Artificial Intelligence
Agents and Environments

Instructor: Dr. B. John Oommen

Chancellor's Professor
Fellow: IEEE; Fellow: IAPR
School of Computer Science, Carleton University, Canada.

1The primary source of these notes are the slides of Professor Hwee Tou Ng from Singapore. I sincerely thank him for this.
Intelligent, Autonomous Agents

Agent
- Anything that can be viewed as perceiving its environment
- Perception done through sensors
- 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
Intelligent, Autonomous Agents

- **Agent**
  - Anything that can be viewed as perceiving its environment
  - Perception done through sensors
  - 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
Intelligent, Autonomous Agents

Agent
- Anything that can be viewed as perceiving its environment
- Perception done through sensors
- 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
Agent: Mapping: Percept Sequences $\Rightarrow$ Actions
Agent Function

Maps from percept histories to actions: \( F : P^* \rightarrow A \)

Agent Program

Runs on the physical architecture to produce \( F \)

Agent = Architecture + Program

Vacuum Cleaner Agent

- Percepts: Location and Contents: \{[LocA, Dirty], ... \}
- Actions: Left, Right, Suck, VacuumOn, VacuumOff
- Agent: Function(PerceptHistory, Vacuum-agent-function-table)
Agent Function
Maps from percept histories to actions: \([F : P^* \rightarrow A]\)

Agent Program
Runs on the physical architecture to produce F

Agent = Architecture + Program

Vacuum Cleaner Agent
- Percepts: Location and Contents: \{[LocA, Dirty], ... \}
- Actions: Left, Right, Suck, VacuumOn, VacuumOff
- Agent:
  Function(PerceptHistory, Vacuum-agent-function-table)
Agent Function
Maps from percept histories to actions: $[F : P^* \rightarrow A]$

Agent Program
Runs on the physical architecture to produce F

Agent = Architecture + Program

Vacuum Cleaner Agent
- Percepts: Location and Contents: {[LocA, Dirty], ... }
- Actions: Left, Right, Suck, VacuumOn, VacuumOff
- Agent: Function(PerceptHistory, Vacuum-agent-function-table)
Agents... 

- Agent Function
  - Maps from percept histories to actions: \( F : P^* \rightarrow A \)

- Agent Program
  - Runs on the physical architecture to produce \( F \)

- Agent = Architecture + Program

- Vacuum Cleaner Agent
  - **Percepts**: Location and Contents: \{[LocA, Dirty], ... \}
  - **Actions**: Left, Right, Suck, VacuumOn, VacuumOff
  - **Agent**: Function(PerceptHistory, Vacuum-agent-function-table)
Agent should strive to “do the right thing”:
- Based on what it can perceive and actions it can do

- The “right action”:
  - One that will cause the agent to be “most successful”

Performance measure:
- Objective criterion for success of an agent’s behavior

Performance of a vacuum-cleaner agent could be:
- Amount of dirt cleaned up
- Amount of time taken
- Amount of electricity consumed
- Amount of noise generated, etc.
Agent should strive to “do the right thing”:
- Based on what it can perceive and actions it can do

The “right action”:
- One that will cause the agent to be “most successful”

Performance measure:
- Objective criterion for success of an agent’s behavior

Performance of a vacuum-cleaner agent could be:
- Amount of dirt cleaned up
- Amount of time taken
- Amount of electricity consumed
- Amount of noise generated, etc.
Agent should strive to “do the right thing”:
- Based on what it can perceive and actions it can do

The “right action”:
- One that will cause the agent to be “most successful”

Performance measure:
- Objective criterion for success of an agent’s behavior
- Performance of a vacuum-cleaner agent could be:
  - Amount of dirt cleaned up
  - Amount of time taken
  - Amount of electricity consumed
  - Amount of noise generated, etc.
Agent should strive to “do the right thing”:
Based on what it can perceive and actions it can do

The “right action”:
One that will cause the agent to be “most successful”

Performance measure:
Objective criterion for success of an agent’s behavior

Performance of a vacuum-cleaner agent could be:
Amount of dirt cleaned up
Amount of time taken
Amount of electricity consumed
Amount of noise generated, etc.
Agent should strive to “do the right thing”: Based on what it can perceive and actions it can do

The “right action”: One that will cause the agent to be “most successful”

Performance measure: Objective criterion for success of an agent’s behavior

Performance of a vacuum-cleaner agent could be:
- Amount of dirt cleaned up
- Amount of time taken
- Amount of electricity consumed
- Amount of noise generated, etc.
Rational Agents?

