Computational Rationality

John Lloyd

Computer Sciences Laboratory
College of Engineering and Computer Science
The Australian National University

Acknowledgements

Many thanks to

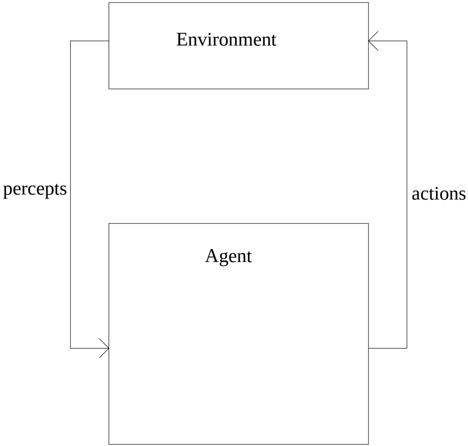
Joshua Cole, Matt Gray, Kee Siong Ng, and Tim Sears

Architectures for Rational Agents

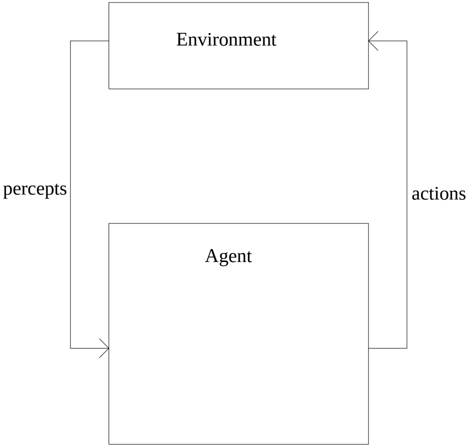
Applications to Recommender Systems and other Domains

Current and Future Work

Agents

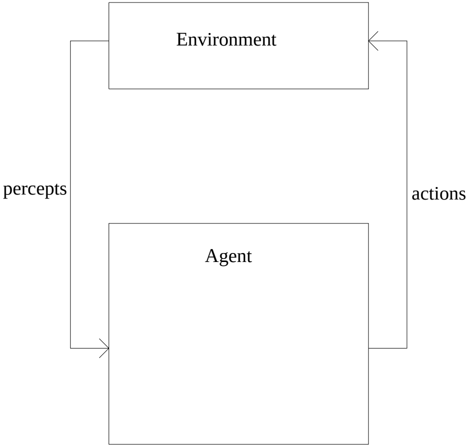


Agents



Agents are:

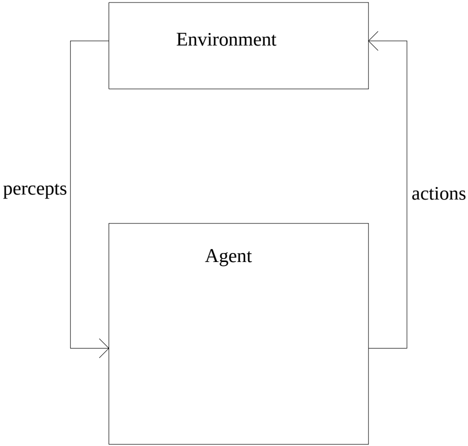
Agents



Agents are:

- Situated

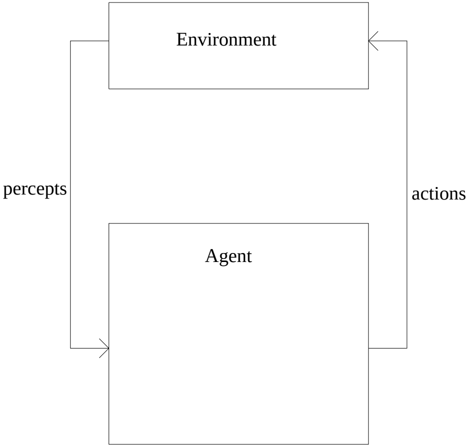
Agents



Agents are:

