L2. Introduction (2)

- What are intelligent agents?
- What are the features of an intelligent agent?
- How to design an intelligent agent?
- An example
- Demo systems
An agent is anything that can viewed as *perceiving* its environment through *sensors* and *acting* upon that environment through *effectors*.

**Examples:**
- a human driver
- a robot driver
- a driver based on programs

A generic agent diagram
What are intelligent agents

- An agent always requires a certain amount of intelligence to perform its tasks. Consequently, one refers to intelligent agent.

- At the highest level, three major categories of agents can be distinguished: human agents, hardware agents, and software agents.
- For example, a human travel agent, a robot, an automated taxi.
Software intelligent agents

- Intelligent software agents are defined as being a software program that can perform specific tasks for a user and possesses a degree of intelligence that permits it to perform parts of its tasks autonomous in a useful manner.
- What is intelligence that refers to software agents?
  - A software program that think like humans
  - A software program that act like humans
  - A software program that think rationally
  - A software program that act rationally

A very wide variation in the area of intelligence can be envisaged that ranges from simple agents with limited intelligence through to complex, highly-intelligent systems.

- The field of AI (artificial intelligence), attempts to understand intelligent entities.
The characteristics of intelligent agents

- Internal characteristics are
  - Learning/reasoning:
    an agent has the ability to learn from previous experience and to successively adapt its own behavior to the environment.
  - Reactivity:
    an agent must be capable of reacting appropriately to influences or information from its environment.
  - Autonomy:
    an agent must have both control over its actions and internal states. The degree of the agent’s autonomy can be specified. There may need intervention from the user only for important decisions.
  - Goal-oriented:
    an agent has well-defined goals and gradually influence its environment and so achieve its own goals.
continue …

- External characteristics are
  - communication:
    an agent often requires an interaction with its environment to fulfill its tasks, such as human, other agents, and arbitrary information sources.
  - cooperation:
    cooperation of several agents permits faster and better solutions for complex tasks that exceed the capabilities of a single agent.
  - mobility:
    an agent may navigate within electronic communication networks.
  - Character:
    like human, an agent may demonstrate an external behavior with many human characters as possible.

- What are software agents different from traditional software programs?
  - **Perception**: Software agents are aware of environment changes
  - **Autonomy**: Software agents perform tasks largely autonomously
  - **Learning**: Software agents are accumulating knowledge (learning through experience)
  - **Communication**: Software agents communicate with users and/or other agents
Areas of influence

Characteristics

- Decision theory
- Autonomy
- Learning reasoning
- Reactivity
- Character
- Psychology
- Distributed AI
- Communication
- Mobility
- Cooperation
- Network communication

Artificial Intelligence (AI)
Applications

- information retrieval and filtering
- news watcher
- friend-making agents
- shopping agents
- scheduling agents
- Web document maintenance agents

......
How to design the agent program

- agent = architecture + agent program
  - The architecture, in general,
    - makes the percepts from the sensors available to the program,
    - runs the program,
    - feeds the program action’s choices to the effectors
  - architecture may be
    - a plain computer
    - a special-purpose hardware
    - some software
  - The agent program is a function that implements agent mapping from percepts to actions. It is run on the architecture.
An example: designing an automated taxi driver

<table>
<thead>
<tr>
<th>Percepts</th>
<th>cameras, speedometer, GPS, sonar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actions</td>
<td>steer, accelerate, brake</td>
</tr>
<tr>
<td>Goals</td>
<td>Safely to destination</td>
</tr>
<tr>
<td>Environment</td>
<td>traffic light, other traffic, pedestrians, in Japan</td>
</tr>
</tbody>
</table>

The taxi driver agent and its PAGE description

Four types of agent program:

- Simple reflex agents
- Agents that keep track of the world
- Goal-based agents
- Utility-based agents
**Simple reflex agents**

It works by finding a rule whose condition matches the current situation and then doing the action associated with that rule.

```
function SRAgent(percept)
return action

static: rules, a set of condition-action rules

state <- INTERPRET-INPUT(percept)
rule <- RULE-MATCH(state, rules)
action <- RULE-ACTION[rule]
return action
```
Agents with awareness of the world

It works by finding a rule whose condition matches the current situation (as defined by the percept and the stored internal state) and then doing the action associated with that rule. (car model evolving)

**function** SRSAgent(percept) **return** action

**static**: 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
Goal-based agents

It works by decision making towards to the final goals.

function Goal-Agent(percept) return action

static: current world state
rules, a set of condition-action rules

state <- UPDATE-STATE(state, percept)
do { rule <- RULE-MATCH(state, rules)
    action <- RULE-ACTION[rule]
    state <- UPDATE-STATE(state, action)
} while (not goals? <- state)
return action
Utility-based agents

