Chapter 2  Agents
Outline

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

- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting** upon that environment through **actuators**
- Human agent: eyes, ears, and other organs for **sensors**;
- hands, legs, mouth, and other body parts for **actuators**
- Robotic agent: cameras and infrared range finders for **sensors**;
- various motors for **actuators**
Rational Agents

How to design this?

Sensors

Agent

Environment

percepts

actions

Effectors

AI chapter 2
Agents and environments

- The **agent function** maps from percept histories to actions:

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

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

- \( \text{agent} = \text{architecture} + \text{program} \)

AI chapter 2
Perception

• Percept
  • Perceptual inputs at an instant
  • May include perception of internal state
• Percept Sequence
  • Complete history of all prior percepts

• Do you need a percept sequence to play Chess?
Intelligent Agents and Artificial Intelligence

- **Example:** Human mind as network of thousands or millions of agents working in parallel. To produce real artificial intelligence, this school holds, we should build computer systems that also contain many agents and systems for arbitrating among the agents' competing results.

- Distributed decision-making and control

- Challenges:
  - Action selection: What next action to choose
  - Conflict resolution

AI chapter 2
Vacuum-cleaner world

- Percepts: location and contents, e.g., [A,Dirty]
- Actions: Left, Right, Suck, NoOp
Simple Agent Function for Vacuum Cleaner World

<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>[A,Clean], [B,Clean]</td>
<td>Left</td>
</tr>
<tr>
<td>[A,Clean], [B,Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[A,Dirty], [A,Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A,Dirty], [A,Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>[A,Clean], [A,Clean], [A,Clean]</td>
<td>Right</td>
</tr>
</tbody>
</table>
function \text{Reflex-Vacuum-Agent}([\text{location}, \text{status}]) \text{ returns an action}

\text{if status} = \text{Dirty} \text{ then return Suck}
\text{else if location} = A \text{ then return Right}
\text{else if location} = B \text{ then return Left}

\text{Figure 2.10}
2.2 Concept of Rational agents

• An agent should strive to "do the right thing", based on what it can perceive and the actions it can perform.
• The right action is the one that will cause the agent to be **most successful**.
• **Performance measure**: An objective criterion for success of an agent's behavior
• E.g., performance measure of a vacuum-cleaner agent could be
  • amount of dirt cleaned up,
  • amount of time taken,
  • amount of electricity consumed,
  • amount of noise generated, etc.
Definition: Rational agents

- **Rational Agent**: For each possible percept sequence, a rational agent should select an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has.
PEAS

- **PEAS:**
  - Performance measure
  - Environment
  - Actuators
  - Sensors
Rational agents (omniscience)

• Rationality is distinct from omniscience (all-knowing with infinite knowledge)
• Accident?
• Agents can perform actions in order to modify future percepts so as to obtain useful information (information gathering, exploration)
Autonomy

• An agent is **autonomous** if its behavior is determined by its own experience (with ability to learn and adapt)

• Agent must learn from what it perceives
  - initial configuration reflects prior knowledge of environment
  - agent gains experience so that knowledge of environment may be modified/augmented
**The Right Thing = The Rational Action**

- **Rational Action:** The action that maximizes the expected value of the performance measure given the percept sequence to date

  - Rational = Best [Yes, to the best of its knowledge]
  - Rational = Optimal [Yes, to the best of its abilities (incl. its constraints)]
  - Rational ≠ Omniscience (全知)
  - Rational ≠ Clairvoyant (有超人力)
  - Rational ≠ Successful
2.3 Environment

- PEAS
- Must first specify the setting for intelligent agent design
- Consider, e.g., the task of designing an automated taxi driver:
  - **Performance measure**: Safe, fast, legal, comfortable trip, maximize profits
  - **Environment**: Roads, other traffic, pedestrians, customers
  - **Actuators**: Steering wheel, accelerator, brake, signal, horn
  - **Sensors**: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard
PEAS **Agent: Medical diagnosis system**

- **Performance measure:** Healthy patient, minimize costs, lawsuits
- **Environment:** Patient, hospital, staff
- **Actuators:** Screen display (questions, tests, diagnoses, treatments, referrals)
- **Sensors:** Keyboard (entry of symptoms, findings, patient's answers)
PEAS Agent: Part-picking robot

- Performance measure: Percentage of parts in correct bins
- Environment: Conveyor belt with parts, bins
- Actuators: Jointed arm and hand
- Sensors: Camera, joint angle sensors
PEAS Agent: Interactive English tutor

• Performance measure: Maximize student's score on test
• Environment: Set of students
• Actuators: Screen display (exercises, suggestions, corrections)
• Sensors: Keyboard
Environment types

- **Fully observable** (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.