- Situated
- Autonomous

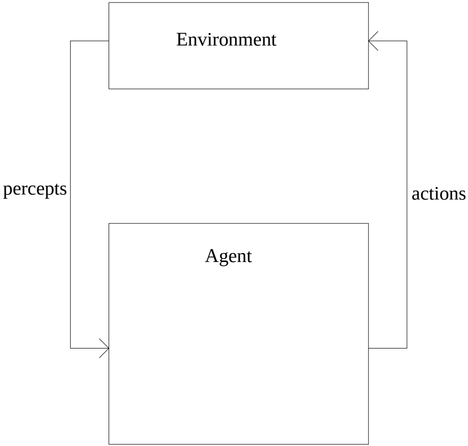
Agents



Agents are:

- Situated
- Autonomous
- Reactive

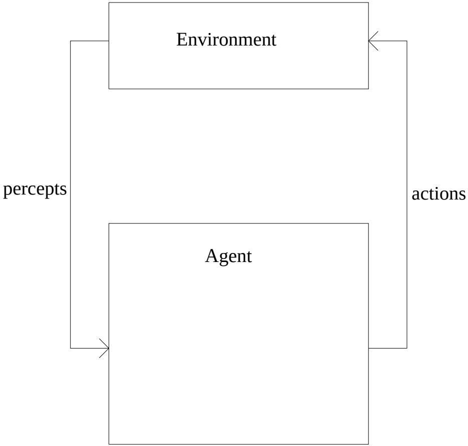
Agents



Agents are:

- Situated
- Autonomous
- Reactive
- Proactive

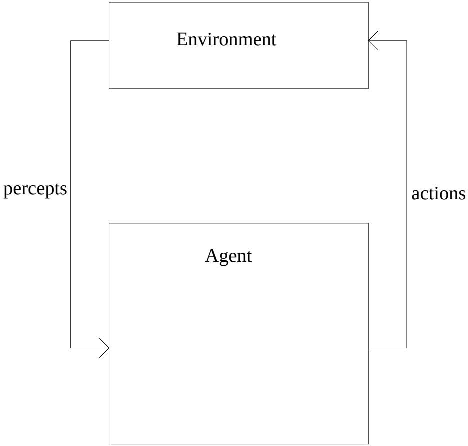
Agents



Agents are:

- Situated
- Autonomous
- Reactive
- Proactive
- Flexible

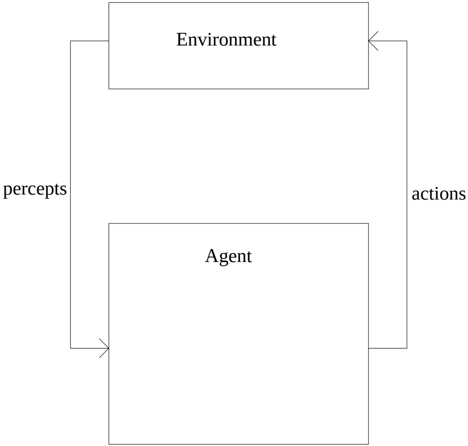
Agents



Agents are:

- Situated
- Autonomous
- Reactive
- Proactive
- Flexible
- Robust

Agents



Agents are:

- Situated
- Autonomous
- Reactive
- Proactive
- Flexible
- Robust
- Social

Rational Agents

"For each possible percept sequence, a rational agent should select an action that is expected to maximise its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has".

Russell & Norvig, "Artificial Intelligence: A Modern Approach"

An Architecture for Rational Agents

An Architecture for Rational Agents

- Scientific problem: The design of effective architectures for agent systems that perform complex tasks.

