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:
\[ f : P^* \rightarrow A \]

The agent program runs on the physical architecture to produce \( f \)

A vacuum-cleaner agent

Percept sequence | Action
---|---
\([A, Clean]\) | Right
\([A, Dirty]\) | Suck
\([B, Clean]\) | Left
\([B, Dirty]\) | Suck
\([A, Clean], [A, Clean]\) | Right
\([A, Clean], [A, Dirty]\) | Suck

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

Vacuum-cleaner world

Percepts: location and contents, e.g., \([A, Dirty]\)
Actions: Left, Right, Suck, NoOp
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

PEAS

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
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
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
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>
<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>No</td>
<td>No</td>
<td>No</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

Environment types

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Semi</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

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

Example

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
Reflex agents with state

Agent

Environment

Sensors

State

How the world evolves

What the world is like now

What my actions do

Condition-action rules

What action I should do now

Actuators

Utility-based agents

Agent

Environment

Sensors

State

How the world evolves

What the world is like now

What my actions do

What it will be like if I do action A

Utility

How happy I will be in such a state

What action I should do now

Actuators

Example

function REFLEX-VACUUM-AGENT([location, status]) returns an action
static: last A, last B, numbers, initially ∞
if status = Dirty then . . .

Goal-based agents

Agent

Environment

Sensors

State

How the world evolves

What the world is like now

What my actions do

Goals

What it will be like if I do action A

What action I should do now

Actuators

Learning agents

Agent

Environment

Critic

Sensors

Performance standard

feedback

Performance element

changes

Performance element

learning goals

knowledge

Problem generator

Actuators

Summary

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