Rational Agents

Paolo Turrini

Department of Computing, Imperial College London

Introduction to Artificial Intelligence
2nd Part
What you have seen

You have seen procedures for computational problem-solving:

- searching
- learning
- planning
What we will be looking at

An agent, a mathematical entity acting in a simple world
What we will be looking at

An **agent**, a mathematical entity acting in a simple world

- Able to reason about the world around
  - true facts (knowledge)
  - plausible facts (beliefs)
What we will be looking at

An agent, a mathematical entity acting in a simple world

- Able to reason about the world around
  - true facts (knowledge)
  - plausible facts (beliefs)

- Able to take decisions under uncertainty
  - Imperfect and incomplete information
  - Quantifying uncertainty, attaching probabilities
  - Going for uncertain outcomes, calculating expected utility
What we will be looking at

An agent, a mathematical entity acting in a simple world

- Able to reason about the world around
  - true facts (knowledge)
  - plausible facts (beliefs)

- Able to take decisions under uncertainty
  - Imperfect and incomplete information
  - Quantifying uncertainty, attaching probabilities
  - Going for uncertain outcomes, calculating expected utility

- Able to update his (or her) beliefs when confronted with new information (learning)
What is rationality?

"A person’s behavior is rational if it is in his best interests, given his information"

Robert J. Aumann
Nobel Prize Winner
Economics
What is rationality?

"A person’s behavior is rational if it is in his best interests, given his information"

Agents (not only humans) can be rational!

Robert J. Aumann
Nobel Prize Winner
Economics
The lectures one by one

- Logical Agents I
- Logical Agents II
- An Uncertain World
- Making Sense of Uncertainty
- Making (Good) Decisions
- Making Good Decisions in time
- Learning from Experience I
- Learning from Experience II
Logical Agents I
The main reference

Stuart Russell and Peter Norvig
Artificial Intelligence: a modern approach
Chapters 7-9
The Wumpus World

- Stench
- Breeze
- PIT
- Gold
- START
Agents

**Sensors**  Breeze, Glitter, Smell

**Actuators**  Turn L/R, Go, Grab, Release, Shoot, Climb

**Rewards**  1000 escaping with gold, -1000 dying, -10 using arrow, -1 walking

**Environment**
- Squares adjacent to Wumpus are smelly
- Squares adjacent to pit are breezy
- Glitter iff gold is in the same square
- Shooting kills Wumpus if you are facing it
- Shooting uses up the only arrow
- Grabbing picks up gold if in same square
- Releasing drops the gold in same square
Knowledge base

- A set of sentences representing what the agent thinks about the world.
  - "I am in [2,1]"
  - "I am out of arrows"
  - "I smell Wumpus"
  - "I’d better not go forward"

- We interpret it as what the agent *knows*, but it can very well work for what the agent *believes*. 
Updating the knowledge base

- What we TELL the knowledge base
- What we ASK the knowledge base

```plaintext
function KB-Agent (percept) returns an action
    static: KB, a knowledge base
    t, a counter, initially 0, indicating time
    Tell(KB, Make-Percept-Sentence(percept, t))
    action ← Ask(KB, Make-Action-Query(t))
    Tell(KB, Make-Action-Sentence(action, t))
    t ← t + 1
    return action
```
Rational explorations

The starting state...
Rational explorations

- and what we know.
Rational explorations

- B stands for Breeze
Rational explorations

- Where is the pit?
- We are ruling out one square!
Rational explorations

- S stands for smell
- What do we know?
Rational explorations

- Logic is the key
Rational explorations

- The further we go the more we know
Rational explorations

- The further we go the more we know
Rational explorations

- Gold!
Rational explorations

- We know the way out
- Game over
Reasoning in the Wumpus World

Let $P_{i,j}$ be true if there is a pit in $[i,j]$. Let $B_{i,j}$ be true if there is a breeze in $[i,j]$. 
Reasoning in the Wumpus World

Let $P_{i,j}$ be true if there is a pit in $[i,j]$. Let $B_{i,j}$ be true if there is a breeze in $[i,j]$.

$\neg P_{1,1}$
$\neg B_{1,1}$
$B_{2,1}$
Reasoning in the Wumpus World

Let $P_{i,j}$ be true if there is a pit in $[i,j]$.
Let $B_{i,j}$ be true if there is a breeze in $[i,j]$.

