INTELLIGENT AGENTS

CS4373: Artificial Intelligence Laboratory
A.Y. 2024-2025
Department of Computer Science and Engineering
National Institute of Technology Rourkela

July 29, 2024

Contents

Agent Programming

Agent
An agent can perceive its environment through sensors and act upon that environment through actuators.

Environment
The outside circumstances in which an agent functions to accomplish a particular task.

Agent Function
The agent function is a mathematical function that maps a sequence of perceptions into action.

Agent Program
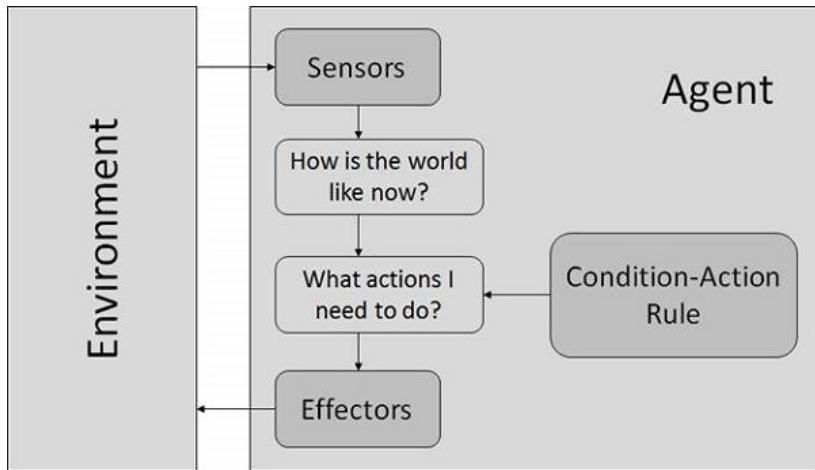An agent program is an implementation of an agent function.

Simple Reflex Agents

Figure 1: Simple Reflex Agents.[1]

[1]Russell, S. J., Norvig, P. (2021). Artificial Intelligence: A Modern Approach. United Kingdom: Pearson.
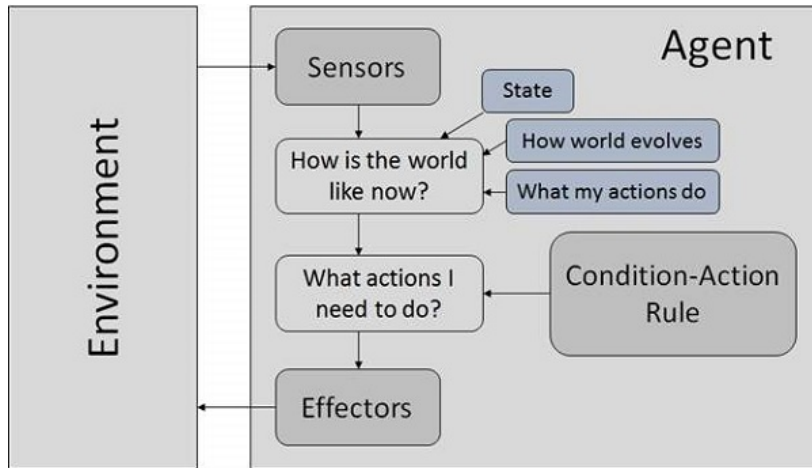
Figure 2: Model-based reflex agents.[1]

[1]Russell, S. J., Norvig, P. (2021). Artificial Intelligence: A Modern Approach. United Kingdom: Pearson.
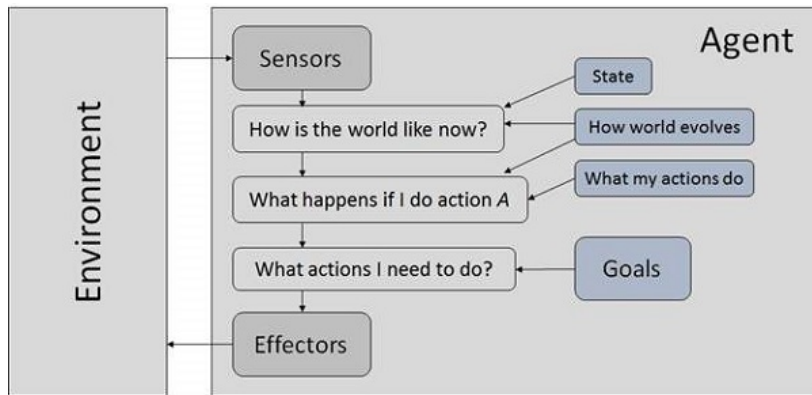
Figure 3: Goal-based agents.[1]

