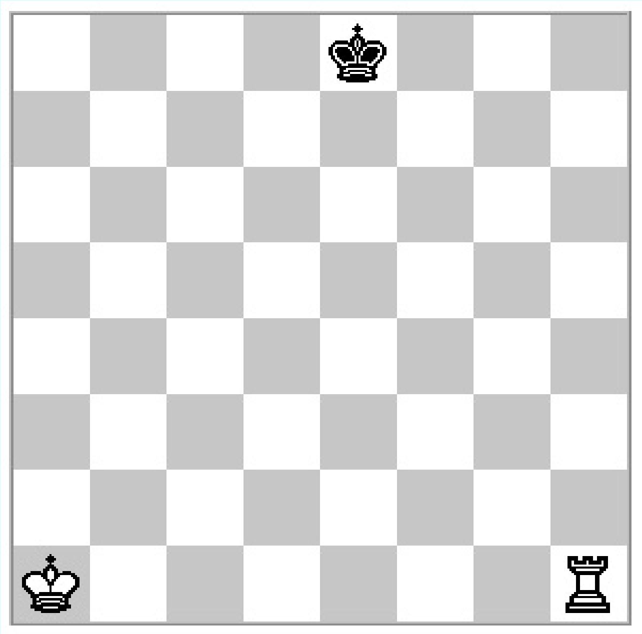
Agent/Robot in dynamic world

- Two-Player Game (world as opponent)

System with multiple agents

- *n*-Player Game

Example



Three autonomous agents.

Goal of White-King and White-Rook: checkmate Black-King

Historical Development

1963 Situation Calculus – the oldest KR formalism

1969 Frame Problem

1972f STRIPS and other planning languages

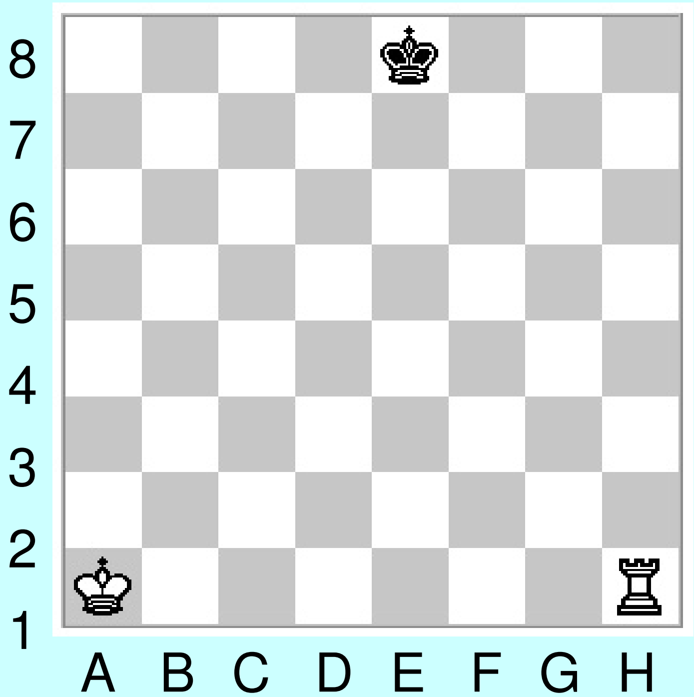
1991f Solving the frame problem:
Situation-, Event-, Fluent Calculus

1997f GOLOG, FLUX – action programming languages

Chapter 2

Situation Calculus and the Frame Problem

Representation



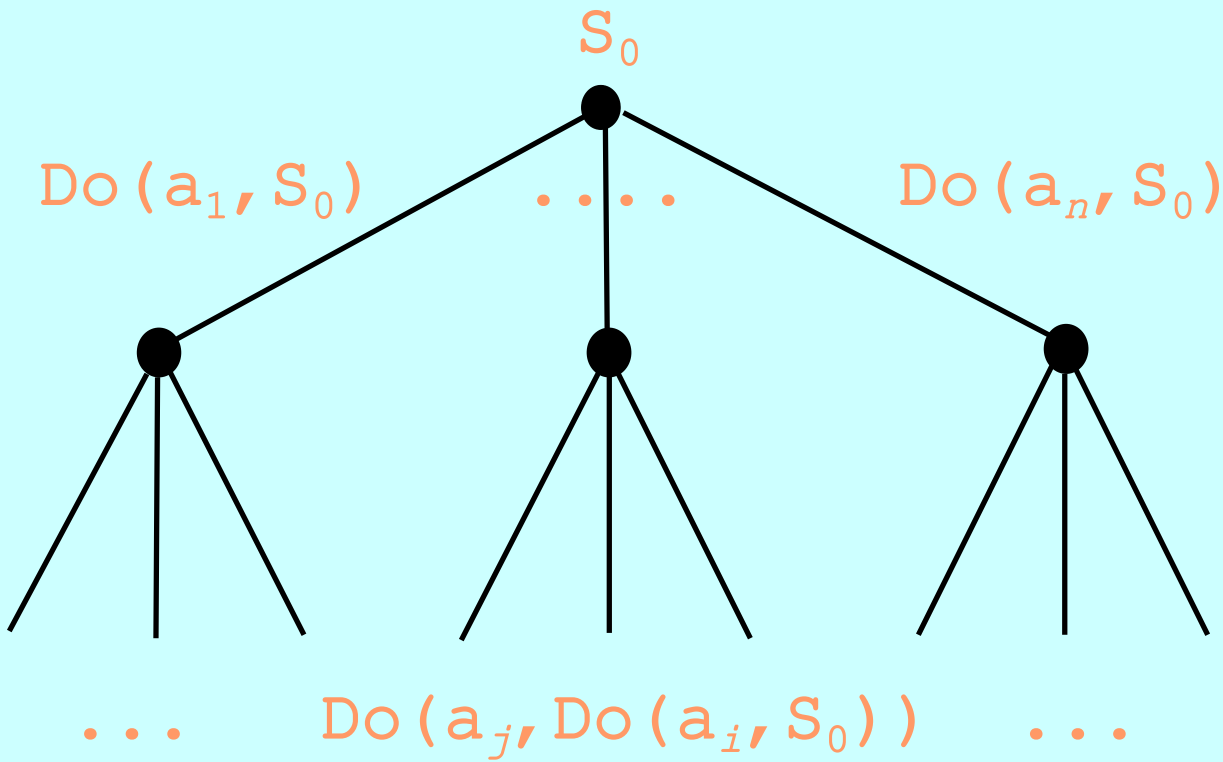
- Fluents

  `Cell(agent,x,y)`

- Actions

  `Move(agent,u,v,x,y)`

Time Structure: Situation Tree



Knowledge I: Abilities

- State Knowledge

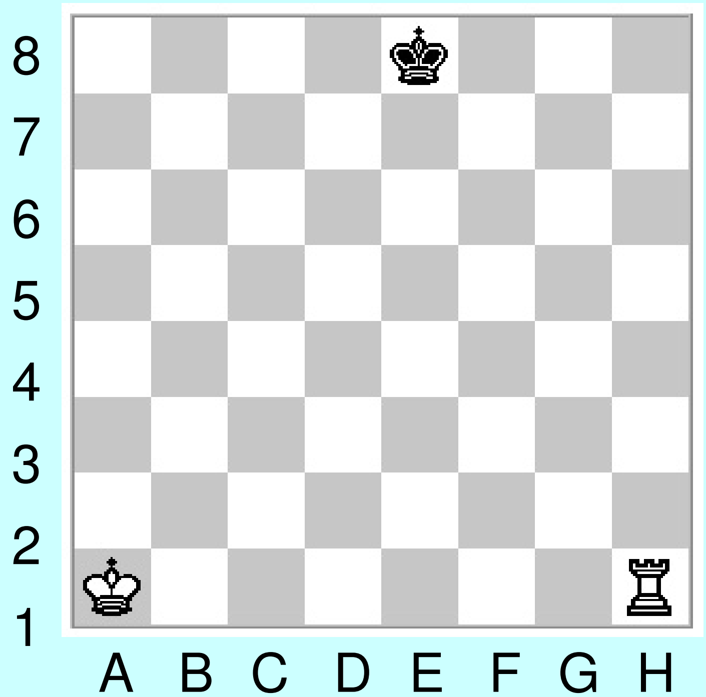
  $\texttt{Holds(Cell(WhiteKing,A,1),}S_0\texttt{)} \land$
  $\texttt{Holds(Cell(WhiteRook,H,1),}S_0\texttt{)}$

