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?

Robert J. Aumann
Nobel Prize Winner
Economics

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

What is rationality?

Robert J. Aumann
Nobel Prize Winner
Economics

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

Agents (not only humans) *can* be rational!

The lectures one by one

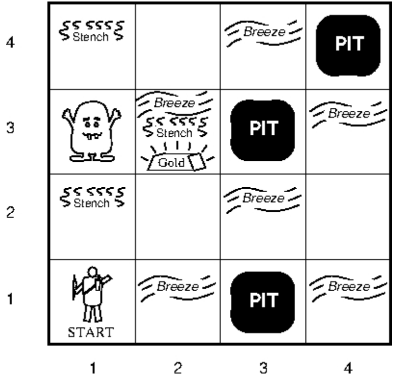
- Logical Agents I
- Logical Agents II
- An Uncertain World
- Making Sense of Uncertainty
- Making (Good) Decisions
- Making Good Decisions in time
- Learning from Experience I
- Learning from Experience II

Logical Agents I

The main reference

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

The Wumpus World



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

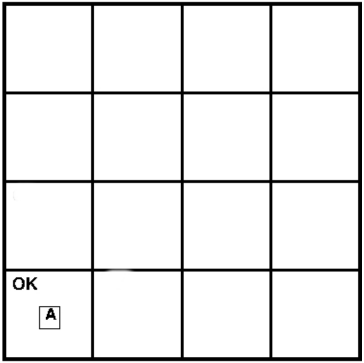
- What we TELL the knowledge base
- What we ASK the knowledge base

```
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
```

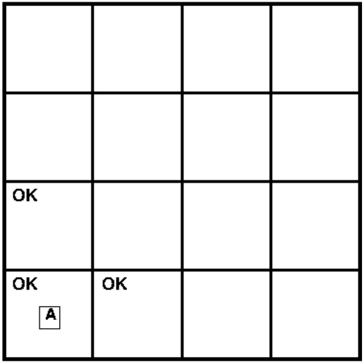
Rational explorations

- The starting state...



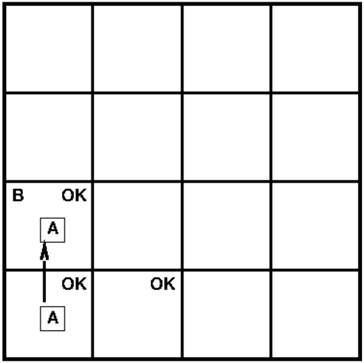
Rational explorations

- and what we know.



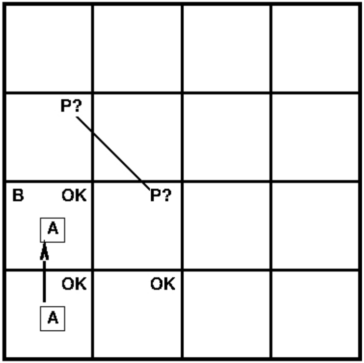
Rational explorations

- B stands for Breeze



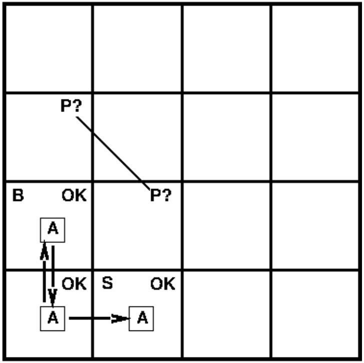
Rational explorations

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



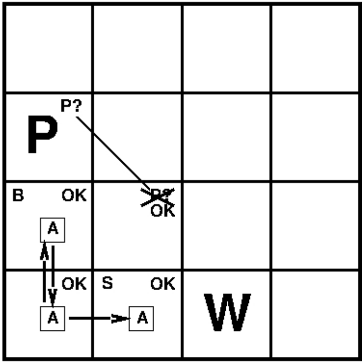
Rational explorations

- S stands for smell
- What do we know?