[1]Russell, S. J., Norvig, P. (2021). Artificial Intelligence: A Modern Approach. United Kingdom: Pearson.
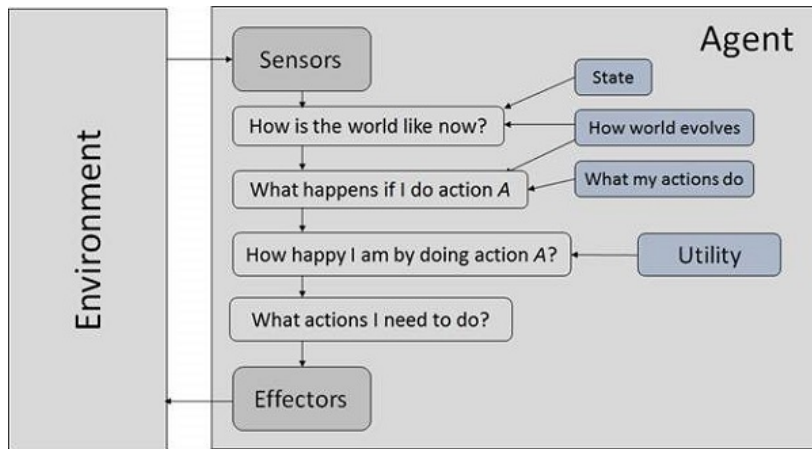
Figure 4: Utility-based Agents.[1]

[1]Russell, S. J., Norvig, P. (2021). Artificial Intelligence: A Modern Approach. United Kingdom: Pearson.

Assignment 1 (a)

Problem Statement

Simulate a simple two-room vacuum cleaner world where a vacuum cleaner agent is placed in a house with two rooms (Room A and Room B), and its goal is to clean the dirty rooms.

Environment Description

- The house has two rooms: Room A and Room B.
- Each room can be in one of two states: "Clean" or "Dirty".
- The agent starts in one of the rooms at random.
- The agent can move between rooms and clean the rooms.

Agent Description

- The vacuum cleaner agent uses a Simple Reflex approach to make decisions.
- The agent has the following sensors:
    - Current room location (Room A or Room B).
    - Dirt sensor (detects if the current room is dirty).

Agent's Decision-Making

- The agent's decision-making is based on a set of simple rules:
  - If the current room is dirty, the agent cleans the room.
  - If the current room is clean, the agent moves to the other room.

Tasks

1. Implement the **'SimpleVacuumEnvironment'** class that represents the vacuum cleaner world and includes methods to:
   - Check if a room is dirty.
   - Clean a room.
   - Move the agent to another room.
   - Display the current state of the environment.

2. Implement the **'SimpleReflexVacuumAgent'** class that represents the vacuum cleaner agent using a Simple Reflex approach. The agent should have methods to:
   - Perceive the current room and dirt status.
   - Decide actions based on the rules described above.
   - Perform the decided action.

3. Create an instance for each class and simulate the agent's actions for a few steps, displaying the environment's state after each action.

Assignment 1 (b)

Problem Statement

Simulate a simple two-room vacuum cleaner world where a vacuum cleaner agent is placed in a house with two rooms (Room A and Room B), and its goal is to clean the dirty rooms.

Environment Description

- The house has two rooms: Room A and Room B.
- Each room can be in one of two states: "Clean" or "Dirty".
- The agent starts in one of the rooms at random.
- The agent can move between rooms and clean the rooms.

Agent Description

- The vacuum cleaner agent uses a Model-Based Reflex approach to make decisions.
- The agent has the following sensors:
  - Current room location (Room A or Room B).
  - Dirt sensor (detects if the current room is dirty).
- The agent maintains a model of the environment, which includes the dirt status of both rooms.