An Architecture for Rational Agents

- Scientific problem: The design of effective architectures for agent systems that perform complex tasks.
- Agents need:
  - *Probability* for uncertainty
  - *Logic* for knowledge representation
  - *Learning* for adaptivity

An Architecture for Rational Agents

- Scientific problem: The design of effective architectures for agent systems that perform complex tasks.
- Agents need:
  - *Probability* for uncertainty
  - *Logic* for knowledge representation
  - *Learning* for adaptivity
- The proposed architecture integrates logic, probability, and learning

(J.W. Lloyd and T.D. Sears, "An Architecture for Rational Agents", DALT 2005, LNAI 3904, 51-71, 2006)

An Architecture for Rational Agents

- Scientific problem: The design of effective architectures for agent systems that perform complex tasks.
- Agents need:
  - *Probability* for uncertainty
  - *Logic* for knowledge representation
  - *Learning* for adaptivity
- The proposed architecture integrates logic, probability, and learning

  (J.W. Lloyd and T.D. Sears, "An Architecture for Rational Agents", DALT 2005, LNAI 3904, 51-71, 2006)
- The rationality principle adopted is that of *maximum expected utility*.

Belief Bases for Agents

Belief Bases for Agents

- Belief bases are theories in a multi-modal, typed, higher-order logic

  (J.W. Lloyd, "Knowledge Representation and Reasoning in Modal Higher-order Logic", in prep., 2006)

Belief Bases for Agents

- Belief bases are theories in a multi-modal, typed, higher-order logic

  (J.W. Lloyd, "Knowledge Representation and Reasoning in Modal Higher-order Logic", in prep., 2006)

- There are epistemic ($\mathbf{B}_i$) and temporal ($\bullet$, $\blacklozenge$, $\blacksquare$, $\mathbf{S}$) modalities

Belief Bases for Agents

- Belief bases are theories in a multi-modal, typed, higher-order logic

  (J.W. Lloyd, "Knowledge Representation and Reasoning in Modal Higher-order Logic", in prep., 2006)

- There are epistemic ($\mathbf{B}_i$) and temporal ($\bullet$, $\blacklozenge$, $\blacksquare$, $\mathbf{S}$) modalities
- 'Higher-order' means that functions can have other functions as arguments.
  So, for example, a set can be represented by a predicate and manipulated by higher-order functions.
  Generally, much use is made in the architecture of higher-order functions

Reasoning (using Bach)

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

- Escher programming language

  (J.W. Lloyd, "Programming in an Integrated Functional and Logic Language", J. Functional and Logic Programming, 1999)

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

- Escher programming language

  (J.W. Lloyd, "Programming in an Integrated Functional and Logic Language", J. Functional and Logic Programming, 1999)

- Reasoning is computation and deduction combined

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

- Escher programming language

  (J.W. Lloyd, "Programming in an Integrated Functional and Logic Language", J. Functional and Logic Programming, 1999)

- Reasoning is computation and deduction combined
- Computation means 'finding the value' of a term

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

- Escher programming language

  (J.W. Lloyd, "Programming in an Integrated Functional and Logic Language", J. Functional and Logic Programming, 1999)

- Reasoning is computation and deduction combined
- Computation means 'finding the value' of a term
- Deduction means theorem proving

Reasoning (using Bach)

- Gödel programming language

  (P.M. Hill and J.W. Lloyd, "The Gödel Programming Language", MIT Press, 1994)

- Escher programming language

  (J.W. Lloyd, "Programming in an Integrated Functional and Logic Language", J. Functional and Logic Programming, 1999)

- Reasoning is computation and deduction combined
- Computation means 'finding the value' of a term
- Deduction means theorem proving
- Bach extends Escher mainly in that it also has a theorem prover and that its logic is modal

Belief Acquisition (using Alkemy)

Belief Acquisition (using Alkemy)

- Definitions of functions in belief bases can be acquired by an algorithm that includes both learning and conventional knowledge base update as special cases

Belief Acquisition (using Alkemy)

- Definitions of functions in belief bases can be acquired by an algorithm that includes both learning and conventional knowledge base update as special cases
- Whether the agent is learning or updating (or a combination of both) is controlled by
  - a *training* language (the language of the training examples), and
  - an *hypothesis* language (the language from which hypotheses are formed)

