CHAPTER 2

Intelligent Agents
Outline

- Vacuum world
- Agent types
- Agent functions and programs
- Environment types
- PAGE (Percepts, Actions, Goals, Environment)
Consider, e.g., the task of designing an autonomous taxi:

Must first specify the setting for intelligent agent design
customers, ... Environment

... US urban streets, freeways, traffic, pedestrians, weather,

... Geiger counter, safety, reach destination, maximize profits, obey laws, passenger,

... Goal: steer, accelerate, brake, horn, speak/display,

... Action: GPS, perception: video, accelerometers, gyroscopes, engine sensors, keyboard

Consider e.g. the task of designing an autonomous taxi:

Must first specify the setting for intelligent agent design

PAGE
Rational action: which ever action maximizes the expected value of the performance measure given the percept sequence to date.

Without loss of generality, "goals" specifiable by performance measure.

Rational agents

Rational action ≠ Successful
Rational action ≠ Clairvoyant
Rational action ≠ Omniscient
<table>
<thead>
<tr>
<th>Environment Types</th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet Shopping</th>
<th>Taxi</th>
<th>Discrete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Episodic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deterministic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accessible</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The real world is (of course) inaccessible, stochastic, sequential, dy-
amic, continuous.

The environment type largely determines the agent design.

<table>
<thead>
<tr>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet Shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete?</td>
<td>Static?</td>
<td>Episodic?</td>
<td>Accessible?</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Semi</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>Party</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
An agent program takes a single percept as input, keeps internal state:

\[
\text{Update-memory(memory, action)}
\]

\[
\text{Choose-best-action(memory)}
\]

\[
\text{Update-memory(memory, preferred)}
\]

**function** `Agent-skeleton(percept)` returns `action`

An agent program is completely specified by the agent function.

\[
\text{Return action}
\]

Aim: find a way to implement the rational agent function concisely.

One agent function (or a small equivalence class) is rational.

Obviously, a lookup table would usually be immense.

In principle, one can supply each possible sequence to see what it does.

Mapping percept sequences to actions
AI/A code

```
((no-op)
 (push percept memory)
 (lambda (percept)
   ((memory nil)
     (define make-dump-agent-program ()

 (define (make-dump-agent-program:
     (set! Joe (make-agent :name 'Joe :body (make-agent-body

 (define run algorithms on domains rather than agents in environments.
   - domains: problem types and instances for input to algorithms
   - environments: code defining environment types, simulations
   - algorithms: code for the methods used by the agent programs
   - agents: code defining agent types and programs

The code for each topic is divided into four directories:
```

The text seems to be about defining a program for an agent, with some code snippets and explanations of the terms and concepts used in the context of AI and agent-based simulations.
Four basic types in order of increasing generality:

- Simple reflex agents
- Reflex agents with state
- Goal-based agents
- Utility-based agents
Agent

Environment

Sensors

Effectors

Condition-action rules

What the world is like now

What action I should do now

Simple reflex agents
Agent

Environment

Effectors

Sensors

State

How the world evolves

What my actions do

What the world

What it is like now

What action I should do now

Condition-action rules

Rethink agents with state
Agent-based agents

Environment

Agent

Goals

What actions do

What it will be like

If I do action A

What action I should do now

What the world is like now

How the world evolves

State

What the world evolves

What my actions do

If I do action A

What it will be like

What action I should do now

What the world is like now

Goals

Effects

Sensors

- such actions put dirt into agent body (or not)
- movement actions work unless bump into wall
- grid, walls/obstacles, dirt distribution and creation, agent body

Environment

-1000 for shutting off away from home
-1 for each action
+100 for each piece of dirt cleaned up

Goals (performance measure on environment history)

Actions shortcuts forward such (turn left) (turn right)

Percepts (bump, dirt, home)

The vacuum world