- There is a:
  - Performance measure
  - Percept sequence
  - Agent’s knowledge about the Environment
  - Agent’s action repertoire

- Rational Agent: For each percept sequence
  - Acts so as to maximize expected performance measure
  - Given percept sequence and its built-in knowledge
Rational Agents?

There is a:
- Performance measure
- Percept sequence
- Agent’s knowledge about the Environment
- Agent’s action repertoire

Rational Agent: For each percept sequence
- Acts so as to maximize expected performance measure
- Given percept sequence and its built-in knowledge
Rational Agents?

- Rationality is distinct from omniscience
- All-knowing with infinite knowledge

- Agents can perform actions to modify future percepts
- Use this to obtain useful information
- Information gathering, Exploration

- An Autonomous Agent:
  - Behavior is determined by its own experience
  - With ability to learn and adapt
Rational Agents?

- Rationality is distinct from *omniscience*
- All-knowing with infinite knowledge

- Agents can perform actions to modify future percepts

- Use this to obtain useful information
- Information gathering, Exploration

- An Autonomous Agent:
  - Behavior is determined by its own experience
  - With ability to learn and adapt
Rational Agents?

- Rationality is distinct from omniscience
- All-knowing with infinite knowledge

- Agents can perform actions to modify future percepts

- Use this to obtain useful information
- Information gathering, Exploration

- An Autonomous Agent:
- Behavior is determined by its own experience
- With ability to learn and adapt
Rational Agents?

- Rationality is distinct from *omniscience*
- All-knowing with infinite knowledge

- Agents can perform actions to modify future percepts
- Use this to obtain useful information
  - Information gathering, Exploration

- An **Autonomous** Agent:
  - Behavior is determined by its own experience
  - With ability to learn and adapt
Rational Agents: PEAS

- **PEAS:**
  - Performance measure, Environment, Actuators, Sensors
  - Must first specify the setting for intelligent agent design
  - Example: Task of designing an Automated Taxi Driver
    - **Performance:** Safe, fast, legal, comfort, maximize profits
    - **Environment:** Roads, other traffic, pedestrians, customers
    - **Actuators:** Steering wheel, accelerator, brake, signal, horn
    - **Sensors:** Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard
Rational Agents: PEAS

- PEAS:
  - Performance measure, Environment, Actuators, Sensors
- Must first specify the setting for intelligent agent design
- Example: Task of designing an Automated Taxi Driver
  - Performance: Safe, fast, legal, comfort, maximize profits
  - Environment: Roads, other traffic, pedestrians, customers
  - Actuators: Steering wheel, accelerator, brake, signal, horn
  - Sensors: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard
Rational Agents: PEAS

- PEAS:
  - Performance measure, Environment, Actuators, Sensors
  - Must first specify the setting for intelligent agent design

Example: Task of designing an Automated Taxi Driver

- **Performance**: Safe, fast, legal, comfort, maximize profits
- **Environment**: Roads, other traffic, pedestrians, customers
- **Actuators**: Steering wheel, accelerator, brake, signal, horn
- **Sensors**: Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard
Rational Agents: PEAS

- **PEAS**: Agent: Medical Diagnosis System
  - **Performance**: Healthy patient, minimize costs, lawsuits
  - **Environment**: Patient, hospital, staff
  - **Actuators**: Screen (questions, tests, diagnoses, treatments, referrals)
  - **Sensors**: Keyboard (entry of symptoms, findings, patient’s answers)
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
Rational Agents: PEAS

- **PEAS**: Agent: Interactive English Tutor
  - **Performance measure**: Maximize student’s score on test
  - **Environment**: Set of students
  - **Actuators**: Screen (exercises, suggestions, corrections)
  - **Sensors**: Keyboard
Rational Agents: PEAS

- Four basic types in order of increasing generality
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based (not just that we reach the goal) agents

- We consider (3) and (4) together.
Rational Agents: PEAS

- Four basic types in order of increasing generality
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based (not just that we reach the goal) agents

