Agents

What’s an agent?

- Russell and Norvig:
  “An agent is anything that can be viewed as perceiving its environment through sensors and acting on that environment through actuators.” (p. 32)
- Examples:
  - Taxi driver
  - Internet shopper
  - Backgammon player
  - Chemical plant controller
  - Spam detector

The agent and the environment

An agent:
- Works in a particular environment
- Has goals
- Perceives the environment
- Performs actions to achieve its goals.

Formalizing Task Environments (PEAS)

- P: Performance
  *This is all important: it defines the goal*
- E: Environment
  *This defines the world the agent lives in*
- A: Actuators
  *This defines how the agent can change the world*
- S: Sensors
  *This defines how the agent sees the world (and how much of the world the agent can see)*

Example: the automated taxi driver

- Possible performance measures:
  - Safe, fast, legal, comfortable trip, maximize profit.
- Environment:
  - Roads, other traffic, pedestrians, other traffic
- Actuators:
  - Steering, accelerator, brake, turn signals, horn
- Sensors:
  - Cameras, engine sensors, laser rangefinders, GPS, keyboard, microphone

Environments

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>Fully</td>
<td>Fully</td>
<td>?</td>
<td>partially</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Episodic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discrete</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Agents</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Environments

**Deterministic vs. stochastic:** is the next environment state completely determined by the current state?

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Deterministic</td>
<td>deterministic</td>
<td>stochastic</td>
<td>?</td>
<td>stochastic</td>
</tr>
</tbody>
</table>

**Episodic vs. sequential:** can the agent’s experience be divided into steps where the agent’s action depends only on the current episode?

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Deterministic</td>
<td>deterministic</td>
<td>stochastic</td>
<td>?</td>
<td>stochastic</td>
</tr>
<tr>
<td>Episodic</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
</tr>
<tr>
<td>Static</td>
<td>static</td>
<td>static</td>
<td>dynamic</td>
<td>dynamic</td>
</tr>
<tr>
<td>Discrete</td>
<td>discrete</td>
<td>discrete</td>
<td>continuous</td>
<td>continuous</td>
</tr>
<tr>
<td>Agents</td>
<td>single</td>
<td>multi</td>
<td>multi</td>
<td>multi</td>
</tr>
</tbody>
</table>

**Static vs. dynamic:** can the environment change while the agent is choosing an action?

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Deterministic</td>
<td>deterministic</td>
<td>stochastic</td>
<td>?</td>
<td>stochastic</td>
</tr>
<tr>
<td>Episodic</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
</tr>
<tr>
<td>Static</td>
<td>static</td>
<td>static</td>
<td>dynamic</td>
<td>dynamic</td>
</tr>
<tr>
<td>Discrete</td>
<td>discrete</td>
<td>discrete</td>
<td>continuous</td>
<td>continuous</td>
</tr>
<tr>
<td>Agents</td>
<td>single</td>
<td>multi</td>
<td>multi</td>
<td>multi</td>
</tr>
</tbody>
</table>

**Discrete vs. continuous:** This distinction can be applied to the state of the environment, the way time is handled and to the percepts/actions of the agent.

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Deterministic</td>
<td>deterministic</td>
<td>stochastic</td>
<td>?</td>
<td>stochastic</td>
</tr>
<tr>
<td>Episodic</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
</tr>
<tr>
<td>Static</td>
<td>static</td>
<td>static</td>
<td>dynamic</td>
<td>dynamic</td>
</tr>
<tr>
<td>Discrete</td>
<td>discrete</td>
<td>discrete</td>
<td>continuous</td>
<td>continuous</td>
</tr>
<tr>
<td>Agents</td>
<td>single</td>
<td>multi</td>
<td>multi</td>
<td>multi</td>
</tr>
</tbody>
</table>

**Single vs. multi-agent:** Does the environment contain other agents who are also maximizing some performance measure that depends on the current agent’s actions?