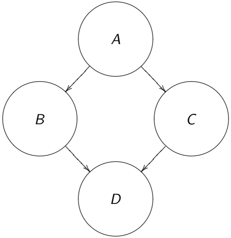
Belief Acquisition (using Alkemy)

- Definitions of functions in belief bases can be acquired by an algorithm that includes both learning and conventional knowledge base update as special cases
- Whether the agent is learning or updating (or a combination of both) is controlled by
  - a *training* language (the language of the training examples), and
  - an *hypothesis* language (the language from which hypotheses are formed)
- All this is (being) implemented by Alkemy

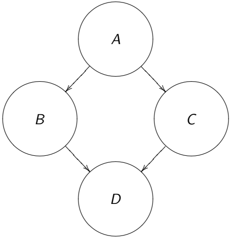
(J.W. Lloyd, "Logic for Learning", Springer, Cognitive Technologies, 2003)

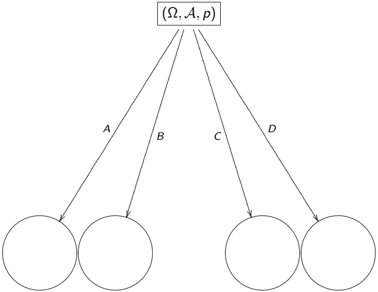
(J.J. Cole, J.W. Lloyd, and K.S. Ng, "An Approach to Belief Acquisition", in prep., 2006)

Graphical Models



Graphical Models





Sample space is explicit
Random variables are functions

Underlying Dynamics for Agents

Let S denote the set of states and A the set of actions.

The probability space of interest here is $S \times A \times S$ which has some probability measure p defined on it.

Thus $p(s, a, s')$ is the probability that if action a is applied to state s then state s' is reached.

By conditioning on states and actions, for each state $s \in S$ and action $a \in A$, a transition probability distribution $p(\cdot \mid s, a)$ is obtained.

$$
\begin{tikzcd}
 & S \times A \times S \arrow[dl, "initial"'] \arrow[d, "action"] \arrow[dr, "final"] & \\
S & A & S
\end{tikzcd}
$$

Evidence and Result Variables

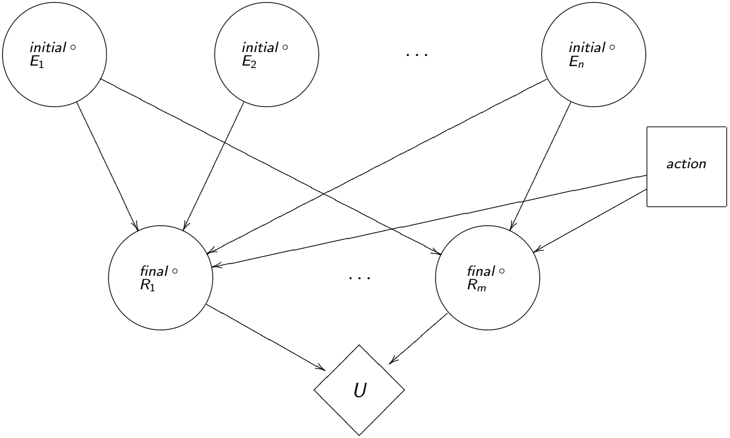
$$
\begin{tikzcd}
 & & & S \times A \times S \arrow[dll, "initial"'] \arrow[dd, "action"'] \arrow[drr, "final"] & & & \\
 & S \arrow[dl, "E_1"'] \arrow[dr, "E_n"] & & & & S \arrow[dl, "R_1"'] \arrow[dr, "R_n"] & \\
V_1 & \cdots & V_n & A & W_1 & \cdots & W_m
\end{tikzcd}
$$

The evidence variables $E_i (i = 1, \dots, n)$ are chosen so as to assist the selection of a good action.

