Intelligent Agents

Chapter 2
Outline

♦ Agents and environments
♦ Rationality
♦ PEAS (Performance measure, Environment, Actuators, Sensors)
♦ Environment types
♦ Agent types
Agents and environments

Agents include humans, robots, softbots, thermostats, etc.

The agent function maps from percept histories to actions:

\[ f : \mathcal{P}^* \rightarrow A \]

The agent program runs on the physical architecture to produce \( f \)
Vacuum-cleaner world

Percepts: location and contents, e.g., \([A, Dirty]\)

Actions: \(Left, Right, Suck, NoOp\)
A vacuum-cleaner agent

<table>
<thead>
<tr>
<th>Percept sequence</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>[A, Clean]</td>
<td></td>
</tr>
<tr>
<td>[A, Dirty]</td>
<td></td>
</tr>
<tr>
<td>[B, Clean]</td>
<td></td>
</tr>
<tr>
<td>[B, Dirty]</td>
<td></td>
</tr>
<tr>
<td>[A, Clean], [A, Clean]</td>
<td></td>
</tr>
<tr>
<td>[A, Clean], [A, Dirty]</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
A vacuum-cleaner agent

<table>
<thead>
<tr>
<th>Percept sequence</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>[A, Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[B, Clean]</td>
<td>Left</td>
</tr>
<tr>
<td>[B, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[A, Clean], [A, Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, Clean], [A, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**function** `REFLEX-VACUUM-AGENT([location,status])` **returns** an action

- if `status = Dirty` then return `Suck`
- else if `location = A` then return `Right`
- else if `location = B` then return `Left`

What is the `right` function?
Can it be implemented in a small agent program?
Rationality

Fixed performance measure evaluates the environment sequence
– one point per square cleaned up in time $T$?
Rationality

Fixed performance measure evaluates the environment sequence
– one point per square cleaned up in time $T$?
– one point per clean square per time step, minus one per move?
Rationality

Fixed performance measure evaluates the environment sequence

– one point per square cleaned up in time \( T \)?
– one point per clean square per time step, minus one per move?
– penalize for \( > k \) dirty squares?
Rationality

Fixed **performance measure** evaluates the **environment sequence**
- one point per square cleaned up in time $T$?
- one point per clean square per time step, minus one per move?
- penalize for $> k$ dirty squares?

A **rational agent** chooses whichever action maximizes the **expected** value of the performance measure **given the percept sequence to date**

Rational $\neq$ omniscient
- percepts may not supply all relevant information
Rational $\neq$ clairvoyant
- action outcomes may not be as expected
Hence, rational $\neq$ successful

Rational $\Rightarrow$ exploration, learning, autonomy
To design a rational agent, we must specify the task environment.

Consider, e.g., the task of designing an automated taxi:

- **Performance measure**
- **Environment**
- **Actuators**
- **Sensors**
To design a rational agent, we must specify the task environment

Consider, e.g., the task of designing an automated taxi:

**Performance measure** safety, destination, profits, legality, comfort, . . .

**Environment** streets in Lower Mainland, traffic, pedestrians, weather, . . .

**Actuators** steering, accelerator, brake, horn, speaker/display, . . .

**Sensors** video, accelerometers, gauges, engine sensors, keyboard, GPS, . . .
Internet shopping agent

Performance measure

Environment

Actuators

Sensors
Internet shopping agent

**Performance measure**  price, quality, appropriateness, efficiency

**Environment**  current and future WWW sites, vendors, shippers

**Actuators**  display to user, follow URL, fill in form

**Sensors**  HTML pages (text, graphics, scripts)
## Environment types

<table>
<thead>
<tr>
<th>Observable??</th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
</table>
### Environment types

<table>
<thead>
<tr>
<th>Observables</th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
# Environment types

<table>
<thead>
<tr>
<th></th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic</td>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>Episodic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Environment types

<table>
<thead>
<tr>
<th>Observable</th>
<th>Deterministic</th>
<th>Episodic</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
## Environment types

<table>
<thead>
<tr>
<th>Observable??</th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic??</td>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>Episodic??</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Static??</td>
<td>Yes</td>
<td>Yes</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td>Discrete??</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic</td>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td>Episodic</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Static</td>
<td>Yes</td>
<td>Yes</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td>Discrete</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Single-agent</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Environment types

<table>
<thead>
<tr>
<th></th>
<th>8-Puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Observable??</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic??</strong></td>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic??</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Static??</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td><strong>Discrete??</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single-agent??</strong></td>
<td>Yes</td>
<td>No</td>
<td>Yes (except auctions)</td>
<td>No</td>
</tr>
</tbody>
</table>

The environment type largely determines the agent design

The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent
Agent types

Four basic types in order of increasing generality:
  – simple reflex agents
  – reflex agents with state
  – goal-based agents
  – utility-based agents

All these can be turned into learning agents
Simple reflex agents

Agent

Sensors

What the world is like now

Condition–action rules

What action I should do now

Actuators

Environment
Example

\[\text{function \textsc{Reflex-Vacuum-Agent}([location, status]) returns an action}\]

\[
\begin{align*}
\text{if } &\text{ status = Dirty then return Suck} \\
\text{else if } &\text{ location = A then return Right} \\
\text{else if } &\text{ location = B then return Left}
\end{align*}
\]
Reflex agents with state

Agent

- State
- How the world evolves
- What my actions do
- Condition–action rules

Environment

- Sensors
- What the world is like now
- Actuators
- What action I should do now
Example

\begin{verbatim}
function Reflex-Vacuum-Agent([location, status]) returns an action
static: last_A, last_B, numbers, initially \infty

if status = Dirty then . . .
\end{verbatim}
Goal-based agents

Agent

Environment

State

How the world evolves

What my actions do

Goals

What the world is like now

What it will be like if I do action A

What action I should do now

Sensors

Actuators

Chapter 2
Utility-based agents

Agent

Environment

State

What the world is like now

What it will be like if I do action A

How happy I will be in such a state

What action I should do now

How the world evolves

What my actions do

Utility

Sensors

Actuators
Learning agents

Agent

Environment

Performance standard

Critic

Sensors

Performance element

feedback

changes

knowledge

learning goals

Problem generator

Actuators

Chapter 2  29
Agents interact with environments through actuators and sensors

The agent function describes what the agent does in all circumstances

The performance measure evaluates the environment sequence

A perfectly rational agent maximizes expected performance

Agent programs implement (some) agent functions

PEAS descriptions define task environments

Environments are categorized along several dimensions:
  - observable?
  - deterministic?
  - episodic?
  - static?
  - discrete?
  - single-agent?

Several basic agent architectures exist:
  - reflex
  - reflex with state
  - goal-based
  - utility-based