- We consider (3) and (4) together.
Basic (Simple Reflex) Agent

- Agent
  - Sensors
    - What the world is like now
  - Condition-action rules
  - What action should I do now?
  - Effectors
    - Environment
Basic (Simple Reflex) Agent

```
function Agent (percept) returns action
    static: memory
    memory ← UpdateMemory(memory, percept)
    ; Agent stores percept sequences in memory
    ; Only one input percept per invocation
    action ← ChooseBestAction(memory)
    memory ← UpdateMemory(memory, action)
    ; Performance measure: Evaluated externally
    return action
```

Issues to be considered:
- Model-based Reflex agents
- Keeping track of the world agents
- Goal-based agents
- Utility-based agents...
Autonomous Agents

Types of Environments/Actions

- Basic (Simple Reflex) Agent
- Model-based Reflex Agent
- Goal and Utility-based Agents
- Evaluating Agents

---

**agent**

- state
- how the world changes
- what my actions do

**sensors**

- what the world is like now
- what it will be like if I do A

**effectors**

- what action should I do now?

**environment**

- cond-action rules
Model-based Reflex Agent

Works only if a correct decision can be made on basis of current percept (à la subsumption architecture)

function Agent (percept) returns action
static: rules
state ← InterpretInput(percept)
; Description of world’s state from percept
rule ← RuleMatch(state, rules)
; Returns a rule matching state description
action ← RuleAction(rule)
return action

NEXT: What to do when world is partially observable
function Agent (percept) returns action

static: rules

state ;World state

state ← InterpretInput(percept)
;Description of world state from percept

state ← UpdateState(state, percept)
;Hard! Presupposes knowledge about how:
    ;(1) World changes independently of agent
    ;(2) Agent’s actions effect the world

rule ← RuleMatch(state, rules)
;Returns a rule matching state description

action ← RuleAction(rule)

state ← UpdateState(state, action)
;Hard! Record unsensed parts of World
;Hard! Record effects of agent’s actions

return action
Goal and Utility-based Agents

- Actions depend on current state and goal...
  - Often: **Goal satisfaction requires sequences of actions**
  - **What will happen if I do this?**

- **Credit assignment**
  - Goals are not enough
  - Some goal-achieving sequences are cheaper, faster, etc.
  - **Utility**: states $\rightarrow$ reals
  - **Tradeoffs on goal...**
Goal and Utility-based Agents

- Actions depend on current state and goal...
  - Often: Goal satisfaction requires sequences of actions
  - What will happen if I do this?

- Credit assignment

- Goals are not enough

- Some goal-achieving sequences are cheaper, faster, etc.

- Utility: states → reals

- Tradeoffs on goal...
Autonomous Agents
Types of Environments/Actions

Agents
Basic (Simple Reflex) Agent
Model-based Reflex Agent
Goal and Utility-based Agents
Evaluating Agents

agent

state

how the world changes

what my actions do

utility

goals

rules

sensors

what the world is like now

what it will be like if I do A

how happy will I be in such a state?

what action should I do now?

effectors

environment
function RunEvalEnvironment

(state, UpdateFn, agents, termination, PerformFn)

; Have multiple agents; Returns scores
; State, UpdateFn: Simulate Environment;
; These are unseen by agents!
; Agent's states: Constructed from percepts
; Agents have no access to PerformFn!

repeat

for each agent in agents do
    Percept[agent] ← GetPercept(agent, state)
for each agent in agents do
    Action[agent] ← Program[agent](Percept[agent])
state ← UpdateFn(actions, agents, state)
scores ← PerformFn(scores, agents, state)

until termination

return scores
### Types of Environments

<table>
<thead>
<tr>
<th>Fully Observable/Accessible vs. Partially Observable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent’s sensors: Access environment’s complete state</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deterministic (or not) i.e., Stochastic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next state completely determined by current state &amp; action</td>
</tr>
<tr>
<td>If the environment is deterministic except for the actions of other agents, then the environment is strategic</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Episodic (or not)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The agent’s experience is divided into atomic “episodes”</td>
</tr>
<tr>
<td>Each episode consists of the agent perceiving and then performing a single action</td>
</tr>
<tr>
<td>The choice of action in each episode depends only on the episode itself</td>
</tr>
</tbody>
</table>
Types of Environments