- **Deterministic** (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is strategic)

- **Episodic** (vs. sequential): The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.
Environment types

- **Static** (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is *semidynamic* if the environment itself does not change with the passage of time but the agent's performance score does)

- **Discrete** (vs. continuous): A limited number of distinct, clearly defined percepts and actions.

- **Single agent** (vs. multiagent): An agent operating by itself in an environment.
  - Competitive
  - Cooperative
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Chess with a clock</th>
<th>Chess without a clock</th>
<th>Taxi driving</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully observable</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Deterministic</td>
<td>Strategic</td>
<td>Strategic</td>
<td>No</td>
</tr>
<tr>
<td>Episodic</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Static</td>
<td>Semi</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Discrete</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Single agent</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
2.4 The structure of Agents

• Agent functions and programs
  • An agent is completely specified by the agent function mapping percept sequences to actions
  • One agent function (or a small equivalence class) is rational
  • Aim: find a way to implement the rational agent function concisely
Table-lookup agent

```plaintext
function TABLE-DRIVEN-AGENT (percept) returns action
  static: percepts, a sequence, initially empty
  table, a table, indexed by percept sequences, initially fully specified

  append percept to the end of percepts
  action ← LOOKUP(percepts, table)
  return action
```

- **Drawbacks:**
  - Huge table
  - Take a long time to build the table
  - No autonomy
  - Even with learning, need a long time to learn the table entries
Structure of Intelligent Agents

- **Agent = architecture + program**

- **Agent program:** the implementation of \( f: P^* \rightarrow A \), the agent’s perception-action mapping

  ```
  function Skeleton-Agent(Percept) returns Action
  memory ← UpdateMemory(memory, Percept)
  Action ← ChooseBestAction(memory)
  memory ← UpdateMemory(memory, Action)
  return Action
  ```

- **Architecture:** a device that can execute the agent program (e.g., general-purpose computer, specialized device, beobot, etc.)
Agent types

• Four basic types in order of increasing generality:
  • Simple reflex agents
  • Model-based reflex agents
  • Goal-based agents
  • Utility-based agents
  • Learning Agents
Agent types

• Reflex agents
  • Reactive: No memory

• Reflex agents with internal states
  • W/o previous state, may not be able to make decision
    • E.g. brake lights at night.

• Goal-based agents
  • Goal information needed to make decision
Agent types

• Utility-based agents
  • How well can the goal be achieved (degree of happiness)

• What to do if there are conflicting goals?
  • Speed and safety

• Which goal should be selected if several can be achieved?
Simple-reflex-Agent

- Select actions on the basis of current percept, ignoring the rest of the percept history.
- Vacuum Agent
- Car driver (barking light).
- **Condition—action rule** written as
  - if *car-in-front-is-braking* then *initiate-braking*
- **Condition—action rule written as**
  - if *car-in-front-is-braking* then *initiate-braking*
- Humans also have many such connections, some of which are **learned responses** (as for drive and some of which are **innate reflexes** (such as, blinking when something approaches the eye.)
Simple-reflex-Agent

Agent

Sensors

What the world is like now

Condition-action rules

What action I should do now

Effectors

Environment

AI chapter 2
function SIMPLE-REFLEX-AGENT(\textit{percept})
returns action

static: \textit{rules}, a set of condition-action rules

\begin{verbatim}
state \leftarrow \text{INTERPRET-INPUT} (\textit{percept})
rule \leftarrow \text{RULE-MATCH} (\textit{state}, \textit{rules})
action \leftarrow \text{RULE-ACTION} [\textit{rule}]
return \textit{action}
\end{verbatim}
Simple-reflex-Agent

- Reactive agents do not have internal symbolic models.
- Act by *stimulus-response* to the current state of the environment.
- Each reactive agent is simple and interacts with others in a basic way.
- Complex patterns of behavior emerge from their interaction.