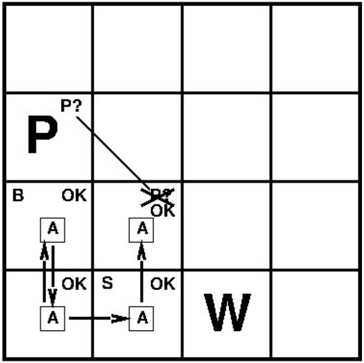
Rational explorations

- Logic is the key



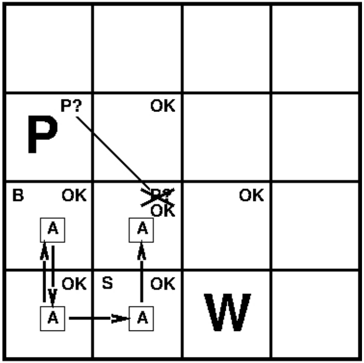
Rational explorations

- The further we go the more we know



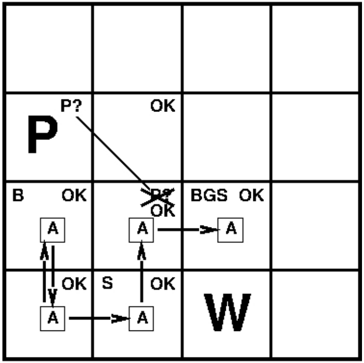
Rational explorations

- The further we go the more we know



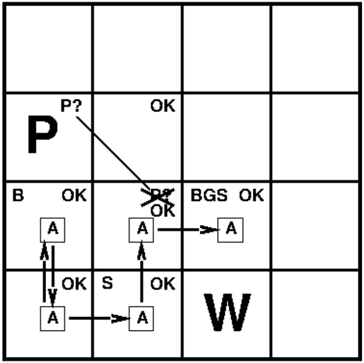
Rational explorations

- 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"

$$\begin{aligned} B_{1,1} &\Leftrightarrow (P_{1,2} \vee P_{2,1}) \\ B_{2,1} &\Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1}) \end{aligned}$$

"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

FOL KB

- We can encode the KB at each particular time point using FOL

```
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
```

FOL

- You already know how to describe the WW in first order logic

FOL

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

FOL

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

FOL

- You already know how to describe the WW in first order logic
  - Percept (at given time), e.g., $Percept([Stench, Breeze, Glitter], 5)$ or $Percept([None, Breeze, None], 3)$
  - Starting Knowledge Base, e.g., $\neg AtGold(0)$
  - Axioms to generate new knowledge from percepts, e.g., $\forall\, s, b, t \;\; Percept([s, b, Glitter], t) \;\Rightarrow\; AtGold(t)$

FOL

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

FOL

- You already know how to describe the WW in first order logic
  - Percept (at given time), e.g., $Percept([Stench, Breeze, Glitter], 5)$ or $Percept([None, Breeze, None], 3)$
  - Starting Knowledge Base, e.g., $\neg AtGold(0)$
  - Axioms to generate new knowledge from percepts, e.g., $\forall\, s, b, t \;\; Percept([s, b, Glitter], t) \;\Rightarrow\; AtGold(t)$
  - Axioms to generate actions (plans) from KB, e.g., $\forall\, t \;\; AtGold(t) \wedge \neg Holding(Gold, t) \;\Rightarrow\; Action(Grab, t)$
  - Axioms from knowledge to knowledge, e.g., $\forall\, t \;\; AtGold(t) \wedge Action(Grab, t) \;\Rightarrow\; Holding(Gold, t+1)$

Describing the world

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

Describing the world

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

Location $At(Agent, s, t)$

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

Location $At(Agent, s, t)$

Decision-making $\forall t \;\; AtGold(t) \Rightarrow Action(Grab, t)$

Internal reflection $\forall t \;\; AtGold(t) \wedge \neg Holding(Gold, t) \Rightarrow Action(Grab, t)$, do we have gold already? (notice we cannot observe if we are holding gold, we need to track it)

Describing the world

Adjacent squares

$$\forall x, y, a, b \;\; Adjacent([x, y], [a, b]) \quad \Leftrightarrow$$
$$(x = a \land (y = b - 1 \lor y = b + 1) \lor$$
$$(y = b \land (x = a - 1 \lor x = a + 1))$$

Describing the world

Adjacent squares

$$\forall x, y, a, b \;\; Adjacent([x, y], [a, b]) \;\; \Leftrightarrow$$
$$(x = a \wedge (y = b - 1 \vee y = b + 1) \vee$$
$$(y = b \wedge (x = a - 1 \vee x = a + 1))$$

"A square is breezy **if and only if** there is an adjacent pit"

$$\forall s \;\; , Breezy(s) \;\Leftrightarrow\; \exists r \, (Adjacent(r, s) \wedge Pit(r))$$

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

$$Tell(KB, King(Joffrey))$$

$$Tell(KB, Person(Jaime))$$

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

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?

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?

