Manchester Metropolitan University

Artificial Intelligence Principles

6G7V0011 - 1CWK100

Dr. Peng Wang

Email: p.wang@mmu.ac.uk

Department of Computing and Mathematics

Wednesday, Oct. 2nd, 2024

Outline

Agents

Outline

Agents

Agent

Agent

"An **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and acting upon that environment through **actuators**."

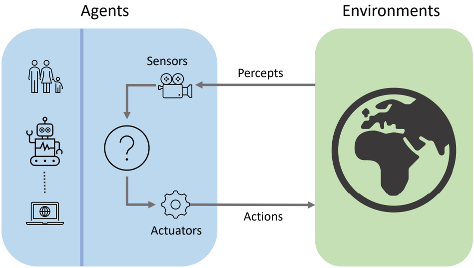

Figure 1: Agents interact with environments.

Agent

Rational Agent

"An **rational agent** is an agent that acts so as to achieve the best outcome or, when there is uncertainty, the best expected outcome."

Examples:

- A **human agent** has eyes, ears, and other organs for sensors and hands, legs, vocal tract, and so on for actuators.
- A **robotic agent** might have cameras and infrared range finders for sensors and various motors for actuators.
- A **software agent** receive file contents, network packets, and human input (keyboard/mouse/touchscreen/voice) as sensory inputs and acts on the environment by writing files, sending network packets, and displaying information or generating sounds.

What's the question mark represents?

Agents and Environments

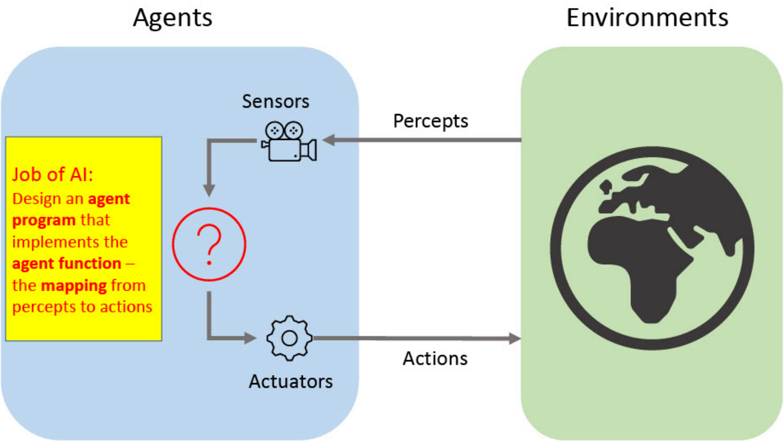

Figure 2: The role AI plays in agent-environment pairs

Note: The question mark is a 'mapping (function)' from percepts to actions.

Agents and Environments - Concepts

Percept: Agents perceptual inputs at any given instant

Percepts: Complete history of everything that the agent has ever perceived.

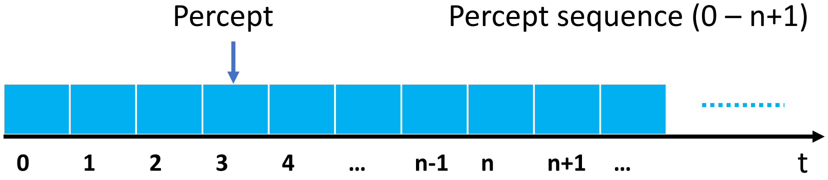

Figure 3: Percept vs. percept sequence

Note: A percept is a percept sequence. Not the other way round.

Agents and Environments - Concepts

Agent function

- A **mathematical** description of an agent's behavior
- A function mapping any given percept sequence to an action

Agent program

- A **practical** description of an agent's behavior
- It is (one) an implementation(s) of the agent function

Note:

- Agent programs take the current percept as input from the sensors and return an action to the actuators
- **The difference** between the agent program and the agent function: the former takes the current percept as input; while the latter may depend on the entire percept history.

Agents and Environments - Examples

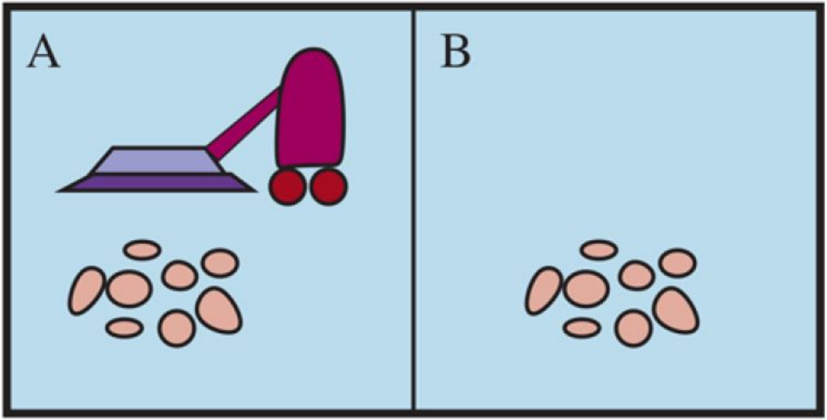

Figure 4: A vacuum-cleaner world with just two locations. Each location can be clean or dirty, and the agent can move left or right and can clean the square that it occupies. Different versions of the vacuum world allow for different rules about what the agent can perceive, whether its actions always succeed, and so on.