- Precondition Axioms

  $\texttt{Poss(Move(WhiteKing,u,v,x,y),s)} \leftrightarrow$
  $\texttt{Holds(Cell(WhiteKing,u,v),s)} \land$
  $\texttt{Legal-King-Move(u,v,x,y,s)}$

  $\texttt{Poss(Move(WhiteRook,u,v,x,y),s)} \leftrightarrow$
  $\texttt{Holds(Cell(WhiteRook,u,v),s)} \land$
  $\texttt{Legal-Rook-Move(u,v,x,y,s)}$

Example



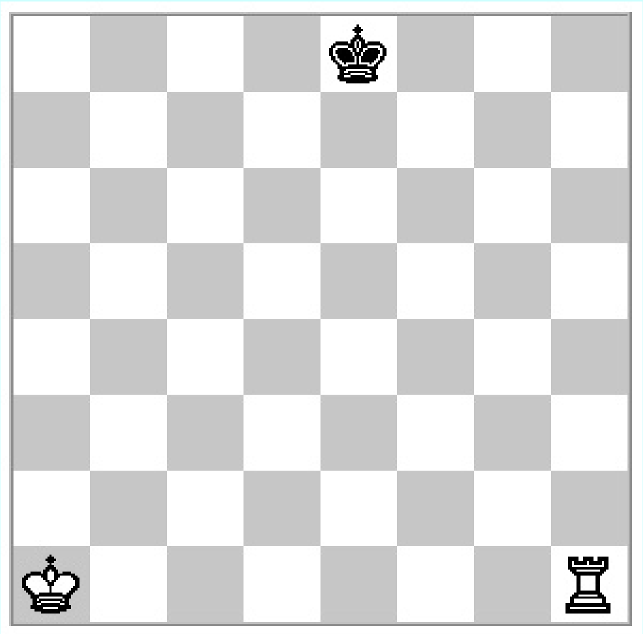
$\texttt{Poss(Move(WhiteKing,A,1,B,2),S}_0\texttt{)}$

Knowledge II: Effects

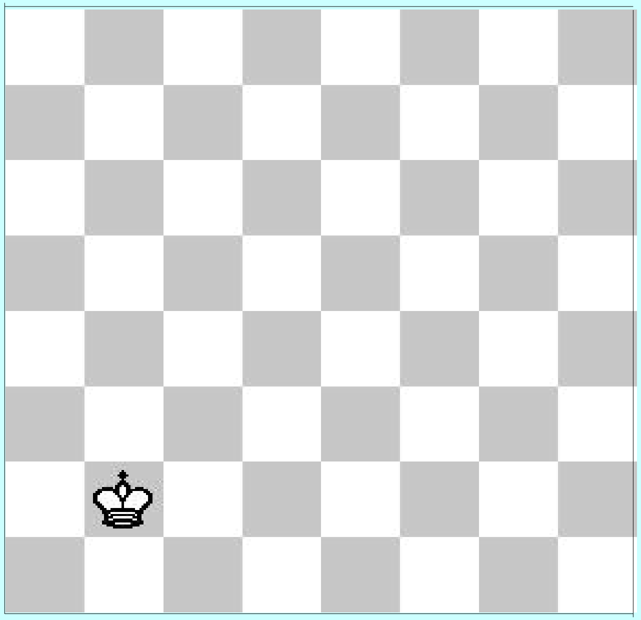
- Effect Axioms

```
Holds(Cell(agent,x,y),
        Do(Move(agent,u,v,x,y),s))

¬Holds(Cell(agent,u,v),
        Do(Move(agent,u,v,x,y),s))

Holds(Cell(c,x,y),s) →
   ¬Holds(Cell(c,x,y),
        Do(Move(agent,u,v,x,y),s))
```

Example



S_0



$Do(Move(WhiteKing, A,1,B,2), S_0)$

The Frame Problem (1969)

- The effect axioms do not allow to conclude

  $$\text{Holds}(\text{Cell}(\text{WhiteRook}, H, 1), \text{Do}(\text{Move}(\text{WhiteKing}, A, 1, B, 2), S_0))$$

- Additional frame axioms are needed

  $$\text{Holds}(\text{Cell}(c, i, j), \text{Do}(\text{Move}(p, u, v, x, y), s)) \leftarrow \text{Holds}(\text{Cell}(c, i, j), s) \wedge [i \neq u \vee j \neq v] \wedge [i \neq x \vee j \neq y]$$

- Representation not succinct
- Reasoning inefficient

On the Frame Problem

Daniel Dennett:

A new, deep epistemological problem – accessible in principle but unnoticed by generations of philosophers – brought to light by the novel methods of AI and still far from being solved.

On the Frame Problem

Ray Reiter:

If AI can be said to have a classic problem, then the Frame Problem is it.

Like all good open problems it is subtle, challenging, and it has led to significant new technical and conceptual developments in the field.

STRIPS (1972)

```
Move(p,u,v,x,y)
ADD-LIST: {Cell(p,x,y)}
DEL-LIST: {Cell(p,u,v)}
```

```
{Cell(WhiteKing,A,1),
 Cell(WhiteRook,H,1),
 Cell(BlackKing,E,8)}
⇨
{Cell(WhiteKing,B,2),
 Cell(WhiteRook,H,1),
 Cell(BlackKing,E,8)}
```

Other Planning Languages: ADL (1989), PDDL (1998)

- Conditional effects
- Fluents that are unknown
- Efficient planning techniques
  - Partial Order Planning
  - Graphplan
  - Planning as satisfiability
- Limited expressiveness:

  No disjunctive state knowledge, quantification,...