The result variables $R_j (j = 1, \dots, m)$ are chosen so as to provide a good evaluation of the resulting state.

Definition A *belief base* is a theory consisting of the definitions of the evidence variables and (some of) the result variables.

Influence Diagram



Conditioned Influence Diagram

		$final \circ (R_1, \ldots, R_m)$		
$initial \circ (E_1, \ldots, E_n)$	$action$	$(w_{1,1}, \ldots, w_{m,1})$	$\ldots$	$(w_{1,l_1}, \ldots, w_{m,l_m})$
$(v_{1,1}, \ldots v_{1,n})$	a_1	$p_{1,1}$	$\ldots$	$p_{1,l_1 \cdots l_m}$
$(v_{2,1}, \ldots v_{2,n})$	a_2	$p_{2,1}$	$\ldots$	$p_{2,l_1 \cdots l_m}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$(v_{k,1}, \ldots v_{k,n})$	a_k	$p_{k,1}$	$\ldots$	$p_{k,l_1 \cdots l_m}$
		u_1	$\ldots$	$u_{l_1 \cdots l_m}$

Applications

- Trading agents (TAC'05 competition at IJCAI'05)
- Recommender systems
- Poker

TV Recommender

These ideas are illustrated with a TV recommender
(part of a larger multi-agent system)

(J.J. Cole, M.J. Gray, J.W. Lloyd and K.S. Ng, "Personalisation for User Agents", AAMAS 05, 603-610, 2005)

Emphasis is on adaptation to the user's interests and preferences
Personalisation component that could be added to TIVO

A state is a pair consisting of an occurrence and a status
An occurrence is a triple (date, time, channel)
Status is either *Unknown* or *Yes* or *No*

Actions are *RecommendYes* and *RecommendNo*
(turns *Unknown* into *Yes* or *No*)

Evidence and Result Variables for the TV Recommender

$$
\begin{tikzcd}
 & & & \mathit{State} \times \mathit{Action} \times \mathit{State} \arrow[dll, "\mathit{initial}"'] \arrow[dd, "\mathit{action}"'] \arrow[dr, "\mathit{final}"] & & \\
 & \mathit{State} \arrow[dl, "\mathit{ultp}"'] \arrow[d, "\mathit{udiary}"'] \arrow[dr, "\mathit{uaccept}"] & & & \mathit{State} \arrow[d, "\mathit{user}"'] \arrow[dr, "\mathit{recomm}"] & \\
\Omega & \Omega & \Omega & \mathit{Action} & \Omega & \mathit{Status}
\end{tikzcd}
$$

ultp: whether or not the user likes the program having that occurrence
udiary: whether or not the user's diary is free at that time
uaccept: whether or not the user is willing to watch television at that time

user: whether user intends to watch the program having that occurrence
recomm: the recommendation of the agent

Belief Base for the TV Recommender

$$\mathbf{B}_t\ ((tv\_guide\ ((20, 7, 2004), (20, 30), ABC)) =_{Program}$$
$$(\text{“}\textit{The Bill}\text{”}, \text{“”}, 50, [Drama], M,$$
$$\text{“}\textit{Sun Hill continues to work at} \ldots \textit{smuggling operation}\text{”})).$$

This fact states that the program on 20 July 2004 at 8.30pm on channel ABC has title “The Bill”, no subtitle, a duration of 50 minutes, genre drama, a classification for mature audiences, and synopsis “Sun Hill continues to work at breaking the people smuggling operation”.