Agents and Environments - Examples

Percepts

- Which location it is in? A or B?
- Is the current location clean or dirty?

Actions

- Move left
- Move right
- Suck the dirt
- No operation

What do we need now? A mapping between the two!

Agents and Environments - Examples

Percept sequence	Action
$[A, Clean]$	*Right*
$[A, Dirty]$	*Suck*
$[B, Clean]$	*Left*
$[B, Dirty]$	*Suck*
$[A, Clean], [A, Clean]$	*Right*
$[A, Clean], [A, Dirty]$	*Suck*
$\vdots$	$\vdots$
$[A, Clean], [A, Clean], [A, Clean]$	*Right*
$[A, Clean], [A, Clean], [A, Dirty]$	*Suck*
$\vdots$	$\vdots$

Figure 5: Partial tabulation of a simple agent function for the vacuum-cleaner world. The agent cleans the current square if it is dirty, otherwise it moves to the other square. Note that the table is of unbounded size unless there is a restriction on the length of possible percept sequences.

Agents and Environments - Examples

Agent Function

If the current square is dirty, then suck; otherwise, move to the other square.

Algorithm 1: Reflex Vacuum Agent (**Agent Program**)

Input: *location*, *status*
Output: Action

1: **if** *status* == *Dirty* **then**
2:     **return** *suck*
3: **else if** *location* == *A* **then**
4:     **return** *move right*
5: **else if** *location* == *B* **then**
6:     **return** *move left*
7: **end if**

What will happen after all the dirt is cleaned up? *irrational?*

Rationality

Rational Agent

- One that does the right thing
- 'The right thing' means actions/performance that make an agent more successful or desirable
- **Performance measure** (e.g. cost function) to 'assess' success or desirability

What is rational at any given time depends on four things:

- The performance measure defining the criterion of success
- The agent's prior knowledge of the environment
- The actions that the agent can perform
- The agent's percept sequence up to now

Rationality

Manchester Metropolitan University

Making the vacuum agent rational

- The performance measure awards one point for each clean square at each time step, over a ‘lifetime’ of 1000 time steps.
- The ‘geography’ of the environment is known a priori but the dirt distribution and the initial location of the agent are not. Clean squares stay clean and sucking cleans the current square. The *move right* and *move left* actions move the agent one square except when this would take the agent outside the environment, in which case the agent remains where it is.
- The only available actions are *move right*, *move left*, *suck*, and **no operation**.
- The agent correctly perceives its location and whether that location contains dirt.

Such an agent is rational!

Summary

- What's agent and rational agent
- Percept vs. percepts
- Agent function vs. agent program
- Rules for designing rational agents
  - Performance measure

Task Environment

We know rationality, and let's see how to build rational agents to solve real problems.

Formulate problems as task environments:

- **Task environments**: essentially the 'problems', and rational agents are the 'solutions'
  1. **P**erformance
  2. **E**nvironment
  3. **A**ctuators
  4. **S**ensors
- The nature of the **task environment** directly affects the appropriate design for the agent program
- The first step in designing an agent is always be to specify the **task environment** as fully as possible

Task Environment

Manchester Metropolitan University

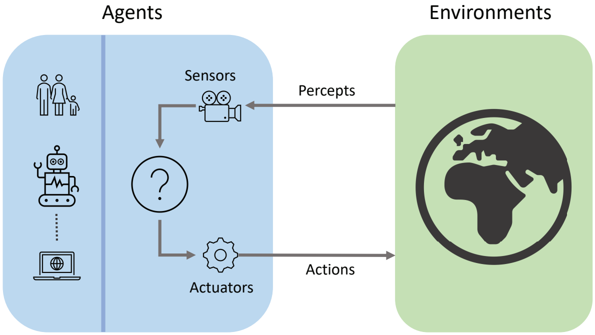

Figure 6: Agents interact with environments.