$\neg P_{1,1}$
$\neg B_{1,1}$
$B_{2,1}$

“Pits cause breezes in adjacent squares”
Reasoning in the Wumpus World

Let $P_{i,j}$ be true if there is a pit in $[i,j]$.
Let $B_{i,j}$ be true if there is a breeze in $[i,j]$.

$\neg P_{1,1}$

$\neg B_{1,1}$

$B_{2,1}$

“Pits cause breezes in adjacent squares”

$B_{1,1} \iff (P_{1,2} \lor P_{2,1})$

$B_{2,1} \iff (P_{1,1} \lor P_{2,2} \lor P_{3,1})$

“A square is breezy if and only if there is an adjacent pit”
Expressivity: at what cost?

- OK if we were only dealing with finite objects
- But even then we would have to enumerate all the possibilities;
Expressivity: at what cost?

- OK if we were only dealing with finite objects
- But even then we would have to enumerate all the possibilities;

Propositional Logic lacks expressive power
First order logic

- Massive increase of expressivity
- But there are costs, e.g., decidability
- We will see how to exploit the gains while limiting the costs
We can encode the KB at each particular time point using FOL

function $\text{KB-Agent}(\text{percept})$ returns an action

static: $KB$, a knowledge base

$t$, a counter, initially 0, indicating time

$\text{Tell}(KB, \text{Make-Percept-Sentence}(\text{percept}, t))$

$\text{action} \leftarrow \text{Ask}(KB, \text{Make-Action-Query}(t))$

$\text{Tell}(KB, \text{Make-Action-Sentence}(\text{action}, t))$

$t \leftarrow t + 1$

return $\text{action}$
You already know how to describe the WW in first order logic.
You already know how to describe the WW in first order logic

- Percept (at given time), e.g.,
  \( \text{Percept}([\text{Stench}, \text{Breeze}, \text{Glitter}], 5) \) or
  \( \text{Percept}([\text{None}, \text{Breeze}, \text{None}], 3) \)
You already know how to describe the WW in first order logic

- Percept (at given time), e.g.,
  \[ \text{Percept}([\text{Stench}, \text{Breeze}, \text{Glitter}], 5) \] or
  \[ \text{Percept}([\text{None}, \text{Breeze}, \text{None}], 3) \]
- Starting Knowledge Base, e.g., \( \neg \text{AtGold}(0) \)
You already know how to describe the WW in first order logic

- Percept (at given time), e.g.,
  \( \text{Percept}([\text{Stench}, \text{Breeze}, \text{Glitter}], 5) \) or
  \( \text{Percept}([\text{None}, \text{Breeze}, \text{None}], 3) \)

- Starting Knowledge Base, e.g., \( \neg \text{AtGold}(0) \)

- Axioms to generate new knowledge from percepts, e.g.,
  \( \forall s, b, t \ \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t) \)
You already know how to describe the WW in first order logic

- Percept (at given time), e.g.,
  \[ \text{Percept}([\text{Stench}, \text{Breeze}, \text{Glitter}], 5) \text{ or } \text{Percept}([\text{None}, \text{Breeze}, \text{None}], 3) \]
- Starting Knowledge Base, e.g., \( \neg \text{AtGold}(0) \)
- Axioms to generate new knowledge from percepts, e.g.,
  \( \forall s, b, t \ \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t) \)
- Axioms to generate actions (plans) from KB, e.g.,
  \( \forall t \ \text{AtGold}(t) \land \neg \text{Holding}(\text{Gold}, t) \Rightarrow \text{Action}(\text{Grab}, t) \)
You already know how to describe the WW in first order logic

- Percept (at given time), e.g.,
  \[\text{Percept}([\text{Stench}, \text{Breeze}, \text{Glitter}], 5)\] or
  \[\text{Percept}([\text{None}, \text{Breeze}, \text{None}], 3)\]
- Starting Knowledge Base, e.g., \(\neg \text{AtGold}(0)\)
- Axioms to generate new knowledge from percepts, e.g.,
  \(\forall s, b, t \ \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t)\)
- Axioms to generate actions (plans) from KB, e.g.,
  \(\forall t \ \text{AtGold}(t) \land \neg \text{Holding} (\text{Gold}, t) \Rightarrow \text{Action} (\text{Grab}, t)\)
- Axioms from knowledge to knowledge, e.g.,
  \(\forall t \ \text{AtGold}(t) \land \text{Action} (\text{Grab}, t) \Rightarrow \text{Holding} (\text{Gold}, t + 1)\)
Describing the world

Perception \( \forall s, b, t \ Percept([s, b, Glitter], t) \Rightarrow AtGold(t) \)
Describing the world