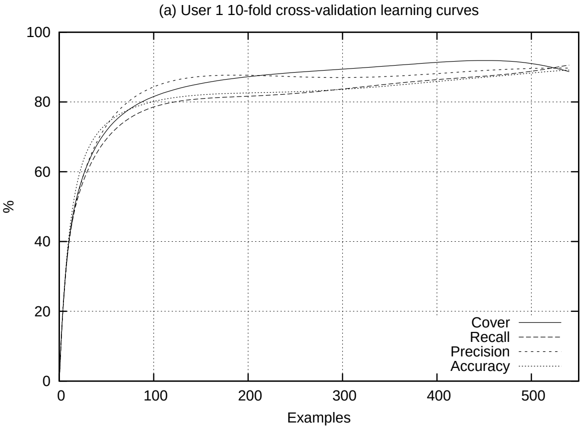
The TV guide (obtained from the Web) contains information about ∼4000 programs

Belief Base for the TV Recommender 2

$$likes\_tv\_program : Program \rightarrow \Omega$$

$$\mathbf{B}_t \mathbf{B}_u \ \forall_{Program} x.$$
$$((likes\_tv\_program\ x) =_{\Omega}$$
$$\quad if\ (proj_{Title} \circ (=_{Title}\ \text{“NFL Football”})\ x)\ then\ \top$$
$$\quad else\ if\ (proj_{(List\ Genre)} \circ (listExists_1\ genre \circ (< 0))\ x)\ then\ \bot$$
$$\quad else\ if\ (proj_{Title} \circ StringToText \circ (listExists_1\ (=_{String}\ \text{“sport”}))\ x)\ then\ \top$$
$$\quad else\ if\ (proj_{(List\ Genre)} \circ (listExists_1\ (=_{Genre}\ Current\_Affairs))\ x)\ then\ \bot$$
$$\vdots$$
$$\quad else\ \bot).$$

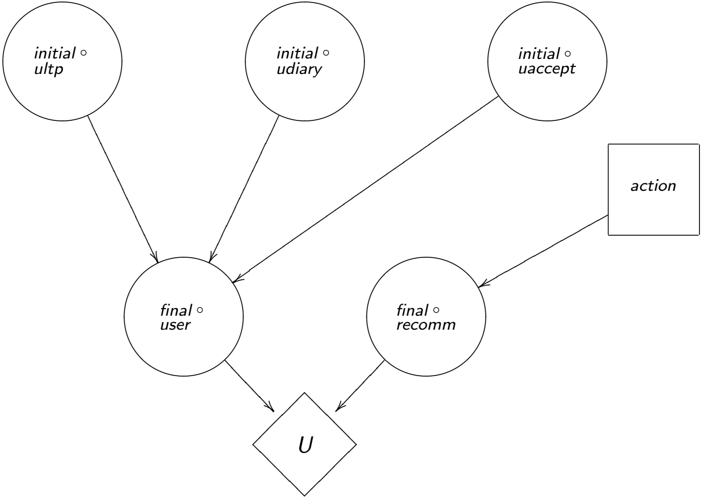
Learning Curve



Belief Base for the TV Recommender 3

$$
\begin{array}{l}
tv\_time\_acceptable : Date \times Time \times Time \rightarrow \Omega \\
\mathbf{B}_t \mathbf{B}_u \; \forall_{Date} d. \; \forall_{Time} t. \; \forall_{Time} t'. \\
\quad ((tv\_time\_acceptable\ (d, t, t')) =_{\Omega} \\
\quad\quad if\ (weekday\ d) \wedge ((proj_{Hour}\ t) \geq 20) \wedge ((proj_{Hour}\ t') \leq 23)\ then\ \top \\
\quad\quad else\ if\ \neg(weekday\ d) \wedge ((proj_{Hour}\ t) \geq 12) \wedge ((proj_{Hour}\ t') \leq 25)\ then\ \top \\
\quad\quad else\ \bot).
\end{array}
$$