Task Environment - Example

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits, minimize impact on other road users	Roads, other traffic, police, pedestrians, customers, weather	Steering, accelerator, brake, signal, horn, display, speech	Cameras, radar, speedometer, GPS, engine sensors, accelerometer, microphones, touchscreen

Figure 7: PEAS description of the task environment for an **automated** taxi driver

Task Environment - Examples

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments	Touchscreen/voice entry of symptoms and findings
Satellite image analysis system	Correct categorization of objects, terrain	Orbiting satellite, downlink, weather	Display of scene categorization	High-resolution digital camera
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, tactile and joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, raw materials, operators	Valves, pumps, heaters, stirrers, displays	Temperature, pressure, flow, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, feedback, speech	Keyboard entry, voice

Figure 8: Examples of agent types and their PEAS descriptions

Agent Programs

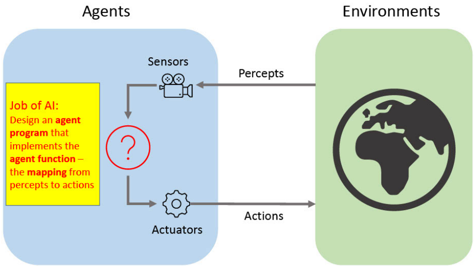

Figure 9: The role AI plays in agent-environment pairs

agent = architecture + program

- **Agent program**: Implementations of the agent function the mapping from percepts to actions
- **Agent architecture**: Computing devices with physical sensors and actuators that agent program runs on

Agent Programs - Challenges

Algorithm 2: Table Driven Agent

Input: *percepts*: an initially empty sequence; *table*: an initially fully specified table of actions, indexed by percept sequences;

Output: Action

1: append *percept* to the end of *percepts*

2: $action \leftarrow lookup(percepts, table)$

3: **return** *action*

- Table-driven approach to agent construction is doomed to failure.
- For P possible pecepts and the total number of perception cycles T, the lookup table will have $\sum_{t=1}^{T} |P|^t$ entries
- More complex scenarios needs more space for storage and time for processing

Key challenge to AI: Find out how to write programs that, to the extent possible, produce rational behavior from a smallish program rather than from a vast table.

Agent Programs - types

Four basic kinds of agent programs that embody the principles underlying almost all intelligent systems

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

Agent Programs - Reflex Agent

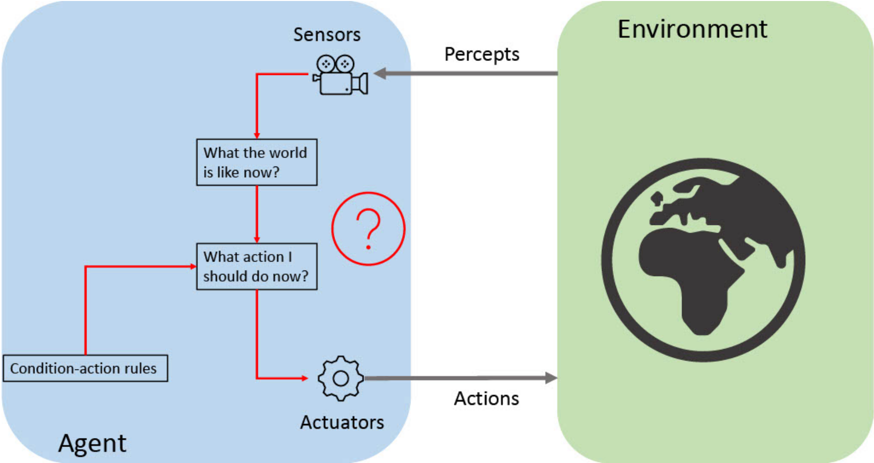

Figure 10: Simple reflex agent

Agent Programs - Reflex Agent

- Condition-action rule based, e.g. **if** *rear-lights-of-the-front-car-turn-red* **then** *decelerate*
- Easy to implement
- Depends on current percepts only, ignoring historical percepts
- Works only in fully observable enviroment.
  - Vacuum cleaner's sensor to detect location is deprived
  - It will it could move left while it is in A, and move right while it is in B, making it irrational

Agent Programs - Model-based Agent

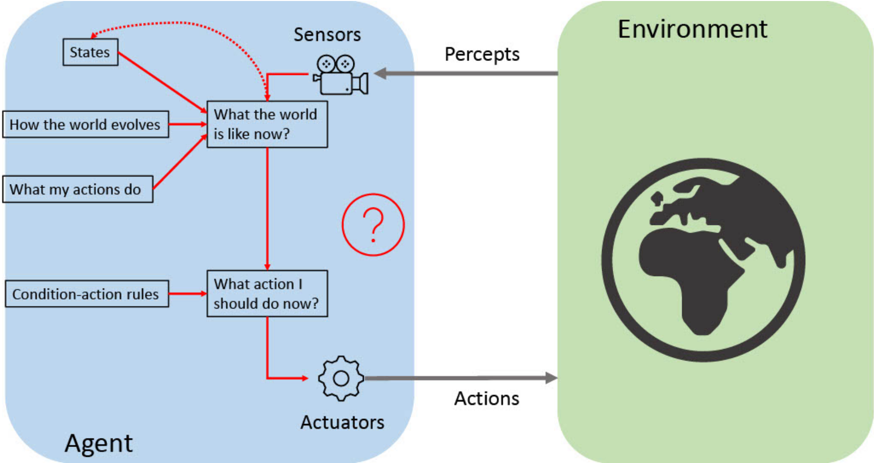

Figure 11: Model-based reflex agent

Keep track of the part of the world it can't see now to **handle partial observability!**