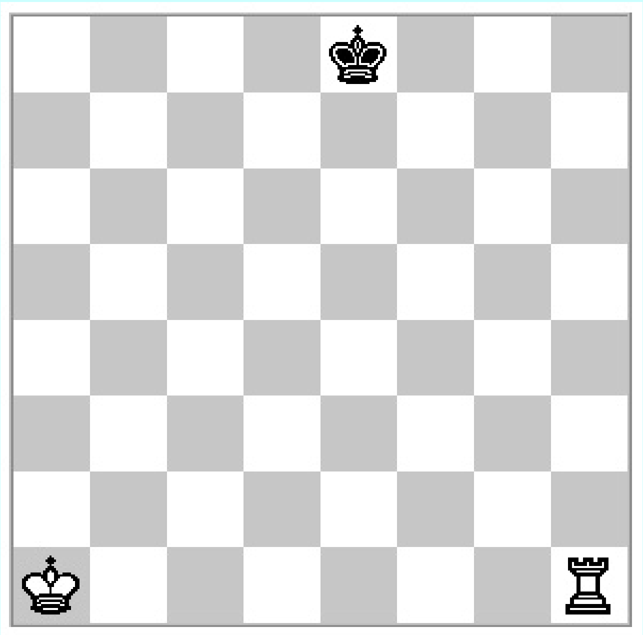
  $(\exists x)(\mathtt{Holds}(\mathtt{Dist}(x), S) \wedge 3.8 \leq x \leq 4.7)$

Successor State Axioms (1991)

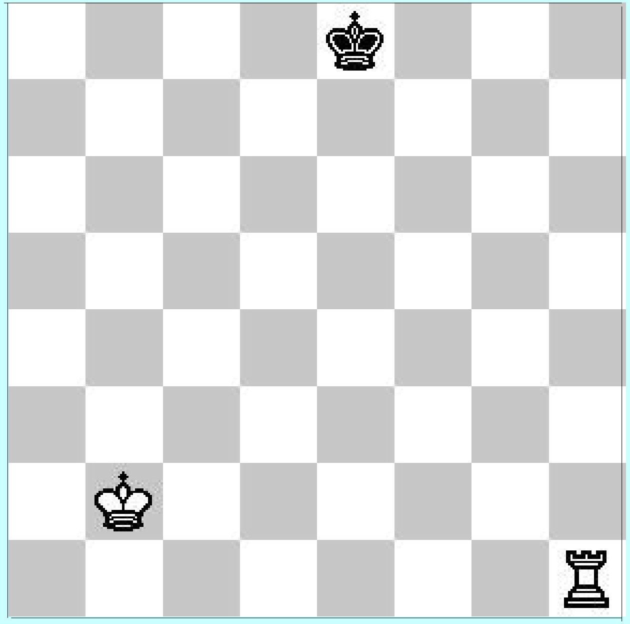
$$\mathrm{Holds}(\mathit{Fluent},\mathrm{Do}(a,s)) \leftrightarrow$$
$$\quad \mathrm{effect}_{+}(a,s)$$
$$\quad \vee\ [\mathrm{Holds}(\mathit{Fluent},s) \wedge \neg \mathrm{effect}_{-}(a,s)]$$

$$\mathrm{Holds}(\mathrm{Cell}(c,x,y),\mathrm{Do}(a,s)) \leftrightarrow$$
$$\quad a = \mathrm{Move}(c,u,v,x,y)$$
$$\quad \vee\ [\mathrm{Holds}(\mathrm{Cell}(c,x,y),s) \wedge$$
$$\qquad (\forall u,v)\, a \neq \mathrm{Move}(c,x,y,u,v) \wedge$$
$$\qquad (\forall p,u,v)\, a \neq \mathrm{Move}(p,u,v,x,y)]$$

Example



S_0



$\mathtt{Do(Move(WhiteKing,\ A,1,B,2),}S_0\mathtt{)}$

Basic Action Theories

- Precondition axioms for all actions
- Successor state axioms for all fluents
- Initial state axiom

Extensions

- Knowledge and sensing via modality `K(s,s')`
- Nondeterministic actions
- Indirect effects of actions (Ramification Problem)
- Probabilities `P(s)=p`
- Time and continuous processes `Start(s)=t`

Chapter 3

Action Programming in GOLOG

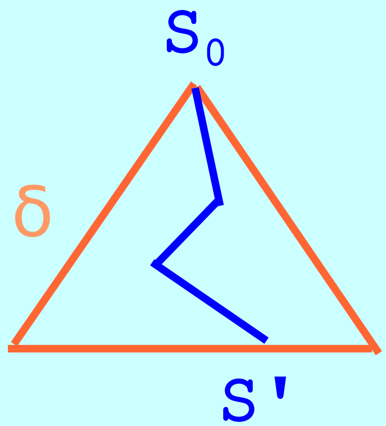
Example Program

```
proc DeliverCoffee
  while (∃p)wantsCoffee(p)∧¬hasCoffee(p) do
    πp.goto(coffeeMachine); pickUp(Coffee);
       goto(p); giveCoffee(p)
  endWhile
endProc


proc goto(loc)
  if robotLocation(rloc) then drive(rloc,loc) endIf
endProc
```

Programming Constructs

primitive actions	`a`
tests	$\phi?$
sequence	$\delta_1 ; \delta_2$
nondeterministic choice	π `x`. δ `(x)`
nondeterministic choice	$\delta_1 \mid \delta_2$
nondeterministic iteration	δ^*
if ϕ then δ_1 else δ_2 endIf	$[\phi? ; \delta_1] \mid [\neg\phi? ; \delta_2]$
while ϕ do δ endWhile	$[\phi? ; \delta]^* ; \neg\phi?$

Execution Modes



- Offline execution: Find terminating run, then execute
  = Planning with search heuristics

  proc `main`
      while ¬`goal` **do** `anyAction` **endWhile**
  endProc

- Interleaved Online-/Offline execution

ConGOLOG

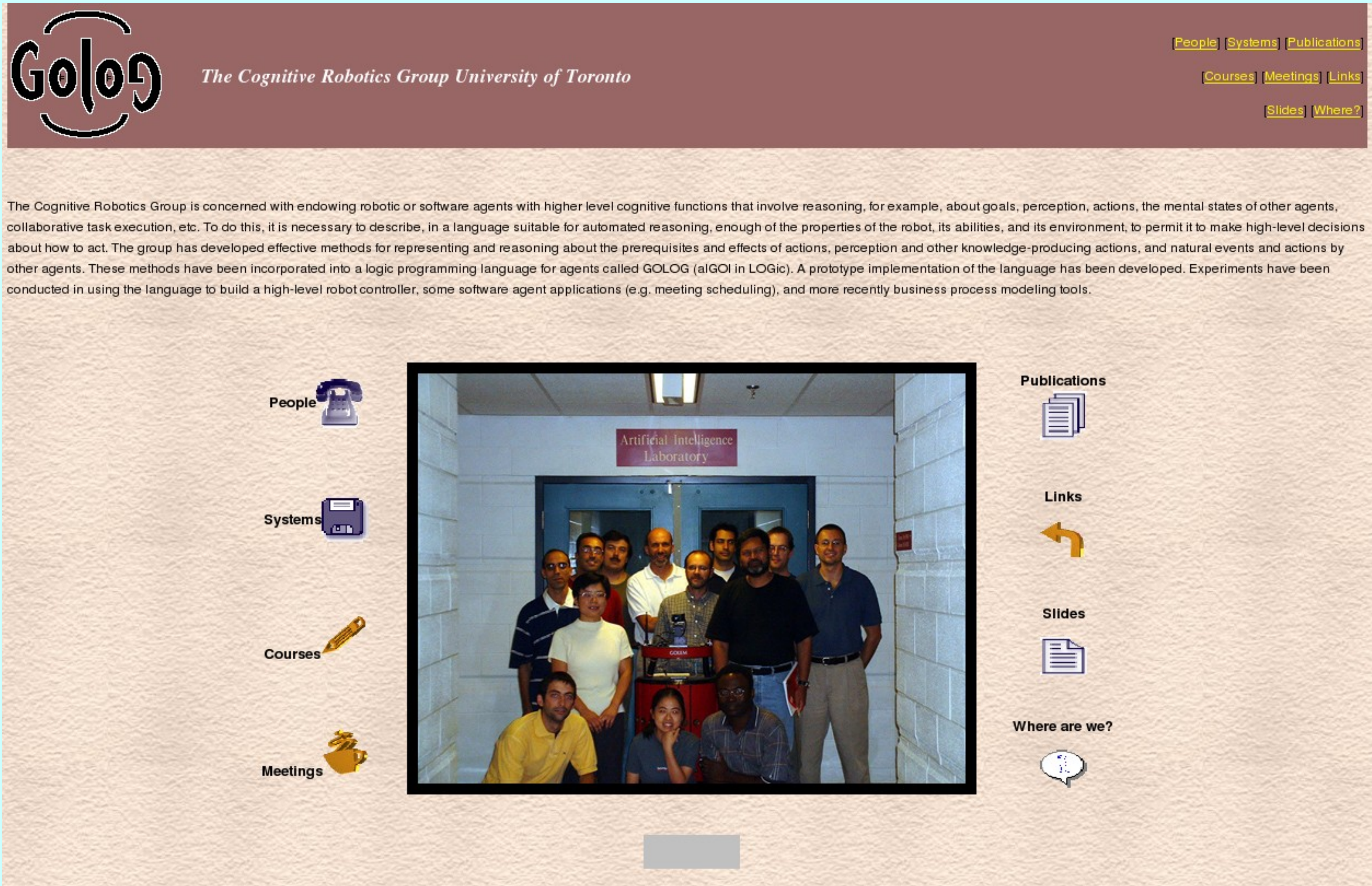
concurrent execution	$\delta_1 \parallel \delta_2$
concurrency w/ priorities	$\delta_1 >> \delta_2$
concurrent iteration	δ'
interrupt	$\langle \phi \rightarrow \delta \rangle$