Influence Diagram for the TV Recommender



Conditioned Influence Diagram for the TV Recommender

$initial \circ (ultp, udiary, uavail)$	$action$	$final \circ (user, recomm)$: $(\top, Yes)$	$(\top, No)$	$(\bot, Yes)$	$(\bot, No)$	
$(\top, \top, \top)$	$RecYes$	0.8	0.0	0.2	0.0	0.88
$(\top, \top, \top)$	$RecNo$	0.0	0.8	0.0	0.2	0.20
$(\top, \top, \bot)$	$RecYes$	0.4	0.0	0.6	0.0	0.64
$(\top, \top, \bot)$	$RecNo$	0.0	0.4	0.0	0.6	0.60
$(\top, \bot, \top)$	$RecYes$	0.0	0.0	1.0	0.0	0.40
$(\top, \bot, \top)$	$RecNo$	0.0	0.0	0.0	1.0	1.00
$(\top, \bot, \bot)$	$RecYes$	0.0	0.0	1.0	0.0	0.40
$(\top, \bot, \bot)$	$RecNo$	0.0	0.0	0.0	1.0	1.00
$(\bot, \top, \top)$	$RecYes$	0.1	0.0	0.9	0.0	0.46
$(\bot, \top, \top)$	$RecNo$	0.0	0.1	0.0	0.9	0.90
...	...	...	...	...	...	...
		1	0	0.4	1	

Policy for the TV Recommender

$$
\begin{aligned}
&\mathit{policy} : \mathit{State} \rightarrow \mathit{Action}\\
&\mathbf{B}_t\ \forall_{\mathit{State}} s.\\
&\qquad ((\mathit{policy}\ s) =_{\mathit{Action}}\\
&\qquad\quad \mathit{if}\ (((\mathit{ultp}\ s) =_{\Omega} \top) \wedge ((\mathit{udiary}\ s) =_{\Omega} \top))\ \mathit{then}\ \mathit{RecYes}\\
&\qquad\quad \mathit{else}\ \mathit{RecNo}).
\end{aligned}
$$

Interaction of Epistemic and Temporal Modalities

Interaction of Epistemic and Temporal Modalities

The most relevant temporal modalities are $\bullet$ ('last'), $\blacksquare$ ('always in the past'), $\blacklozenge$ ('sometime in the past'), and $\mathbf{S}$ ('since').

Interaction of Epistemic and Temporal Modalities

The most relevant temporal modalities are $\bullet$ ('last'), $\blacksquare$ ('always in the past'), $\blacklozenge$ ('sometime in the past'), and $\mathbf{S}$ ('since').

Part of a belief base:

$$\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma A\ \mathit{then}\ \top\ \mathit{else}\ \mathit{if}\ x =_\sigma B\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$

$$\bullet\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma A\ \mathit{then}\ \top\ \mathit{else}\ \mathit{if}\ x =_\sigma C\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$

$$\bullet^2\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma C\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$

$$\bullet^3\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega \bot)$$

Interaction of Epistemic and Temporal Modalities

The most relevant temporal modalities are $\bullet$ ('last'), $\blacksquare$ ('always in the past'), $\blacklozenge$ ('sometime in the past'), and $\mathbf{S}$ ('since').

Part of a belief base:

$$\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma A\ \mathit{then}\ \top\ \mathit{else\ if}\ x =_\sigma B\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$
$$\bullet\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma A\ \mathit{then}\ \top\ \mathit{else\ if}\ x =_\sigma C\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$
$$\bullet^2\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega\ \mathit{if}\ x =_\sigma C\ \mathit{then}\ \top\ \mathit{else}\ \bot)$$
$$\bullet^3\mathbf{B}\ \forall_\sigma x.((p\ x) =_\Omega \bot)$$

The following axiom models an agent with a form of perfect recall:

$$\bullet\mathbf{B}_i\varphi \longrightarrow \mathbf{B}_i\bullet\varphi$$

Final Remarks

- Currently working on:
  - making good use of the epistemic and temporal modalities
  - implementation of Bach/Alkemy
  - application to poker

Final Remarks

- Currently working on:
  - making good use of the epistemic and temporal modalities
  - implementation of Bach/Alkemy
  - application to poker
- What comes next?
  - a general study of induction principles
  - putting the whole approach together
  - applications which show that the approach is fully convincing