Chapter 2

Outline

Agents and environments

Outline

Agent types

Environment types

PEAS (Performance measure, Environment, Actuators, Sensors)

Rationality

Agents and environments

Vacuum-cleaner world

Percepts: location and contents, e.g., “A; Clean”
Actions: Left, Right, Suck, NoOp

Vacuum-cleaner agent
Percept sequence
Action

A; Clean
Right
A; Dirty
Suck
B; Clean
Left
B; Dirty
Suck
A; Clean

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

Assignment 0 (lisp refresher) due 1/28

Assignment 0 lisp refresher (due 1/28)

Assignment 0 lisp refresher (due 1/28)

Assignment 0 lisp refresher (due 1/28)

Assignment 0 lisp refresher (due 1/28)

Assignment 0 lisp refresher (due 1/28)

Assignments

Agents and environments

Agents include humans, robots, softbots, thermostats, etc.
The agent function maps from percept histories to actions:

What is the right function?

The agent program runs on the physical architecture to produce f
The agent function maps from percept histories to actions:
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 = omniscient { percepts may not supply all relevant information

Rational = clairvoyant { action outcomes may not be as expected

Hence, rational = successful

Rational = autonomous

Exploration, learning, autonomy

Chapter 2

To design a rational agent, we must specify the task environment.

Performance measure

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: price, quality, appropriateness, efficiency

Environment: current and future WWW sites, vendors, shippers

Actuators: display to user, follow URL, fill in forms, …

Sensors: HTML pages (text, graphics, scripts)

Environment types

Solitaire Backgammon Internet shopping Taxi

Observable

Deterministic

Episodic

Static

Discrete

Single-agent

Chapter 2
The environment type largely determines the agent design.

<table>
<thead>
<tr>
<th>Environment types</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>
</tbody>
</table>

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

Four basic types in order of increasing generality:

1. Simple reflex agents
2. Reflex agents with state
3. Goal-based agents
4. Utility-based agents

All these can be turned into learning agents.

Simple reflex agents

Agent

Environment

Sensors

What the world is like now

Condition-action rules

Actuators

What the world is like now

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

(setq joe (make-agent :name 'joe :body (make-agent-body) :program (make-reflex-vacuum-agent-program)))

(defun make-reflex-vacuum-agent-program ()
  #'(lambda (percept)
     (let ((location (first percept)) (status (second percept))
          (last-A infinity) (last-B infinity))
       (cond ((eq status 'dirty)
              (if (eq location 'A) (setq last-A 0) (setq last-B 0))
                'Suck)
             ((eq location 'A) (if (> last-B 3) 'Right 'NoOp))
             ((eq location 'B) (if (> last-A 3) 'Left 'NoOp))))))
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.

Several basic agent architectures exist:

- reflex
- reflex with state
- goal-based
- utility-based

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

Summary

Learning agents

Performance standard

Agent

Environment

Performance standard

Learning

Critic

Actuators

Sensors

Learning agents

Utility-based agents

Agent

Environment

Actuators

Sensors

Utility-based agents

Performance standard