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 σ,

Substitutions

Definition

Given a sentence S and a substitution σ,

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

Substitutions

Definition

Given a sentence S and a substitution σ,

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

$S = Smarter(x, y)$

Substitutions

Definition

Given a sentence S and a substitution σ,

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

$S = Smarter(x, y)$

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

Substitutions

Definition

Given a sentence S and a substitution σ,

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

$S = Smarter(x, y)$

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

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

Substitutions

Definition

Given a sentence S and a substitution σ,

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

$S = Smarter(x, y)$

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

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

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

Unification

$$\forall x \;\; King(x) \land Greedy(x) \;\Rightarrow\; Evil(x)$$

$$King(Joffrey)$$

$$\forall y \;\; Greedy(y)$$

Unification

$$\forall x \;\; King(x) \wedge Greedy(x) \;\Rightarrow\; Evil(x)$$

$$King(Joffrey)$$

$$\forall y \;\; Greedy(y)$$

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

Unification

$$\forall x \;\; King(x) \wedge Greedy(x) \;\Rightarrow\; Evil(x)$$

$$King(Joffrey)$$

$$\forall y \;\; Greedy(y)$$

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

$\theta = \{x/Joffrey, y/Joffrey\}$ works

Unification

$\textsc{Unify}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

Unification

$\text{UNIFY}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(Joffrey, x)$	$Knows(Joffrey, Sansa)$	
$Knows(Joffrey, x)$	$Knows(y, Sansa)$	
$Knows(Joffrey, x)$	$Knows(y, Mother(Joffrey))$	
$Knows(Joffrey, x)$	$Knows(x, Sansa)$	

Unification

$\textsc{Unify}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(Joffrey, x)$	$Knows(Joffrey, Sansa)$	$\{x/Sansa\}$
$Knows(Joffrey, x)$	$Knows(y, Sansa)$	
$Knows(Joffrey, x)$	$Knows(y, Mother(Joffrey))$	
$Knows(Joffrey, x)$	$Knows(x, Sansa)$	

Unification

$\textsc{Unify}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(Joffrey, x)$	$Knows(Joffrey, Sansa)$	$\{x/Sansa\}$
$Knows(Joffrey, x)$	$Knows(y, Sansa)$	$\{x/Sansa, y/Joffrey\}$
$Knows(Joffrey, x)$	$Knows(y, Mother(Joffrey))$	
$Knows(Joffrey, x)$	$Knows(x, Sansa)$	

Unification

$\textsc{Unify}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(Joffrey, x)$	$Knows(Joffrey, Sansa)$	$\{x/Sansa\}$
$Knows(Joffrey, x)$	$Knows(y, Sansa)$	$\{x/Sansa, y/Joffrey\}$
$Knows(Joffrey, x)$	$Knows(y, Mother(Joffrey))$	$\{y/Joffrey, x/Mother(Joffrey)\}$
$Knows(Joffrey, x)$	$Knows(x, Sansa)$	

Unification

$\text{UNIFY}(\alpha, \beta)$ returns θ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(Joffrey, x)$	$Knows(Joffrey, Sansa)$	$\{x/Sansa\}$
$Knows(Joffrey, x)$	$Knows(y, Sansa)$	$\{x/Sansa, y/Joffrey\}$
$Knows(Joffrey, x)$	$Knows(y, Mother(Joffrey))$	$\{y/Joffrey, x/Mother(Joffrey)\}$
$Knows(Joffrey, x)$	$Knows(x, Sansa)$	$fail$

Standardising apart

$Knows(Joffrey, x)$ & $Knows(x, Sansa)$ fails

Standardising apart

$Knows(Joffrey, x)$ & $Knows(x, Sansa)$ 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., $(p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)$

Generalized Modus Ponens (GMP)

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

$$\frac{p_1{}', \;\; p_2{}', \;\ldots, \;\; p_n{}', \;\; (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \qquad \text{where } p_i{}'\theta = p_i\theta \text{ for all } i$$

Generalized Modus Ponens (GMP)

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

$$\frac{p_1{}',\ \ p_2{}',\ \ldots,\ p_n{}',\ \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta} \qquad \text{where } p_i{}'\theta = p_i\theta \text{ for all } i$$

Assuming all variables are universally quantified...

Generalized Modus Ponens (GMP)

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

$$\frac{p_1{}', \ p_2{}', \ \ldots, \ p_n{}', \ (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{q\theta}$$ where $p_i{}'\theta = p_i\theta$ for all i

Assuming all variables are universally quantified...

$p_1{}'$ is $King(Joffrey)$	p_1 is $King(x)$
$p_2{}'$ is $Greedy(y)$	p_2 is $Greedy(x)$
θ is $\{x/Joffrey, y/Joffrey\}$	q is $Evil(x)$
$q\theta$ is $Evil(Joffrey)$	

Soundness of GMP

Need to show that

$$p_1', \ldots, p_n', (p_1 \wedge \ldots \wedge p_n \Rightarrow q) \models q\theta$$

provided that $p_i'\theta = p_i\theta$ for all i
Lemma: If φ is definite clause, then $\varphi \models \varphi\theta$ by Universal Instantiation.

1. $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \models (p_1 \wedge \ldots \wedge p_n \Rightarrow q)\theta = (p_1\theta \wedge \ldots \wedge p_n\theta \Rightarrow q\theta)$
2. $p_1', \ldots, p_n' \models p_1' \wedge \ldots \wedge p_n' \models p_1'\theta \wedge \ldots \wedge p_n'\theta$
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?