Knowledge-Based GOLOG

knowledge tests — Knows(ϕ)? Kwhether(ϕ)?

sensing actions

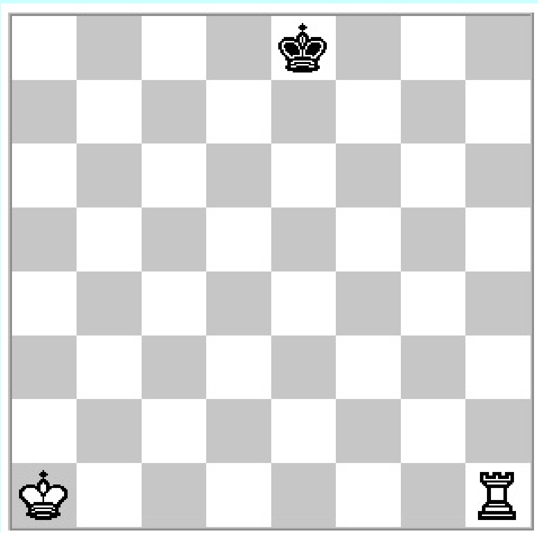
www.cs.toronto.edu/cogrobo/



Chapter 4

Fluent Calculus and FLUX

The Frame Problem Revisited

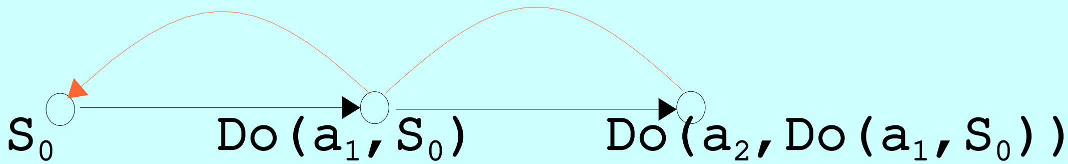


$$
\begin{aligned}
&\mathrm{Holds}(\mathrm{Cell}(c,x,y),\mathrm{Do}(a,s)) \leftrightarrow \\
&\quad a = \mathrm{Move}(c,u,v,x,y) \\
&\quad \vee\ [\mathrm{Holds}(\mathrm{Cell}(c,x,y),s) \wedge \\
&\qquad (\forall)\, a \neq \mathrm{Move}(c,x,y,u,v) \wedge \\
&\qquad (\forall)\, a \neq \mathrm{Move}(p,u,v,x,y)]
\end{aligned}
$$

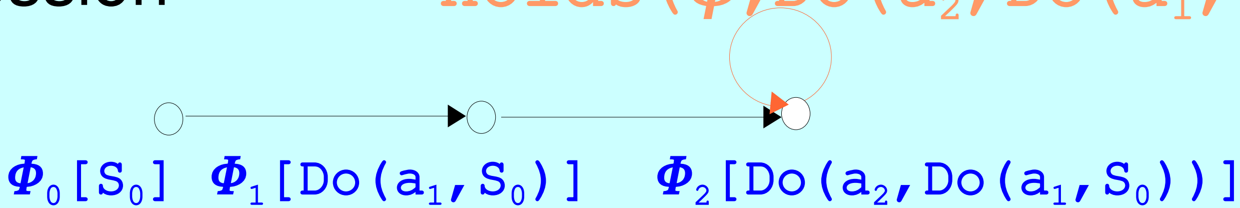
- 64 instances needed to update the state
- Regression requires to roll back an entire history

Regression vs. Progression

Regression $Holds(\phi, Do(a_2, Do(a_1, S_0)))?$



Progression $Holds(\phi, Do(a_2, Do(a_1, S_0)))?$



State Update Axioms (1999)

$$\Delta(\texttt{s}) \rightarrow \texttt{State(Do(}\textit{Action}\texttt{,s))=}$$
$$\texttt{State(s)} - \texttt{effects}_{-} + \texttt{effects}_{+}$$

$$\texttt{Holds(Cell(c,x,y),s)} \rightarrow$$
$$\texttt{State(Do(Move(p,u,v,x,y),s))=}$$
$$\texttt{State(s)} - \texttt{Cell(c,x,y)}$$
$$- \texttt{Cell(p,u,v)} + \texttt{Cell(p,x,y)}$$

- Axiomatic definition of + and -

Fluent Calculus: Basic Action Theories

- Precondition axioms for all actions
- State update axioms for all actions
- Initial state axiom