- **Fully Observable/Accessible vs. Partially Observable:**
  - Agent’s sensors: Access environment’s complete state

- **Deterministic (or not) i.e., Stochastic:**
  - Next state completely determined by current state & action
  - If the environment is deterministic except for the actions of other agents, then the environment is strategic

- **Episodic (or not):**
  - The agent’s experience is divided into atomic “episodes”
  - Each episode consists of the agent perceiving and then performing a single action
  - The choice of action in each episode depends only on the episode itself
Types of Environments

- Fully Observable/Accessible vs. Partially Observable:
  - Agent’s sensors: Access environment’s complete state
- Deterministic (or not) i.e., Stochastic
  - Next state completely determined by current state & action
  - If the environment is deterministic except for the actions of other agents, then the environment is strategic
- Episodic (or not)
  - The agent’s experience is divided into atomic “episodes”
  - Each episode consists of the agent perceiving and then performing a single action
  - The choice of action in each episode depends only on the episode itself
Types of Environments

- **Static (or not)**
  - Environment does not change while the agent deliberates
- **Discrete (or not)**
  - Fixed number of well-defined percepts and actions
- **Single agent (vs. Multiagent)**
  - An agent operating by itself in an environment
- **The Real World**
  - Of course: partially observable, stochastic, sequential, dynamic, continuous, multi-agent
- **Chess**: Accessible, Deterministic, ¬Episodic, Static, Discrete
- **Diagnosis**: ¬Access., ¬Determin., ¬Episodic, ¬Static, ¬Discrete
Types of Environments

- **Static (or not)**
  - Environment does not change while the agent deliberates

- **Discrete (or not)**
  - Fixed number of well-defined percepts and actions

- **Single agent (vs. Multiagent)**
  - An agent operating by itself in an environment

- **The Real World**
  - Of course: partially observable, stochastic, sequential, dynamic, continuous, multi-agent

- **Chess**: Accessible, Deterministic, ¬Episodic, Static, Discrete
  - **Diagnosis**: ¬Access., ¬Determin., ¬Episodic, ¬Static, ¬Discrete
Types of Environments

- **Static (or not)**
  - Environment does not change while the agent deliberates

- **Discrete (or not)**
  - Fixed number of well-defined percepts and actions

- **Single agent (vs. Multiagent)**
  - An agent operating by itself in an environment

- **The Real World**
  - Of course: partially observable, stochastic, sequential, dynamic, continuous, multi-agent

- **Chess**: Accessible, Deterministic, ¬Episodic, Static, Discrete
- **Diagnosis**: ¬Access., ¬Determin., ¬Episodic, ¬Static, ¬Discrete
Types of Environments

- **Static (or not)**
  - Environment does not change while the agent deliberates
- **Discrete (or not)**
  - Fixed number of well-defined percepts and actions
- **Single agent (vs. Multiagent)**
  - An agent operating by itself in an environment

**The Real World**
- Of course: partially observable, stochastic, sequential, dynamic, continuous, multi-agent

- **Chess**: Accessible, Deterministic, ¬Episodic, Static, Discrete
- **Diagnosis**: ¬Access., ¬Determin., ¬Episodic, ¬Static, ¬Discrete
Types of Environments

- **Static (or not)**
  - Environment does not change while the agent deliberates
- **Discrete (or not)**
  - Fixed number of well-defined percepts and actions
- **Single agent (vs. Multiagent)**
  - An agent operating by itself in an environment
- **The Real World**
  - Of course: partially observable, stochastic, sequential, dynamic, continuous, multi-agent
- **Chess**: Accessible, Deterministic, ¬Episodic, Static, Discrete
- **Diagnosis**: ¬Access., ¬Determin., ¬Episodic, ¬Static, ¬Discrete
Source of Actions Selected by the Agent

**Performance Element**
- Agent program to select actions

**Learning Element**
- Improves PE and makes agent’s behavior robust
- In initially unknown environments

**Problem Generator**
- Suggests actions
- May lead to new, informative experiences

**Exploitation vs Exploration**
Source of Actions Selected by the Agent

- Agent
- Problem Generator
- Learning Element
- Performance Element
- Sensors
- Feedback
- Knowledge
- Changes
- Actuators
- Environment
- Performance Standard