CS 4804: Introduction to Artificial Intelligence

Virginia Tech CS 4804 Fall 2021
Instructor: Yinlin Chen
Topics

- Intelligent Agents
- Project 0
Agents and Environments

- **Agent**: computer program expected to be intelligent
Agent Functions

- Agent’s behavior is described by the agent function
- Input: percept sequence
- Output: Agent’s actual response to any sequence of percepts
Agent Programs

• A concrete implementation of agent function
• Runs on some machines (physical system).
• Agent function depends on
  – Machines (have limited on physical resource)
  – Program (how well did it write)
Example: Vacuum world

- Percepts: [location, status], e.g., [A, Dirty]
- Actions: Left, Right, Suck, NoOp
### Vacuum cleaner agent

#### Agent function

<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>
<tr>
<td>[A, Clean], [A, Clean], [A, Clean]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, Clean], [A, Clean], [A, Dirty]</td>
<td>Suck</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

#### Agent program

```python
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 agent function?

Can it be implemented by a small agent program?
Rational agent

- Rational: **Maximally** achieving pre-defined goals
- Select an action to **maximize** its (expected) utility
  - based on initial knowledge
  - learn from what it perceives
- Utility function: an internalization of the **performance measure**
- Succeed in a vast variety of **environments**
- Should be autonomous
- Being rational means **maximizing** your **expected utility**
Task Environment

• PEAS: the problem specification for the task environment
• The rational agent we want to design for this task environment is the solution.
• PEAS stands for:
  – Performance measures
  – Environment
  – Actuators
  – Sensors
The Task environment - PEAS

- **Performance measure**
  - -1 per step; +10 food; +500 win; -500 die; +200 hit scared ghost

- **Environment**
  - Pacman dynamics (incl ghost behavior)

- **Actuators**
  - Left Right Up Down

- **Sensors**
  - Entire state is visible (except power pellet duration)
PEAS: Automated Taxi

- Performance measure
- Environment
- Actuators
- Sensors
Properties of Task Environments

- Observability (Full or partially or unobservable)
- Single-agent or multi-agent
- Deterministic or nondeterministic or stochastic
- Episodic or sequential
- Static or dynamic or semi-dynamic
- Discrete or continuous
- Known or unknown
<table>
<thead>
<tr>
<th>Task Environment</th>
<th>Observable</th>
<th>Agents</th>
<th>Deterministic</th>
<th>Episodic</th>
<th>Static</th>
<th>Discrete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crossword puzzle</td>
<td>Fully</td>
<td>Single</td>
<td>Deterministic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Chess with a clock</td>
<td>Fully</td>
<td>Multi</td>
<td>Deterministic</td>
<td>Sequential</td>
<td>Semi</td>
<td>Discrete</td>
</tr>
<tr>
<td>Poker</td>
<td>Partially</td>
<td>Multi</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Backgammon</td>
<td>Fully</td>
<td>Multi</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Taxi driving</td>
<td>Partially</td>
<td>Multi</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Dynamic</td>
<td>Continuous</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>Partially</td>
<td>Single</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Dynamic</td>
<td>Continuous</td>
</tr>
<tr>
<td>Image analysis</td>
<td>Fully</td>
<td>Single</td>
<td>Deterministic</td>
<td>Episodic</td>
<td>Semi</td>
<td>Continuous</td>
</tr>
<tr>
<td>Part-picking robot</td>
<td>Partially</td>
<td>Single</td>
<td>Stochastic</td>
<td>Episodic</td>
<td>Dynamic</td>
<td>Continuous</td>
</tr>
<tr>
<td>Refinery controller</td>
<td>Partially</td>
<td>Single</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Dynamic</td>
<td>Continuous</td>
</tr>
<tr>
<td>English tutor</td>
<td>Partially</td>
<td>Multi</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Dynamic</td>
<td>Discrete</td>
</tr>
</tbody>
</table>
Agent design

- The environment type largely determines the agent design
  - *Partially observable* => agent requires *memory* (internal state)
  - *Stochastic* => agent may have to prepare for *contingencies*
  - *Multi-agent* => agent may need to behave *randomly*
  - *Static* => agent has time to compute a rational decision
  - *Continuous time* => continuously operating *controller*
  - *Unknown physics* => need for *exploration*
  - *Unknown perf. measure* => observe/interact with *human principal*
Agent types

• Simple reflex agents
• Model-based agents
• Goal-based agents
• Utility-based agents
• Learning agents

An environment and the agents that reside within it create a world
Simple reflex agent

- direct input-output
Model-based reflex agent

- Store internal state information and keep track of the state of the world
Goal-based agent

- Want to achieve its goals
Utility-based agent

• Want to improve utility function
Learning agent

• Adapt to previous experience and data
Spectrum of representations

(a) Atomic

(b) Factored

(c) Structured
Outline of the course

First-order logic

- Atomic
- Factored
- Structured
- Deterministic
- Stochastic
- Known
- Unknown

Logic
Search
MDPs
Bayes nets
RL
Summary

• An **agent** interacts with an **environment** through **sensors** and **actuators**

• The **agent function**, implemented by an **agent program** running on a machine, describes what the agent does in all circumstances

• **Rational** agents choose actions that **maximize their expected utility**

• **PEAS** descriptions define task environments; precise PEAS specifications are essential and strongly influence agent designs

• More difficult environments require more complex agent designs and more sophisticated representations
Project 0

- Release today. Check course website
  - Due 09/03 11:59pm
Reading and Next Class

• Agents: AIMA Ch. 2.1 – 2.4
• Next class:
  – Uninformed Search: AIMA Ch. 3.1-3.4