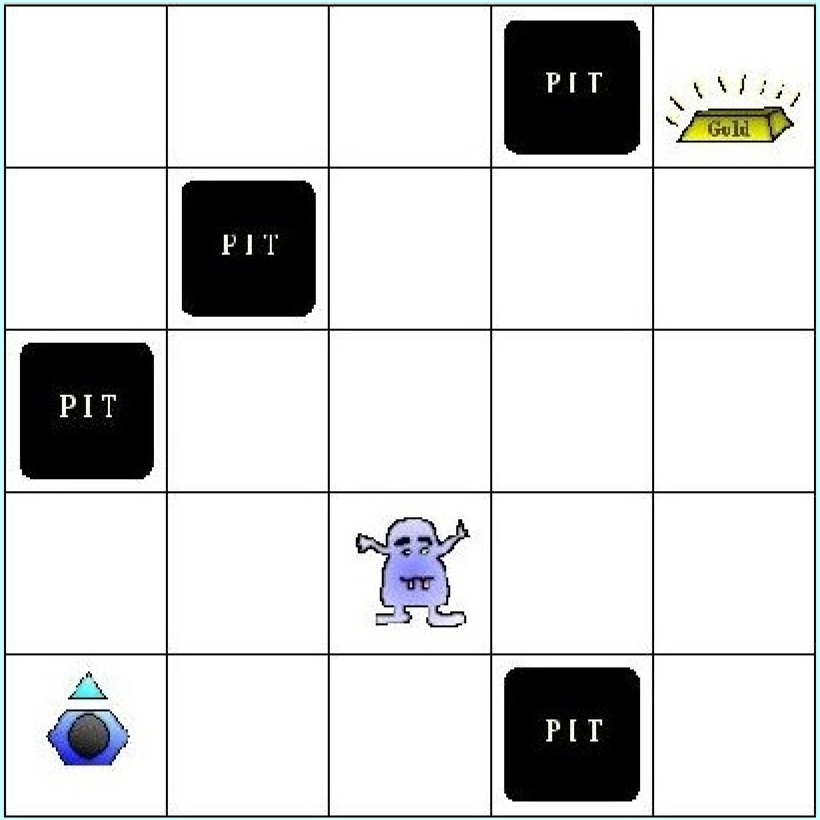
Extensions

- Knowledge and sensing via modality `KState(s,z)`
- Nondeterministic actions
- Indirect effects of actions (Ramification Problem)
- Unexpected action failure (Qualification Problem)
- Probabilities `P(z,s)=p`
- Time and continuous processes

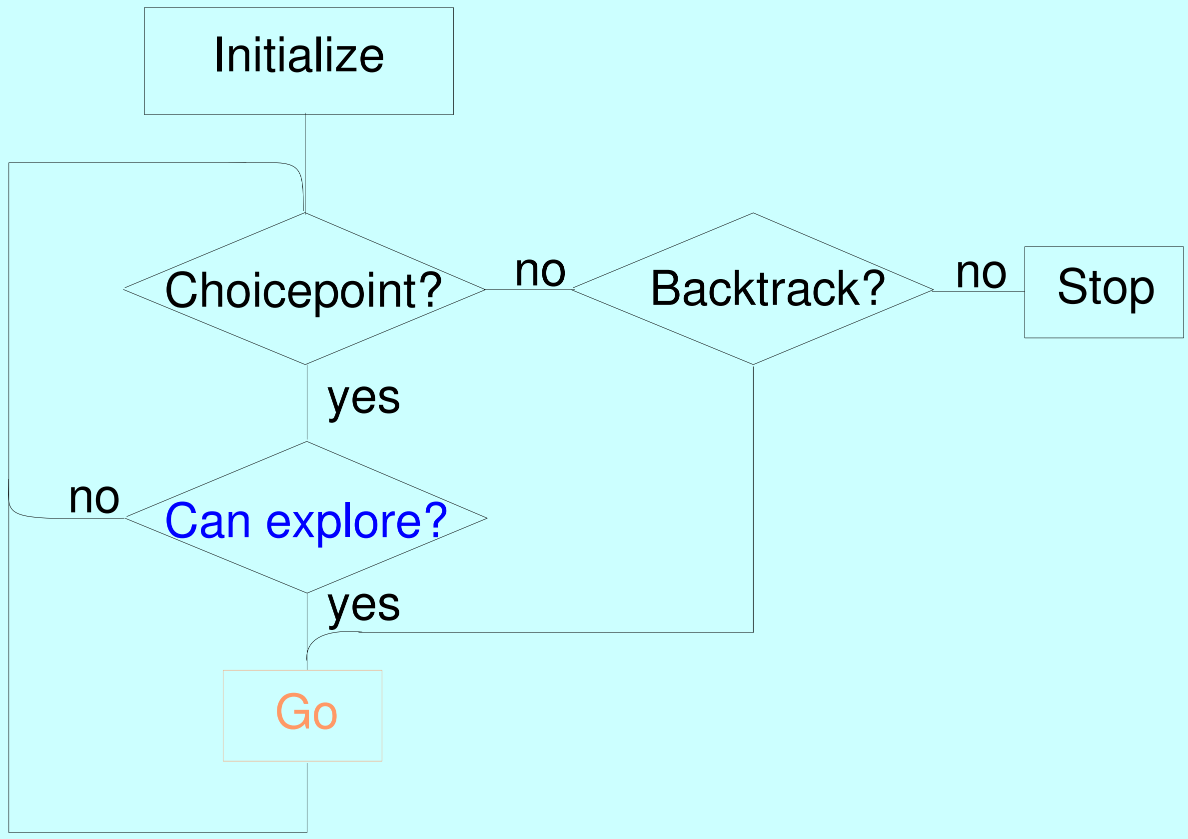
Action Programming with FLUX

- Constraint Logic Programming-based language
- State update as Constraint Rewriting
- Progression
- Interleaved Online-/Offline execution