**Benefits**: robustness, fast response time

**Challenges**: scalability, how intelligent? and how do you debug them?
Problem?

• Remove location sensor from Vacuum agent?
• Different environment?
A model-based Reflex agent

• Thus, even for the simple braking rule, our driver will have to maintain some sort of **internal state** in order to choose an action.

• Here, the internal state is not too extensive—it just needs the previous frame from the camera to detect when two red lights at the edge of the vehicle go on or off simultaneously.

• requires two kinds of knowledge to be encoded in the agent program.
  • how the world evolves independently of the agent
  • how the agent's own action affect the

• Model of the world
A model-based reflex agent

- State
- How the world evolves
- What my actions do
- Condition-action rules
- What the world is like now
- What action I should do now
- Sensors
- Effectors

AI chapter 2
A model-based reflex agent

```plaintext
function REFLEX-AGENT-WITH-STATE (percept) returns action
  static: state, a description of the current world state
           rules, a set of condition-action rules

  state ← UPDATE-STATE (state, percept)
  rule ← RULE-MATCH (state, rules)
  action ← RULE-ACTION [rule]
  state ← UPDATE-STATE (state, action)
  return action
```

AI chapter 2
Goal-based Agent

• The agent needs some sort of goal information, which describes situations that are desirable.
  • Taxi: the passenger's destination.
• The agent program can combine this information about the results of possible actions (the same information as was used to u internal state in the reflex agent) in order to choose actions that achieve the goal.
• Sometime this will be simple, when goal satisfaction results immediately from a single action; sometime it will be more tricky, when the agent has to consider long sequences of twists and turns to.
Goal-based agents
Utilities-based agents

• Goals alone are not really enough to generate high-quality behavior.
• There are action sequences that will get the taxi to its destination, thereby achieving the goal, but so are quicker, safer, more reliable, or cheaper than others.
• Goals just provide a crude distinction between "happy" and "unhappy" states, whereas a more general performance measure should allow a comparison of different world states (or sequences of states) according to exactly how happy they would make the agent if they could be achieved.
• Utility function map a state (or a sequence of states) onto a real number, which describes the associated degree of happiness.
Utility-based agents

Diagram showing the flow of information from the environment to the agent, including state, how the world evolves, what my actions do, utility, 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, and effectors.
Learning agent

- Four components:
  - CRITIC (評論者)
  - Learning Element
  - Performance Element
  - Problem Generator
Learning agents
Mobile agents

- Programs that can migrate from one machine to another.
- Execute in a platform-independent execution environment.
- Require agent execution environment (places).
- Mobility not necessary or sufficient condition for agenthood.
- Practical but non-functional advantages:
  - Reduced communication cost (e.g., from PDA)
  - Asynchronous computing (when you are not connected)
- Two types:
  - One-hop mobile agents (migrate to one other place)
  - Multi-hop mobile agents (roam the network from place to place)
- Applications:
  - Distributed information retrieval.
  - Telecommunication network routing.

AI chapter 2
Mobile agents

• Programs that can migrate from one machine to another.
• Execute in a platform-independent execution environment.
• Require agent execution environment (places).
• Mobility not necessary or sufficient condition for agenthood.

A mail agent
Summary

• Intelligent Agents:
  • Anything that can be *viewed as perceiving* its *environment* through *sensors* and *acting* upon that environment through its *effectors* to maximize progress towards its *goals*.
  • PAGE (Percepts, Actions, Goals, Environment)
  • Described as a Perception (sequence) to Action Mapping: \( f : P^* \rightarrow A \)
  • Using look-up-table, closed form, etc.

• **Agent Types:** Reflex, state-based, goal-based, utility-based

• **Rational Action:** The action that maximizes the expected value of the performance measure *given the percept sequence to date*