Intelligent Agents

CMSC 421: Chapter 2
Russell & Norvig’s book is organized around the concept of *intelligent agents*:

- humans, robots, softbots, thermostats, etc.

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

\[ f : \mathcal{P}^* \rightarrow \mathcal{A} \]

The *agent program* runs on the physical *architecture* to produce \( f \)
Chapter 2 - Purpose and Outline

Purpose of Chapter 2:  
- basic concepts relating to agents

- Agents and environments
- Rationality
- The PEAS model of an environment
- Environment types
- Agent types
Vacuum-cleaner world

Percepts: location and contents, e.g., \([A, Dirty]\)

Actions: \(Left, Right, Suck, NoOp\)
A vacuum-cleaner agent

<table>
<thead>
<tr>
<th>Percept sequence</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>[A, \text{Clean}]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, \text{Dirty}]</td>
<td>Suck</td>
</tr>
<tr>
<td>[B, \text{Clean}]</td>
<td>Left</td>
</tr>
<tr>
<td>[B, \text{Dirty}]</td>
<td>Suck</td>
</tr>
<tr>
<td>[A, \text{Clean}], [A, \text{Clean}]</td>
<td>Right</td>
</tr>
<tr>
<td>[A, \text{Clean}], [A, \text{Dirty}]</td>
<td>Suck</td>
</tr>
<tr>
<td>\vdots</td>
<td>\vdots</td>
</tr>
</tbody>
</table>

function \textsc{Reflex-Vacuum-Agent}( [location, status] ) returns an action

\begin{align*}
\text{if } \text{status} &= \text{Dirty} \text{ then return } \text{Suck} \\
\text{else if } \text{location} &= A \text{ then return } \text{Right} \\
\text{else if } \text{location} &= B \text{ then return } \text{Left}
\end{align*}

What is the \textbf{right} function?
Can it be implemented in a small agent program?
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 $\neq$ omniscient
- percepts may not supply all relevant information
Rational $\neq$ clairvoyant
- action outcomes may not be as expected
Hence, rational $\neq$ successful

Rational $\Rightarrow$ exploration, learning, autonomy
PEAS

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

Consider, e.g., the task of designing an automated taxi:

Performance measure?

Environment?

Actuators?

Sensors?
PEAS

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

Consider, e.g., the task of designing an automated taxi:

**Performance measure?** safety, destination, profits, legality, comfort, . . .

**Environment?** 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?

Environment?

Actuators?

Sensors?
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 form

**Sensors?** HTML pages (text, graphics, scripts)
## Environment types

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Solitaire</td>
<td>Backgammon</td>
<td>Internet shopping</td>
<td>Taxi</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fully observable?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Static?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Discrete?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single-agent?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fully observable?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic?</strong></td>
<td>Yes*</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic?</strong></td>
<td>Yes*</td>
<td>No</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Static?</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Discrete?</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Single-agent?</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*After the cards have been dealt

**Episodic**: task divided into atomic episodes, each to be considered by itself  
**Sequential**: Current decision may affect all future decisions
### Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fully observable?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic?</strong></td>
<td>Yes*</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Static?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Discrete?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single-agent?</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*After the cards have been dealt

*Static*: the world does not change while the agent is thinking
### Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully observable?</td>
<td>No</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></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single-agent?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*After the cards have been dealt*
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fully observable?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic?</strong></td>
<td>Yes*</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Static?</strong></td>
<td>Yes</td>
<td>Semi</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td><strong>Discrete?</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single-agent?</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*After the cards have been dealt*
## Environment types

<table>
<thead>
<tr>
<th></th>
<th>Solitaire</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fully observable?</strong></td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Deterministic?</strong></td>
<td>Yes*</td>
<td>No</td>
<td>Partly</td>
<td>No</td>
</tr>
<tr>
<td><strong>Episodic?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td><strong>Static?</strong></td>
<td>Yes</td>
<td>Semi</td>
<td>Semi</td>
<td>No</td>
</tr>
<tr>
<td><strong>Discrete?</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Single-agent?</strong></td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

*After the cards have been dealt

The environment type largely determines the agent design

Real world: partially observable, stochastic, sequential, dynamic, continuous, multi-agent
Agent types

Four basic types in order of increasing generality:
– simple reflex agents
– reflex agents with state
– goal-based agents
– utility-based agents

All of these can be turned into learning agents
Simple reflex agents

Agent

Environment

Sensors

What the world is like now

Condition–action rules

What action I should do now

Actuators
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
Reflex agents with state

Agent

Environment

Sensors

State

How the world evolves

What my actions do

Condition–action rules

What the world is like now

What action I should do now

Actuators
Example

Suppose the percept didn’t tell the agent what room it’s in. Then the agent could remember its location:

```plaintext
function VACUUM-AGENT-WITH-STATE([location]) returns an action
static: location, initially A
    if status = Dirty then return Suck
    else if location = A then
        location ← B
        return Right
    else if location = B then
        location ← A
        return Left
```

Above, I’ve assumed we know the agent always starts out in room \textit{A}. What if we didn’t know this?
Goal-based agents

- Agent
  - Environment
  - Sensors
  - State
  - How the world evolves
  - What the world is like now
  - What my actions do
  - What it will be like if I do action A
  - Goals
  - What action I should do now
  - Actuators

CMSC 421: Chapter 2  23
Utility-based agents

- Agent
  - State
  - How the world evolves
  - What my actions do
  - Utility

- Environment
  - Sensors
  - 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

- Actuators
Learning agents

- Performance standard
- Agent
- Environment
- Sensors
- Performance element
- Changes
- Knowledge
- Learning goals
- Problem generator
- Learning element
- Performance element
- Feedback
- Critic
- Actuators
- Environment
**Summary**

*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

*PEAS* descriptions define task environments

Environments are categorized along several dimensions:


Some basic agent architectures:

- *reflex, reflex with state, goal-based, utility-based*