Intelligent 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
Agents and environments

- 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 \( f \)

  - agent = architecture + program
Vacuum-cleaner world

- Percepts: location and contents, e.g., [A,Dirty]
- Actions: Left, Right, Suck, NoOp
- Agent’s function → table
  - For many agents this is a very large table

<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>
</tbody>
</table>
Rational agents

- **Rationality**
  - Performance measuring success
  - Agents prior knowledge of environment
  - Actions that agent can perform
  - Agent’s percept sequence to date

- **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.
Rationality

- Rational is different to omniscient
  - Percepts may not supply all relevant information

- Rational is different to being perfect
  - Rationality maximizes expected outcome while perfection maximizes actual outcome.
Autonomy in Agents

The **autonomy** of an agent is the extent to which its behaviour is determined by its own experience.

- **Extremes**
  - No autonomy – ignores environment/data
  - Complete autonomy – must act randomly/no program
- **Example**: baby learning to crawl
- **Ideal**: design agents to have some autonomy
  - Possibly good to become more autonomous in time
**PEAS**

- **PEAS**: Performance measure, Environment, Actuators, Sensors

- 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**: 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
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)
- Deterministic (vs. stochastic)
- Episodic (vs. sequential)
- Static (vs. dynamic)
- Discrete (vs. continuous)
- Single agent (vs. multiagent)
Fully observable (vs. partially observable)

- Is everything an agent requires to choose its actions available to it via its sensors?
  - If so, the environment is fully accessible
- If not, parts of the environment are inaccessible
  - Agent must make informed guesses about world

<table>
<thead>
<tr>
<th>Cross Word</th>
<th>Poker</th>
<th>Backgammon</th>
<th>Taxi driver</th>
<th>Part parking robot</th>
<th>Image analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully</td>
<td>Partially</td>
<td>Partially</td>
<td>Partially</td>
<td>Fully</td>
<td>Fully</td>
</tr>
</tbody>
</table>

Artificial Intelligence a modern approach
Deterministic (vs. stochastic)

- Does the change in world state
  - Depend only on current state and agent’s action?
- Non-deterministic environments
  - Have aspects beyond the control of the agent
  - Utility functions have to guess at changes in world

<table>
<thead>
<tr>
<th>Cross Word</th>
<th>Poker</th>
<th>Backgammon</th>
<th>Taxi driver</th>
<th>Part parking robot</th>
<th>Image analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Deterministic</td>
</tr>
</tbody>
</table>
Episodic (vs. sequential):

- Is the choice of current action
  - Dependent on previous actions?
  - If not, then the environment is episodic

- In non-episodic environments:
  - Agent has to plan ahead:
    - Current choice will affect future actions

<table>
<thead>
<tr>
<th>Cross Word</th>
<th>Poker</th>
<th>Backgammon</th>
<th>Taxi driver</th>
<th>Part parking robot</th>
<th>Image analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequential</td>
<td>Sequential</td>
<td>Sequential</td>
<td>Sequential</td>
<td>Episodic</td>
<td>Episodic</td>
</tr>
</tbody>
</table>
Static (vs. dynamic):

- **Static environments don’t change**
  - While the agent is deliberating over what to do

- **Dynamic environments do change**
  - So agent should/could consult the world when choosing actions
  - Alternatively: anticipate the change during deliberation OR make decision very fast

- **Semidynamic**: If the environment itself does not change with the passage of time but the agent's performance score does)

<table>
<thead>
<tr>
<th>Cross Word</th>
<th>Poker</th>
<th>Backgammon</th>
<th>Taxi driver</th>
<th>Part parking robot</th>
<th>Image analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>Static</td>
<td>Static</td>
<td>Dynamic</td>
<td>Dynamic</td>
<td>Semi</td>
</tr>
</tbody>
</table>
Discrete (vs. continuous)

- A limited number of distinct, clearly defined percepts and actions or a big range of values (continuous)
Single agent (vs. multiagent):

- An agent operating by itself in an environment or there are many agents working together

<table>
<thead>
<tr>
<th>Cross Word</th>
<th>Poker</th>
<th>Backgammon</th>
<th>Taxi driver</th>
<th>Part parking robot</th>
<th>Image analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td>Multi</td>
<td>Multi</td>
<td>Multi</td>
<td>Single</td>
<td>Single</td>
</tr>
<tr>
<td></td>
<td>Observable</td>
<td>Deterministic</td>
<td>Episodic</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>---------------</td>
<td>----------</td>
<td>--------</td>
<td>----------</td>
</tr>
<tr>
<td>Cross Word</td>
<td>Fully</td>
<td>Deterministic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Poker</td>
<td>Fully</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Backgammon</td>
<td>Partially</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Static</td>
<td>Discrete</td>
</tr>
<tr>
<td>Taxi driver</td>
<td>Partially</td>
<td>Stochastic</td>
<td>Sequential</td>
<td>Dynamic</td>
<td>Conti</td>
</tr>
<tr>
<td>Part parking robot</td>
<td>Partially</td>
<td>Stochastic</td>
<td>Episodic</td>
<td>Dynamic</td>
<td>Conti</td>
</tr>
<tr>
<td>Image analysis</td>
<td>Fully</td>
<td>Deterministic</td>
<td>Episodic</td>
<td>Semi</td>
<td>Conti</td>
</tr>
</tbody>
</table>
Agent types

- Four basic types in order of increasing generality:
  - Simple reflex agents
  - Reflex agents with state
  - Goal-based agents
  - Utility-based agents
  - All these can be turned into learning agents
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
Simple reflex agents

- Simple but very limited intelligence
- Infinite loops
  - Suppose vacuum cleaner does not keep track of location. What do you do on clean left of A or right on B is infinite loop
  - Randomize action
- Chess – openings, endings
  - Lookup table (not a good idea in general)
    - $35^{100}$ entries required for the entire game
Model-based reflex agents

- Know how world evolves
  - Overtaking car gets closer from behind
- How agents actions affect the world
  - Wheel turned clockwise takes you right
- Model base agents update their state

```
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
```
Goal-based agents

- knowing state and environment? Enough?
  - Taxi can go left, right, straight
- Have a goal
  - A destination to get to
- Uses knowledge about a goal to guide its actions
  - E.g., Search, planning
Goal-based agents

- Reflex agent breaks when it sees brake lights. Goal based agent reasons
  - Brake light -> car in front is stopping -> I should stop -> I should use brake
Utility-based agents

- Goals are not always enough
  - Many action sequences get taxi to destination
  - Consider other things. How fast, how safe.....

- A utility function maps a state onto a real number which describes the associated degree of happiness.
Utility-based agents
Learning agents

- Performance element is what was previously the whole agent
  - Input sensor
  - Output action

- Learning element
  - Modifies performance element
Learning agents

- Critic: how the agent is doing
  - Input: checkmate?
  - Fixed

- Problem generator
  - Tries to solve the problem differently instead of optimizing
Learning agents (Taxi driver)

- **Performance element**
  - How it currently drives

- Taxi driver Makes quick left turn across 3 lanes
  - Critics observe shocking language by passenger and other drivers and informs bad action
  - Learning element tries to modify performance elements for future
  - Problem generator suggests experiment out something called Brakes on different Road conditions

- Critics is not always easy
  - shocking language
  - Less tip
  - Less passengers