Agent Programs - Model-based Agent

- Transition model:

$$X_{t+1} = AX_t + U \tag{1}$$

- Sensor model:

$$Y_t = BX_t \tag{2}$$

- Rule for action depends on both state and percept
- Different from reflex, which only depends on percept

Agent Programs - Goal-based Agent

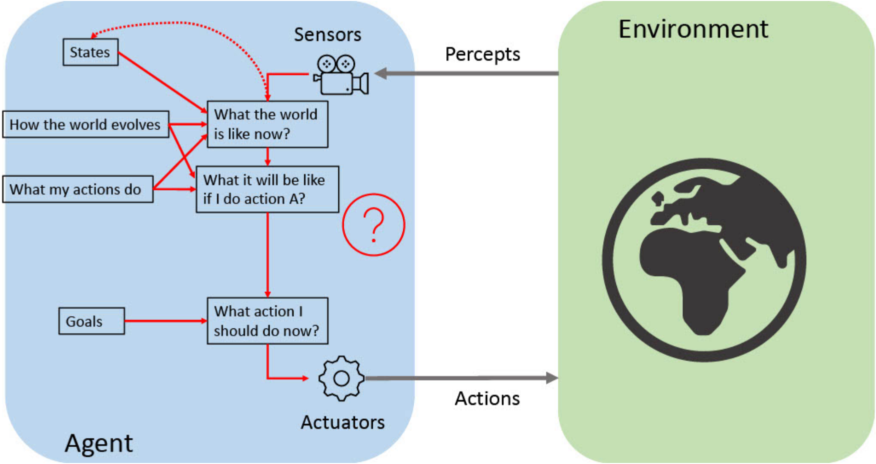

Figure 12: Goal-based agent

Agent Programs - Goal-based Agent

- Agent continues to receive percepts and maintain state
- Agent also has a goal
  - Makes decisions based on achieving goal
- **Search** and planning are the subfields of AI devoted to finding action sequences that achieve the agent's goals
- Example
  - Pathfinder goal: reach a boulder
  - If pathfinder trips or gets stuck, can make decisions to reach goal

Agent Programs - Utility-based Agent

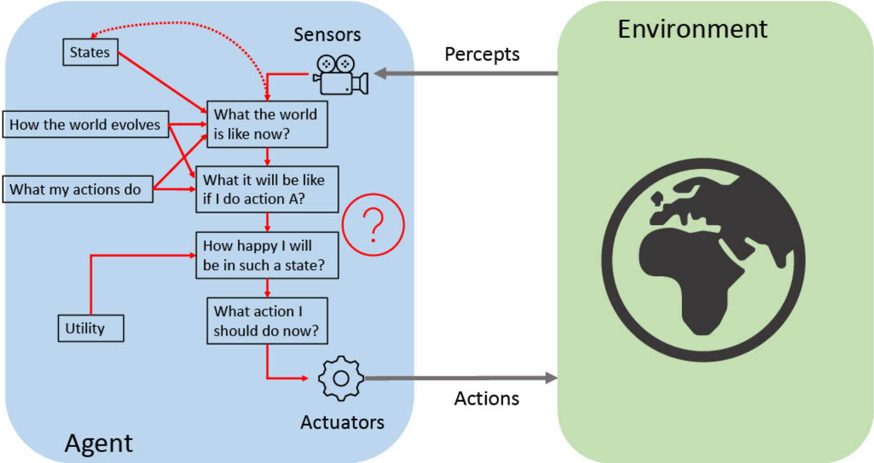

Figure 13: Utility-based agent

Agent Programs - Utility-based Agent

- Goals are not enough need to know value of goal
  - Is this a minor accomplishment, or a major one?
  - Affects decision making will take greater risks for more major goals
- Utility: numerical measurement of importance of a goal
- A utility-based agent will attempt to make the appropriate tradeoff
  1. When there are conflicting goals, and only some of the goals but not all can be achieved, utility then describes the appropriate trade-off
  2. When there are several goals and none of them are achieved certainly, then utility provides a way for the decision-making

Summary

We have learned:

- Difference types of agents
- Subsequent content will be based on utility-based agents.