Perception \( \forall s, b, t \ Percept([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t) \)

Location \( \text{At}(\text{Agent}, s, t) \)
Describing the world

Perception \( \forall s, b, t \ Percept([s, b, Glitter], t) \Rightarrow AtGold(t) \)

Location \( At(Agent, s, t) \)

Decision-making \( \forall t \ AtGold(t) \Rightarrow Action(Grab, t) \)
Describing the world

Perception \( \forall s, b, t \; \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t) \)

Location \( \text{At}(\text{Agent}, s, t) \)

Decision-making \( \forall t \; \text{AtGold}(t) \Rightarrow \text{Action}(\text{Grab}, t) \)

Internal reflection \( \forall t \; \text{AtGold}(t) \land \neg \text{Holding}(\text{Gold}, t) \Rightarrow \text{Action}(\text{Grab}, t) \), do we have gold already? (notice we cannot observe if we are holding gold, we need to track it)
Describing the world

Adjacent squares

\[ \forall x, y, a, b \quad \text{Adjacent}([x, y], [a, b]) \iff \\
\quad (x = a \land (y = b - 1 \lor y = b + 1)) \lor \\
\quad (y = b \land (x = a - 1 \lor x = a + 1)) \]

“A square is breezy if and only if there is an adjacent pit”

\[ \forall s, \text{Breezy}(s) \iff \exists r \ (\text{Adjacent}(r, s) \land \text{Pit}(r)) \]
Describing the world

Adjacent squares

\[ \forall x, y, a, b \quad \text{Adjacent}([x, y], [a, b]) \iff (x = a \land (y = b - 1 \lor y = b + 1)) \lor (y = b \land (x = a - 1 \lor x = a + 1)) \]

“A square is breezy if and only if there is an adjacent pit”

\[ \forall s, \text{Breezy}(s) \iff \exists r \ (\text{Adjacent}(r, s) \land \text{Pit}(r)) \]
Describing the world

- We can go on and describe plans, causal rules, etc.
- But let’s do some reasoning now
Facts and Knowledge Bases

"Richard the Lionheart is a king"
Facts and Knowledge Bases

"Joffrey Baratheon is a king"
Telling and Asking

Tell($KB, King(Joffrey)$)
Telling and Asking

Tell(KB, King(Joffrey))

Tell(KB, Person(Jaime))
Telling and Asking

\[\text{Tell}(KB, \text{King}(\text{Joffrey}))\]

\[\text{Tell}(KB, \text{Person}(\text{Jaime}))\]

\[\text{Tell}(KB, \forall x \, \text{King}(x) \Rightarrow \text{Person}(x))\]
Telling and Asking

\[
\text{Tell}(KB, \text{King}(\text{Joffrey})) \\
\text{Tell}(KB, \text{Person}(\text{Jaime})) \\
\text{Tell}(KB, \forall x \text{ King}(x) \Rightarrow \text{Person}(x)) \\
\text{Ask}(KB, \exists x \text{Person}(x)) \text{ is there a person?}
\]
Telling and Asking

Tell(KB, King(Joffrey))
Tell(KB, Person(Jaime))
Tell(KB, ∀ x King(x) ⇒ Person(x))
Ask(KB, ∃ x Person(x)) is there a person?
Askvar(KB, Person(x)) who is a person?
Telling and Asking

Tell($KB, King(Joffrey)$)

Tell($KB, Person(Jaime)$)

Tell($KB, \forall x \ King(x) \Rightarrow Person(x)$)

Ask($KB, \exists x Person(x)$) is there a person?

Askvar($KB, Person(x)$) who is a person?

Askvar returns a list of substitutions: \{x/Joffrey\}, \{x/Jaime\}
Substitutions

Definition

Given a sentence $S$ and a substitution $\sigma$,
Substitutions

Definition

Given a sentence $S$ and a substitution $\sigma$, $S\sigma$ denotes the result of plugging $\sigma$ into $S$; e.g.,
Substitutions

Definition

Given a sentence $S$ and a substitution $\sigma$,

$S\sigma$ denotes the result of plugging $\sigma$ into $S$; e.g.,

$S = Smarter(x, y)$
Substitutions

Definition

Given a sentence $S$ and a substitution $\sigma$, $S\sigma$ denotes the result of plugging $\sigma$ into $S$; e.g.,