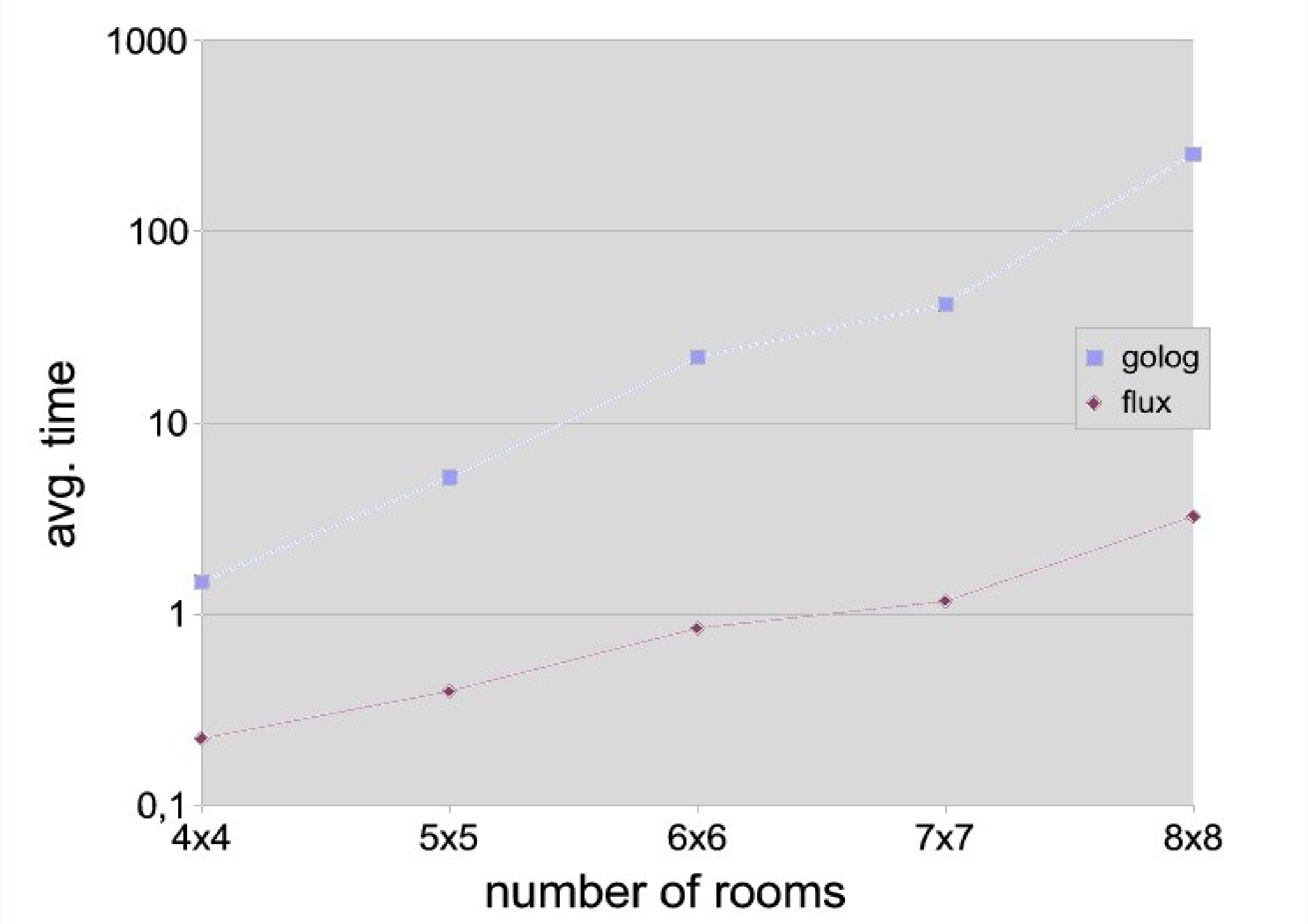
Example: Wumpus World



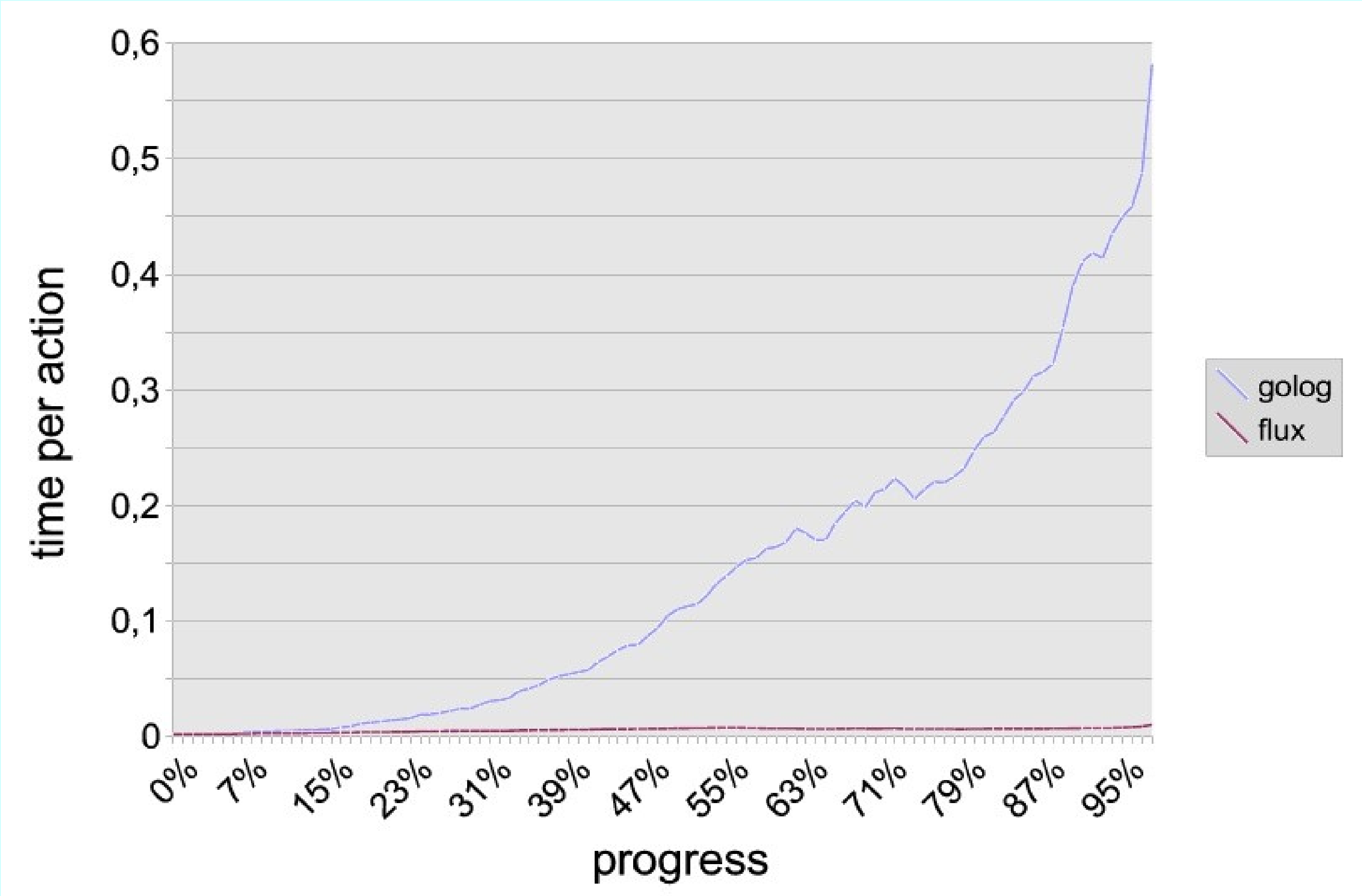
A Systematic Exploration Strategy



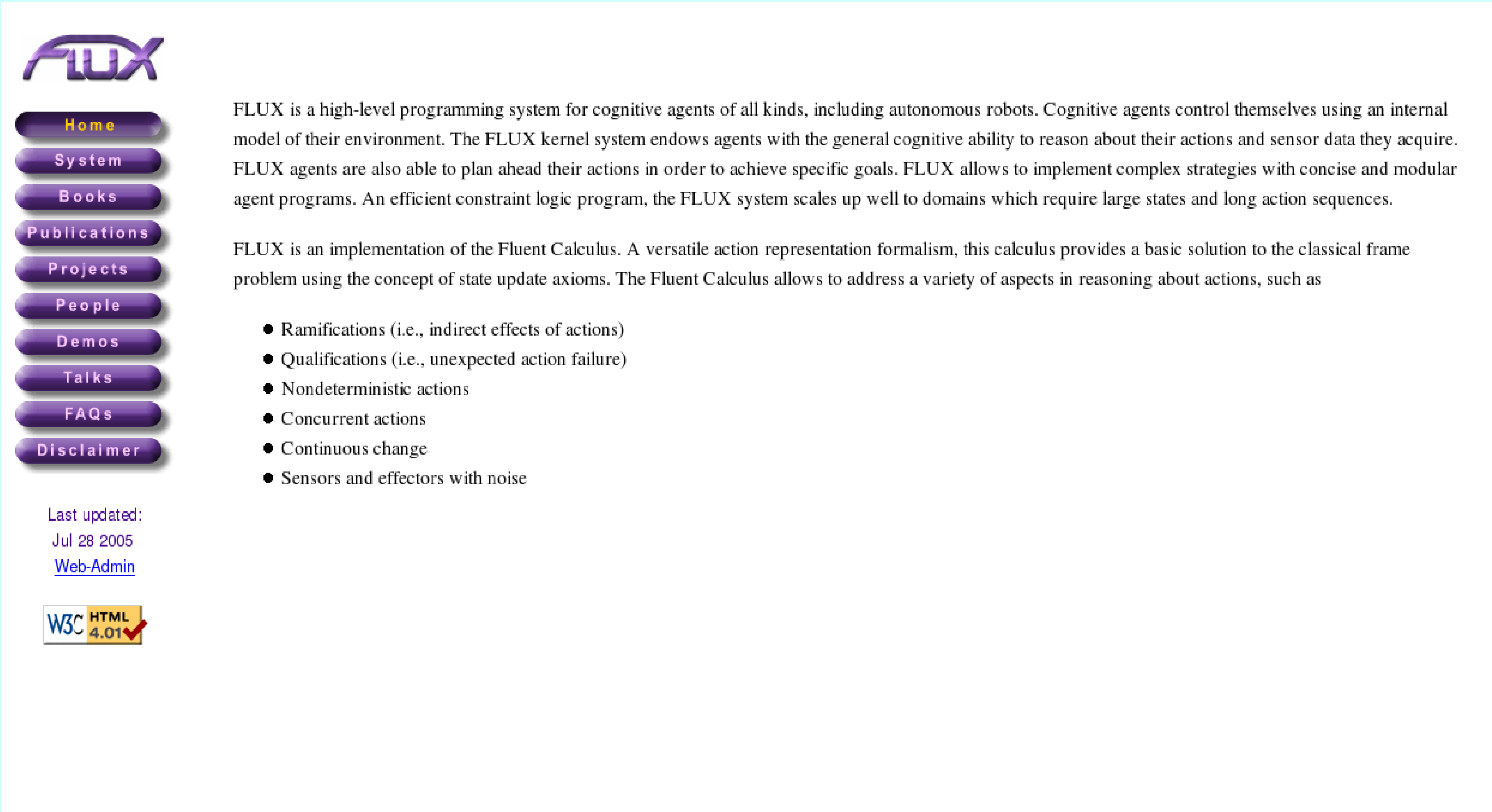
Runtime Comparison



Runtime Comparison



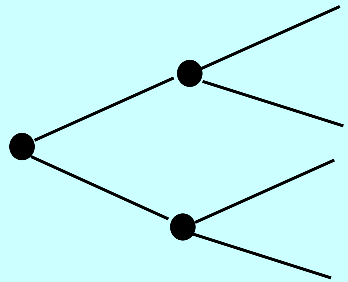
www.fluxagent.org



Chapter 5

Event Calculus and Other Formalisms

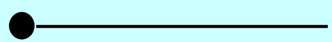
Linear Time Structure



... `Holds(f,Do(...,S`$_0$`))`

Situation Calculus

Fluent Calculus



... `Holds(f,t)`

Event Calculus

Event Calculus

`Holds(f,t)` `Happens(e,s,t)`	`f` holds at time `t` `e` happens between times `s` and `t`
`Initiates(e,f,s,t)` `Terminates(e,f,s,t)`	`e` initiates `f` between `s` and `t` `e` terminates `f` between `s` and `t`

Event Calculus: Basic Theories

- Narrative (using `Happens`)
- Observations (using `Holds`)
- Effect axioms (using `Initiates` / `Terminates`)