<table>
<thead>
<tr>
<th></th>
<th>Crossword puzzle</th>
<th>Backgammon</th>
<th>Internet shopping</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observable</td>
<td>fully</td>
<td>fully</td>
<td>?</td>
<td>partially</td>
</tr>
<tr>
<td>Deterministic</td>
<td>deterministic</td>
<td>stochastic</td>
<td>?</td>
<td>stochastic</td>
</tr>
<tr>
<td>Episodic</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
<td>sequential</td>
</tr>
<tr>
<td>Static</td>
<td>static</td>
<td>static</td>
<td>dynamic</td>
<td>dynamic</td>
</tr>
<tr>
<td>Discrete</td>
<td>discrete</td>
<td>discrete</td>
<td>continuous</td>
<td>continuous</td>
</tr>
<tr>
<td>Agents</td>
<td>single</td>
<td>multi</td>
<td>multi</td>
<td>multi</td>
</tr>
</tbody>
</table>

---

**The vacuum world**

- Environment: square A and B
- Percepts: [location and content] e.g. [A, Dirty]
- Actions: left, right, suck, no-op
The vacuum world (cont.)

<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>
</tbody>
</table>

A simple agent function

```python
def REFLEX-VACUUM-AGENT (location, status) :
    if status == Dirty:
        return Suck
    elif location == A:
        return Right
    elif location == B:
        return Left
Is this the best agent for the job?
```

Rational agents

- A rational agent is one that does the right thing.
- What is the right thing?
  - Approximation: the most successful agent.
  - Measure of success?
- Performance measure according to what is wanted in the environment (goal).
- Performance measures for the vacuum world:
  - The amount of dirt cleaned per unit time.
  - How much energy was spent on moving and cleaning.

Rationality

- What is rational at a given time depends on:
  - Performance measure.
  - The available actions.
  - The built-in knowledge about the environment.
  - Percept sequence to date (sensors).
- A rational agent chooses an action which maximizes the expected value of the performance measure given the percept sequence and its built-in knowledge.

Agent structure

- Agent: architecture + program
- The agent program: maps percepts to actions
- The agent program receives as input the current percept and returns an action for the agent's actuators.

Agent types

Function TABLE-DRIVEN_AGENT(percept) returns an action

```python
static: percepts, a sequence, initially empty
table, a table of actions, indexed by percept sequences
append.percept to percepts
action ← LOOKUP(percepts, table)
return action
Why won’t this work?
```
Simple reflex agent

- Selects action only on the basis of the current percept
- Large reduction in possible percept/action combinations

```plaintext
function REFLEX-VACUUM-AGENT ((location, status))
    if status == Dirty then return Suck
    else if location == A then return Right
    else if location == B then return Left
```

Simple reflex agent (cont)

function SIMPLE-REFLEX-AGENT (percept) returns an action

```plaintext
static: rules, a set of condition-action rules
state ← INTERPRET-INPUT (percept)
rule ← RULE-MATCH (state, rule)
action ← RULE-ACTION (rule)
return action
```

Will only work if the environment is fully observable

Our taxi-driver agent would not work as a simple reflex agent!

Model-based reflex agent

- Maintain an internal state
- Update the state using information on “how the world works” (the model of the world)
- Taxi-driver agent: needs to keep track of cars in his blind spot

Goal-based agents

- Our taxi-driver agent needs to get somewhere: it has a goal. Chooses actions to achieve goal.
- Search and planning are subfields of AI devoted to finding a sequence of actions that achieve the agent’s goals.

Utility-based agents

- Utility function maps a (sequence of) state(s) onto a real number.
- Certain goals can be reached in different ways. Some are better, have a higher utility.
- Improves on goals:
  - Selecting between conflicting goals
  - Select appropriately between several goals that have varying probability of success.

Learning agents

- All previous agent-programs describe methods for selecting actions.
  - Yet it does not explain the origin of these programs.
  - Learning mechanisms can be used to perform this task.
  - Advantage is the robustness of the program toward initially unknown environments.
Learning agents

- Performance element: selects actions based on percepts.
  - Corresponds to the previous agent programs.
- Learning element: introduce improvements in performance element.
  - Critic provides feedback on agents performance based on fixed performance standard.
- Problem generator: suggests actions that will lead to new and informative experiences.
  - Exploration vs. exploitation