$S = Smarter(x, y)$

$\sigma = \{x/\text{Tyrion}, y/\text{Joffrey}\}$
Substitutions

**Definition**

Given a sentence $S$ and a substitution $\sigma$,

$S\sigma$ denotes the result of plugging $\sigma$ into $S$; e.g.,

$S = Smarter(x, y)$

$\sigma = \{x/\text{Tyrion}, y/\text{Joffrey}\}$

$S\sigma = Smarter(\text{Tyrion}, \text{Joffrey})$
Substitutions

Definition

Given a sentence $S$ and a substitution $\sigma$, $S\sigma$ denotes the result of plugging $\sigma$ into $S$; e.g.,

$S = Smarter(x, y)$

$\sigma = \{ x / Tyrion, y / Joffrey \}$

$S\sigma = Smarter(Tyrion, Joffrey)$

$Askvar(KB, S)$ returns some/all $\sigma$ such that $KB \models S\sigma$
Unification

\[ \forall x \ (\text{King}(x) \land \text{Greedy}(x)) \Rightarrow \text{Evil}(x) \]

\[ \text{King(Joffrey)} \]

\[ \forall y \ (\text{Greedy}(y)) \]
Unification

\[ \forall x \ King(x) \land Greedy(x) \Rightarrow Evil(x) \]

King(Joffrey)

\[ \forall y \ Greedy(y) \]

We can get the inference immediately if we can find a substitution matching the premises of the implication to the known facts.
Unification

\[ \forall x \ King(x) \land Greedy(x) \Rightarrow Evil(x) \]

King(Joffrey)

\[ \forall y \ Greedy(y) \]

We can get the inference immediately if we can find a substitution matching the premises of the implication to the known facts.

\[ \theta = \{x/Joffrey, y/Joffrey\} \text{ works} \]
Unification

\( \text{UNIFY}(\alpha, \beta) \) returns \( \theta \) if \( \alpha \theta = \beta \theta \)
## Unification

\[
\text{UNIFY}(\alpha, \beta) \text{ returns } \theta \text{ if } \alpha \theta = \beta \theta
\]

<table>
<thead>
<tr>
<th>(p)</th>
<th>(q)</th>
<th>(\theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Knows(Joffrey, x)</code></td>
<td><code>Knows(Joffrey, Sansa)</code></td>
<td></td>
</tr>
<tr>
<td><code>Knows(Joffrey, x)</code></td>
<td><code>Knows(y, Sansa)</code></td>
<td></td>
</tr>
<tr>
<td><code>Knows(Joffrey, x)</code></td>
<td><code>Knows(y, Mother(Joffrey))</code></td>
<td></td>
</tr>
<tr>
<td><code>Knows(Joffrey, x)</code></td>
<td><code>Knows(x, Sansa)</code></td>
<td></td>
</tr>
</tbody>
</table>
Unification

\[
\text{UNIFY}(\alpha, \beta) \text{ returns } \theta \text{ if } \alpha\theta = \beta\theta
\]

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( \theta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knows(Joffrey, ( x ))</td>
<td>Knows(Joffrey, Sansa)</td>
<td>{x/Sansa}</td>
</tr>
<tr>
<td>Knows(Joffrey, ( x ))</td>
<td>Knows(( y ), Sansa)</td>
<td></td>
</tr>
<tr>
<td>Knows(Joffrey, ( x ))</td>
<td>Knows(( y ), Mother(Joffrey))</td>
<td></td>
</tr>
<tr>
<td>Knows(Joffrey, ( x ))</td>
<td>Knows(( x ), Sansa)</td>
<td></td>
</tr>
</tbody>
</table>
**Unification**

\[ \text{UNIFY}(\alpha, \beta) \text{ returns } \theta \text{ if } \alpha\theta = \beta\theta \]

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( \theta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Knows(Joffrey, x)} )</td>
<td>( \text{Knows(Joffrey, Sansa)} )</td>
<td>( {x/Sansa} )</td>
</tr>
<tr>
<td>( \text{Knows(Joffrey, x)} )</td>
<td>( \text{Knows(y, Sansa)} )</td>
<td>( {x/Sansa, y/Joffrey} )</td>
</tr>
<tr>
<td>( \text{Knows(Joffrey, x)} )</td>
<td>( \text{Knows(y, Mother(Joffrey))} )</td>
<td></td>
</tr>
<tr>
<td>( \text{Knows(Joffrey, x)} )</td>
<td>( \text{Knows(x, Sansa)} )</td>
<td></td>
</tr>
</tbody>
</table>
Unification

\[ \text{UNIFY}(\alpha, \beta) \text{ returns } \theta \text{ if } \alpha \theta = \beta \theta \]