- Frame Problem solved by Circumscription

Extensions

- Concurrent events
- Nondeterministic events
- Planning by abduction
- Continuous processes

Other Agent Programming Systems

- A Behavior Language (ABL)
- Practical Reasoning System (PRS)
- SRI Procedural Agent Realization Kit (SPARK)

Systematic Assessment Methods

Meta Action Theories allow to systematically assess the range of applicability of specific calculi.

- Features-and-Fluents [Sandewall, 1994]
- Action Description Languages [Lifschitz etal, 1993f]

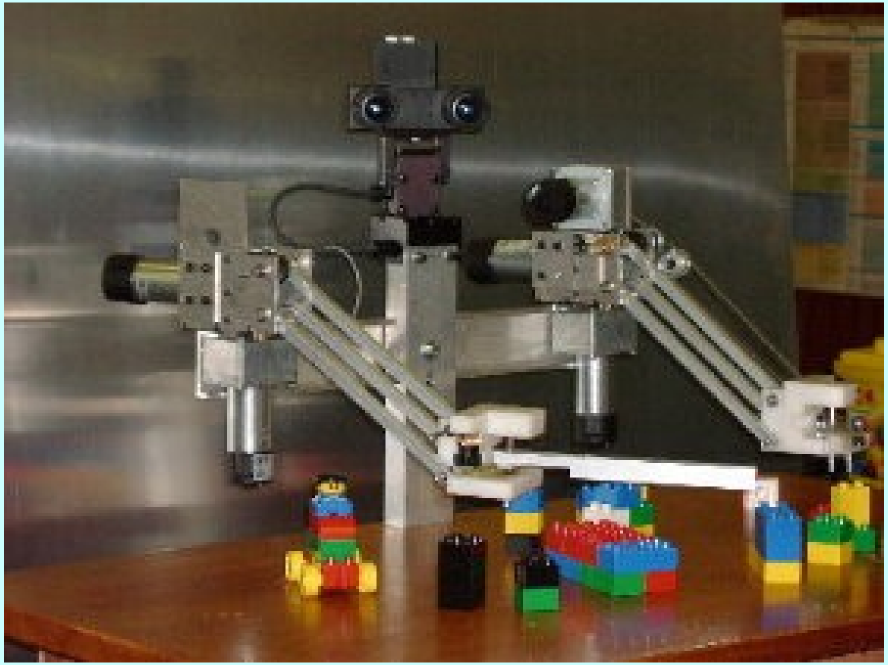
Chapter 6

Applications and Future Challenges

Application: Museum Tour Guide Robot

- *Experiences with an interactive museum tour-guide robot* [Wolfram Burgard etal, 1999]
- *GOLEX: Bridging the gap between logic (GOLOG) and a real robot* [Rainer Hähnel etal, 1998]

