Outline

♦ Agents and environments
♦ Rationality
♦ PEAS (Performance measure, Environment, Actuators, Sensors)
♦ Environment types
♦ Agent types
An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators.

The agent function maps from percept histories to actions:

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

The agent program runs on the physical architecture to produce \( f \)
Percepts: location \([A, B]\) and contents \([\text{Dirty, Clean}]\), e.g., \([A, \text{Dirty}]\)

How many state combinations are possible? \(2 \times 2^2 = 8\)

How about when we have more advanced vacuum? vacuum \([\text{on/off/sleep}]\) and more locations \([1 \ 10]\)? 3072

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>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 $T$?
- 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??** US streets/freeways, 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)

More examples in AIMA Fig. 2.5
Properties of task environments

- Fully observable vs partially observable.
- Deterministic vs stochastic
- Episodic vs sequential
- Statics vs dynamic
- Discrete vs continuous
- Benign vs adversarial
## Environment types

<table>
<thead>
<tr>
<th>Observable</th>
<th>Deterministic</th>
<th>Episodic</th>
<th>Static</th>
<th>Discrete</th>
<th>Single-agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solitaire</td>
<td>Backgammon</td>
<td>Internet shopping</td>
<td>Taxi</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[528x515]Environment types
Environment types

<table>
<thead>
<tr>
<th>Environment Type</th>
<th>Solitaire</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>Semi</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>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

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

Example

function REFLEX-VACUUM-AGENT([location, status]) returns an action
static: last_A, last_B, numbers, initially \( \infty \)

    if status = Dirty then . . .
Goal-based agents

Agent

Sensors

State

What the world is like now

How the world evolves

What my actions do

What it will be like if I do action A

Goals

What action I should do now

Actuators

Environment

Chapter 2  17
Utility-based agents

Agent

- State
- How the world evolves
- What my actions do
- Utility

Environment

- Sensors
  - 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

Actuators
Learning agents

Performance standard

Agent

Critic

Sensors

Environment

feedback

changes

knowledge

learning goals

Problem generator

Performance element

Learning element

Actuators

Learning goals
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:

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