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( \theta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(Joffrey, Sansa)</td>
<td>{x/Sansa}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(y, Sansa)</td>
<td>{x/Sansa, y/Joffrey}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(y, Mother(Joffrey))</td>
<td>{y/Joffrey, x/Mother(Joffrey)}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(x, Sansa)</td>
<td></td>
</tr>
</tbody>
</table>
**Unification**

\[
\text{UNIFY}(\alpha, \beta) \text{ returns } \theta \text{ if } \alpha\theta = \beta\theta
\]

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>\theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(Joffrey, Sansa)</td>
<td>{x/Sansa}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(y, Sansa)</td>
<td>{x/Sansa, y/Joffrey}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(y, Mother(Joffrey))</td>
<td>{y/Joffrey, x/Mother(Joffrey)}</td>
</tr>
<tr>
<td>Knows(Joffrey, x)</td>
<td>Knows(x, Sansa)</td>
<td>fail</td>
</tr>
</tbody>
</table>
Standardising apart

\[ \text{Knows(Joffrey, } x) \land \text{Knows}(x, \text{Sansa}) \] fails
Standardising apart

\[ Knows(Joffrey, x) \land Knows(x, Sansa) \] fails

Standardising apart eliminates overlap of variables, e.g.,
\[ Knows(z_{17}, Sansa) \]
Generalized Modus Ponens (GMP)

**Definite clause:**
disjunction of literals, *exactly* one of which positive
Generalized Modus Ponens (GMP)

**Definite clause:**
disjunction of literals, **exactly** one of which positive
e.g., \((p_1 \land p_2 \land \ldots \land p_n \Rightarrow q)\)
Generalized Modus Ponens (GMP)

**Definite clause:**
disjunction of literals, **exactly** one of which positive

e.g., \((p_1 \land p_2 \land \ldots \land p_n \Rightarrow q)\)

\[
p_1', \; p_2', \; \ldots, \; p_n', \; (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q) \quad \Rightarrow \quad q_{\theta}
\]

where \(p_i'\theta = p_i\theta\) for all \(i\)
Generalized Modus Ponens (GMP)

Definite clause:
disjunction of literals, exactly one of which positive
e.g., \((p_1 \land p_2 \land \ldots \land p_n \Rightarrow q)\)

\[
\begin{align*}
p_1', \ p_2', \ \ldots, \ p_n', \ (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q) \\
\frac{q}{q^\theta}
\end{align*}
\]

where \(p_i' \theta = p_i \theta\) for all \(i\)

Assuming all variables are universally quantified...
Generalized Modus Ponens (GMP)

**Definite clause:**
disjunction of literals, **exactly** one of which positive
e.g., \((p_1 \land p_2 \land \ldots \land p_n \Rightarrow q)\)

\[
\begin{align*}
p_1', ~ p_2', ~ \ldots, ~ p_n', \quad (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q) \\
q_\theta
\end{align*}
\]

where \(p_i' \theta = p_i \theta\) for all \(i\)

Assuming all variables are universally quantified...

\(p_1'\) is \(\text{King}(\text{Joffrey})\)  \(p_1\) is \(\text{King}(x)\)
\(p_2'\) is \(\text{Greedy}(y)\)  \(p_2\) is \(\text{Greedy}(x)\)
\(\theta\) is \(\{x/\text{Joffrey}, \ y/\text{Joffrey}\}\)  \(q\) is \(\text{Evil}(x)\)
\(q_\theta\) is \(\text{Evil}(\text{Joffrey})\)
Soundness of GMP

Need to show that

\[ p_1', \ldots, p_n', (p_1 \land \ldots \land p_n \Rightarrow q) \models q \theta \]

provided that \( p_i' \theta = p_i \theta \) for all \( i \)

Lemma: If \( \varphi \) is definite clause, then \( \varphi \models \varphi \theta \) by Universal Instantiation.

1. \( (p_1 \land \ldots \land p_n \Rightarrow q) \models (p_1 \land \ldots \land p_n \Rightarrow \ q) \theta = (p_1 \theta \land \ldots \land p_n \theta \Rightarrow q \theta) \)

2. \( p_1', \ldots, p_n' \models p_1' \land \ldots \land p_n' \models p_1' \theta \land \ldots \land p_n' \theta \)

3. From 1 and 2, \( q \theta \) follows by ordinary Modus Ponens
Coming next

- Making sound and efficient inferences
- Where to start?
- How to go on?