Application: Upper-Torso Humanoid Robot



Event Calculus for perception and cognition

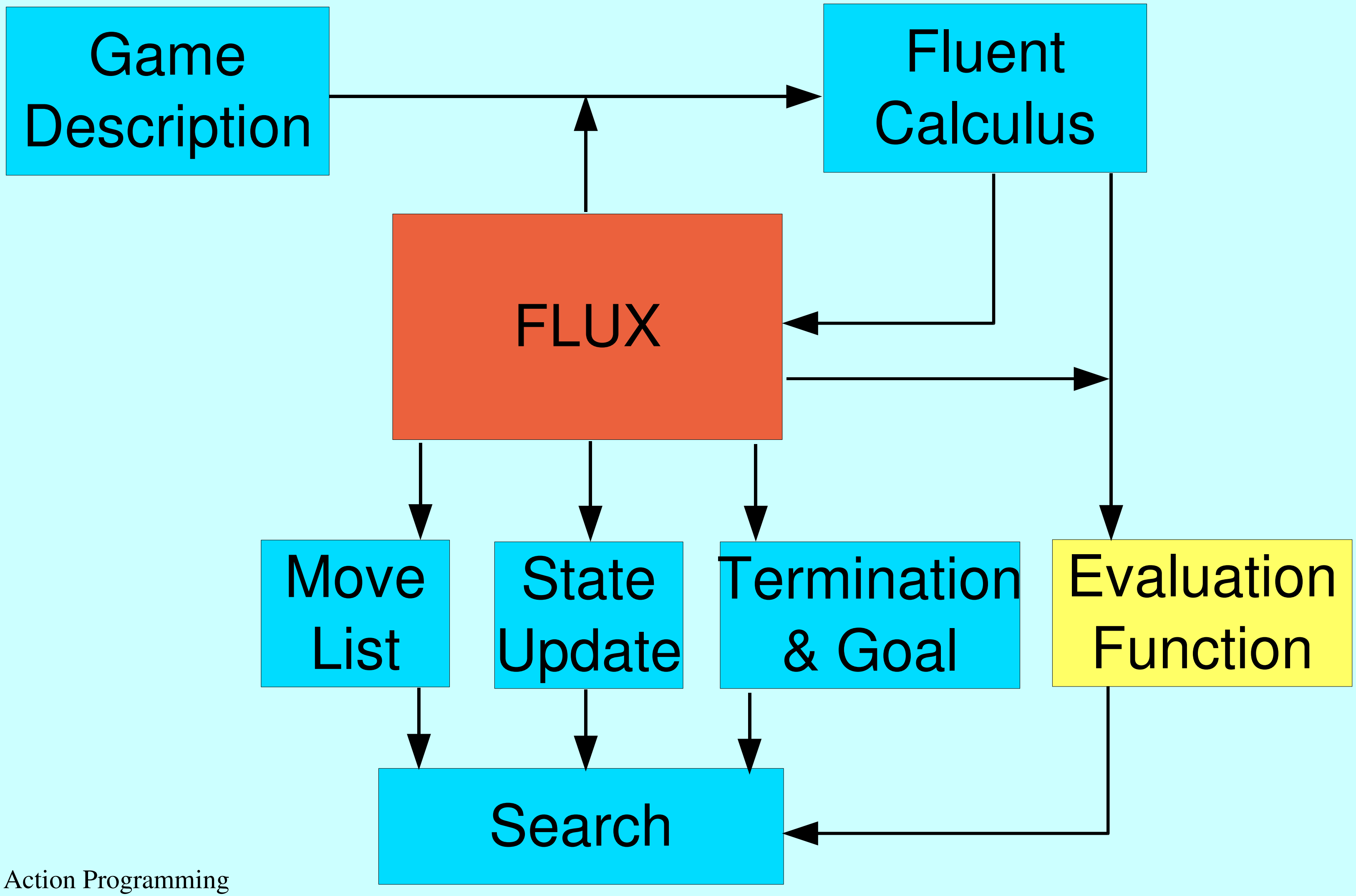
`www.iis.ee.ic.ac.uk/~mpsha/ludwig/`

Application: UAVs

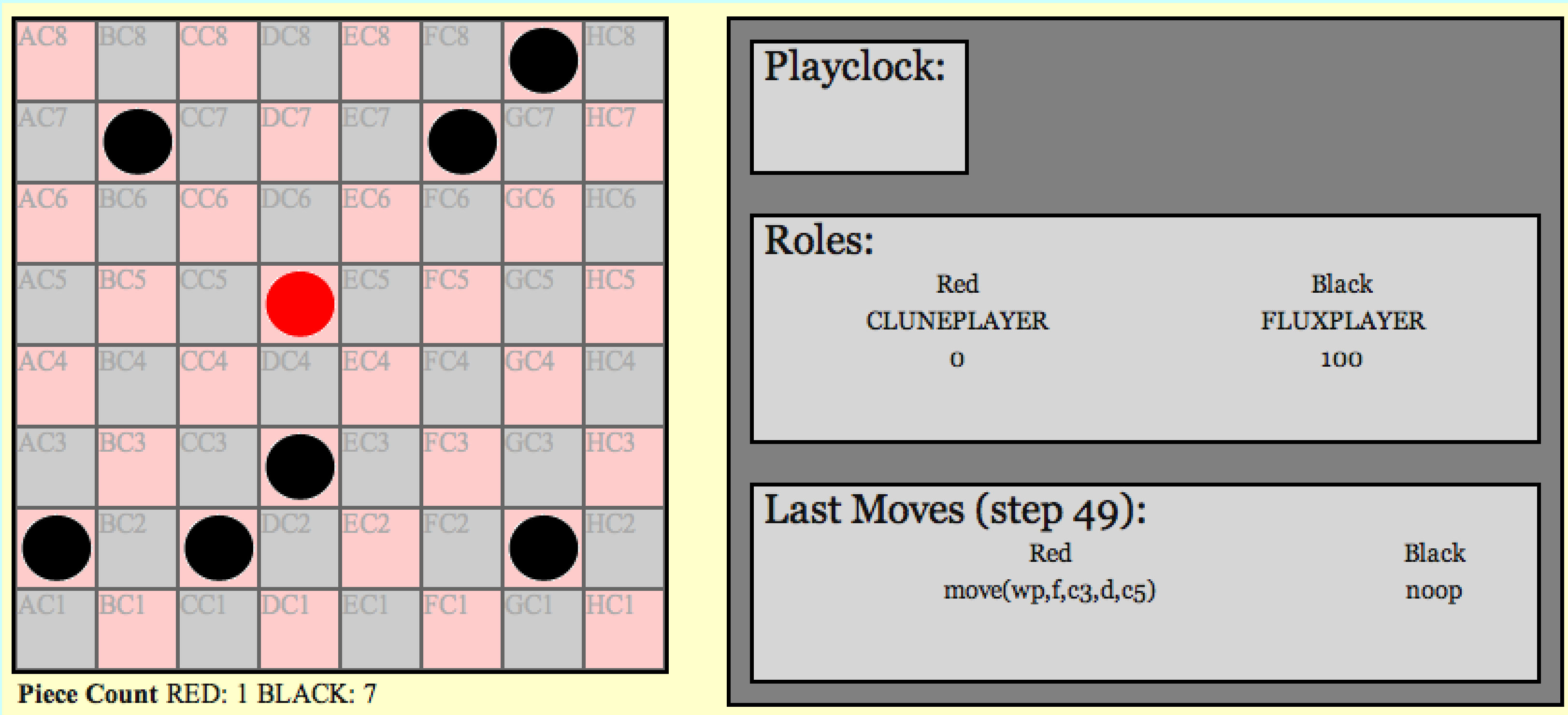
A UAV for traffic control monitors the ground and can autonomously track down and follow a car if necessary.

www.ida.liu.se/ext/witas/

Application: FLUXPLAYER



GGP World Championship 2006



Cylindrical Checkers: The Championship Final @AAAI-06

www.fluxagent.org

Future Challenge 1: Uncertainty

- Logic
  - Controlled uncertainty in logic via incomplete state descriptions (disjunction, ...)
  - Symbolic reasoning can deal with large state spaces thanks to abstraction and local inference
- Probability
  - Robot control in real-world environments requires probabilistic knowledge representation

- `P(z,s)` $\triangleq$ probability of `z` to be actual state in `s`

Challenge: A computational model for logic & probability!

Future Challenge 2: Symbol Grounding

- Symbols (like "Sandra's-coffee-mug") need to be grounded in actual perceptions of the real world
- In today's systems, the grounding of symbols is pre-defined
- Cognitive robots should ultimately be able to ground symbols autonomously

Challenge: Solve the Symbol Grounding Problem!

Recommended Literature

- Situation Calculus and GOLOG

  Raymond Reiter: *Logic in Action*. MIT Press 2001

- Fluent Calculus and FLUX

  Michael Thielscher: *Reasoning Robots*. Springer 2005

- Event Calculus

  Murray Shannahan: *Solving the Frame Problem*. MIT Press 1996

  Erik Mueller: *Commonsense Reasoning*. Morgan Kaufmann 2006