Agent's Decision-Making

- The agent's decision-making is based on following rules:
  - If the current room is dirty based on the actual state and the model, the agent cleans the room.
  - If the current room is clean based on the actual state but dirty in the model, the agent cleans the room based on the model.
  - Otherwise, the agent moves to the other room.

Tasks

1. Implement the **'SimpleVacuumEnvironment'** class that represents the vacuum cleaner world and includes methods to:
   - Check if a room is dirty.
   - Clean a room.
   - Move the agent to another room.
   - Display the current state of the environment.

2. Implement the **'ModelBasedReflexVacuumAgent'** class that represents the vacuum cleaner agent using a Model-Based Reflex approach. The agent should have methods to:
   - Perceive the current room and dirt status.
   - Maintain and update a model of the environment.
   - Decide actions based on the rules described above.
   - Perform the decided action.

3. Create an instance for each class and simulate the agent's actions for a few steps, displaying the environment's state after each action.

Assignment 1 (c)

Problem Statement

Simulate a simple two-room vacuum cleaner world where a vacuum cleaner agent is placed in a house with two rooms (Room A and Room B), and its goal is to clean the dirty rooms.

Environment Description

- The house has two rooms: Room A and Room B.
- Each room can be in one of two states: "Clean" or "Dirty".
- The agent starts in one of the rooms at random.
- The agent can move between rooms and clean the rooms.

Agent Description

- The vacuum cleaner agent uses a Goal-Based Agent approach to make decisions.
- The agent has the following sensors:
  - Current room location (Room A or Room B).
  - Dirt sensor (detects if the current room is dirty).

Agent's Decision-Making

- The agent's decision-making is based on explicit goals and priorities:
    - The agent maintains a list of goals, each represented as a tuple with an action and a priority.
    - The agent prioritizes and selects the highest-priority goal and performs the associated action.

Tasks

1. Implement the **'SimpleVacuumEnvironment'** class that represents the vacuum cleaner world and includes methods to:
    - Check if a room is dirty.
    - Clean a room.
    - Move the agent to another room.
    - Display the current state of the environment.

2. Implement the **'GoalBasedVacuumAgent'** class that represents the vacuum cleaner agent using a Goal-Based Agent approach. The agent should have methods to:
    - Set goals with actions and priorities.
    - Prioritize goals based on their priorities.
    - Perceive the current room and dirt status.
    - Decide actions based on the highest-priority goal.
    - Perform the decided action.

3. Create an instance for each class and simulate the agent's actions for a few steps, displaying the environment's state after each action.

Problem Statement

Simulate a simple two-room vacuum cleaner world where a vacuum cleaner agent is placed in a house with two rooms (Room A and Room B), and its goal is to clean the dirty rooms.

Environment Description

- The house has two rooms: Room A and Room B.
- Each room can be in one of two states: "Clean" or "Dirty".
- The agent starts in one of the rooms at random.
- The agent can move between rooms and clean the rooms.

Agent Description

- The vacuum cleaner agent uses a Utility-Based Agent approach to make decisions.
- The agent has the following sensors:
  - Current room location (Room A or Room B).
  - Dirt sensor (detects if the current room is dirty).
- The agent maintains utility values for each room, representing the desirability of its state.

Agent's Decision-Making

- The agent's decision-making is based on utility values assigned to different states:
  - The agent calculates utility values for each room based on its cleanliness.
  - The agent prioritizes actions based on the calculated utilities and selects the action with the highest utility.

Tasks

1. Implement the **'SimpleVacuumEnvironment'** class that represents the vacuum cleaner world and includes methods to:
   - Check if a room is dirty.
   - Clean a room.
   - Move the agent to another room.
   - Display the current state of the environment.

2. Implement the **'UtilityBasedVacuumAgent'** class that represents the vacuum cleaner agent using a Utility-Based Agent approach. The agent should have methods to:
   - Calculate utility values for each room based on cleanliness.
   - Decide actions based on utility values.
   - Perform the decided action.

3. Create an instance for each class and simulate the agent's actions for a few steps, displaying the environment's state after each action.

Thank You.