It works by evaluating each decision in quantity. Utility is a function that map a state onto a real number, which describes the associated quality.

```
function Utility-Agent(percept) return action

    static: current world state
    rules, a set of condition-action rules

    state <- UPDATE-STATE(state, percept)
    do { rule <- RULE-MATCH(state, rules)
        action <- RULE-ACTION[rule]
        state <- UPDATE-STATE(state, action)
    } while ((not goals? <- state)||(not good quality?))

    return action
```
Properties of environment

- Accessible vs. inaccessible
- Deterministic vs. non-deterministic
- Episodic vs. non-episodic
- Static vs. dynamic
- Discrete vs. continuous
Multi-agent system

- Tamura-san’s smart eco-home
Simulated home

Data collection

<table>
<thead>
<tr>
<th>Space</th>
<th>Person</th>
<th>Device</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>living</td>
<td>Bob</td>
<td>light</td>
<td>19:52</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner</td>
<td>19:53</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner, refrigerator</td>
<td>19:55</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner, refrigerator, TV</td>
<td>19:58</td>
</tr>
</tbody>
</table>

Activity matrices based data mining

<table>
<thead>
<tr>
<th>Hand</th>
<th>Hip</th>
<th>Sofa</th>
<th>Cup</th>
<th>Air conditioner</th>
<th>TV</th>
<th>Remote controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

People’s profile

<table>
<thead>
<tr>
<th>Person</th>
<th>Device</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>light</td>
<td>19:52</td>
</tr>
<tr>
<td>Bob</td>
<td>air-conditioner</td>
<td>19:53</td>
</tr>
<tr>
<td>Bob</td>
<td>refrigerator</td>
<td>19:55</td>
</tr>
<tr>
<td>Bob</td>
<td>TV</td>
<td>19:58</td>
</tr>
</tbody>
</table>

Space log

<table>
<thead>
<tr>
<th>Space</th>
<th>Person</th>
<th>Device</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>living</td>
<td>Bob</td>
<td>light</td>
<td>19:52</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner</td>
<td>19:53</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner, refrigerator</td>
<td>19:55</td>
</tr>
<tr>
<td>living</td>
<td>Bob</td>
<td>light, air-conditioner, refrigerator, TV</td>
<td>19:58</td>
</tr>
</tbody>
</table>

Device log

<table>
<thead>
<tr>
<th>Device</th>
<th>State</th>
<th>Time</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>light</td>
<td>on</td>
<td>19:52</td>
<td></td>
</tr>
<tr>
<td>air-conditioner</td>
<td>on</td>
<td>25℃</td>
<td>19:53</td>
</tr>
<tr>
<td>refrigerator</td>
<td>on</td>
<td>open</td>
<td>19:55</td>
</tr>
<tr>
<td>TV</td>
<td>on</td>
<td>ch1</td>
<td>19:58</td>
</tr>
</tbody>
</table>

Person log

<table>
<thead>
<tr>
<th>Person</th>
<th>Device</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>light</td>
<td>19:52</td>
</tr>
<tr>
<td>Bob</td>
<td>air-conditioner</td>
<td>19:53</td>
</tr>
<tr>
<td>Bob</td>
<td>refrigerator</td>
<td>19:55</td>
</tr>
<tr>
<td>Bob</td>
<td>TV</td>
<td>19:58</td>
</tr>
</tbody>
</table>

Intelligent Entity Pool

Extracting people’s activity

Bob is watching TV.

Refrigerator is opened!!!

What is Bob doing?

Refrigerator is opened.

Agent make advice to close

Recommendation engine

make suggestions

Agent

Hand Hip Sofa Cup Air

conditioner

TV Remote

controller

Hand 1 0 0 0 0 0 1
Hip 1 1 0 0 0 0 0
Sofa - 1 0 0 0 0 0
Cup - - 1 0 0 0 0
Air - - - 1 0 1 conditioner
TV - - - - 1 1
Remote controller - - - - - - 1

Drinking

<table>
<thead>
<tr>
<th>Hand</th>
<th>Hip</th>
<th>Sofa</th>
<th>Cup</th>
<th>Air conditioner</th>
<th>TV</th>
<th>Remote controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Watching TV
Real world applications

- Demo
  - Searching for a solution
  - Game: computer as player
  - Smart garden
  - Treasure finding agent
  - Maze Robot: find a way out
  - Automatic conversation
Work in class/home work

- Run 1-6 demo applets from my course home page [https://cis.k.hosei.ac.jp/~rhuang/](https://cis.k.hosei.ac.jp/~rhuang/)
- if you are an indoor garden designer, how will you design a smart indoor garden? Please describe your smart indoor garden system.