

Chapter 2: Intelligent Agents





Outline

- Last class, introduced AI and rational agent
- Today's class, focus on intelligent agents
 - Agent and environments
 - Nature of environments influences agent design
 - Basic “skeleton” agent designs

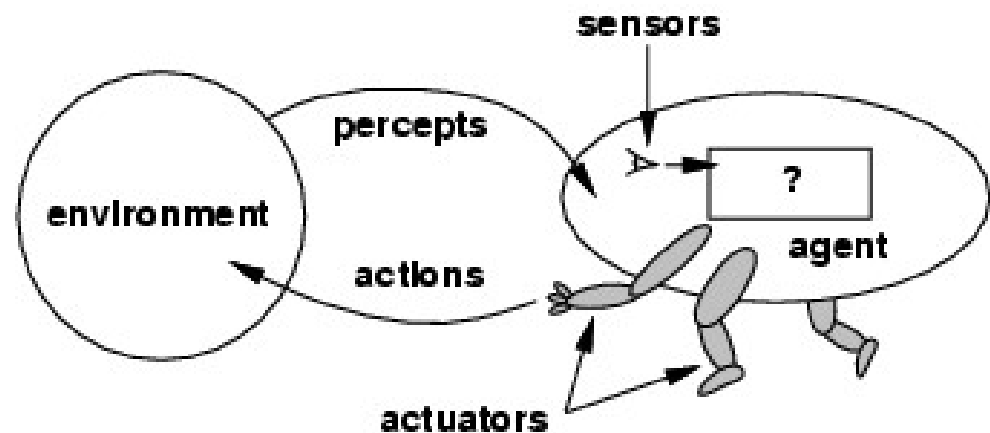


Outline

- Last class, introduced AI and rational agent
- Today's class, focus on intelligent agents
 - **Agent and environments**
 - Nature of environments influences agent design
 - Basic “skeleton” agent designs

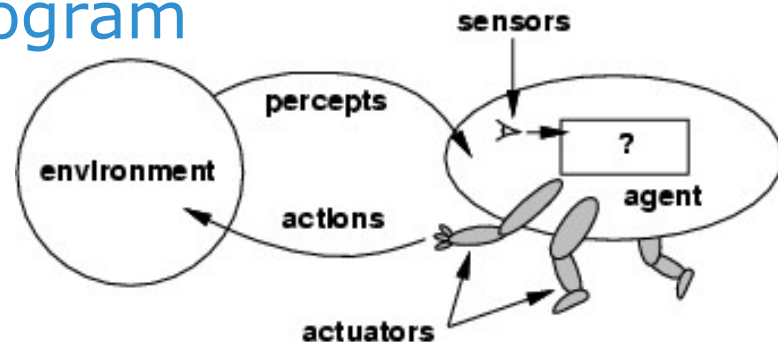
Agents

- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting** upon that environment through **actuators**
- Examples:
 - Human agent
 - Robotic agent
 - Software agent



Terminologies

- **Percept**: the agent's perceptual inputs
- **Percept sequence**: the complete history of everything the agent has perceived
- **Agent function** maps any given percept sequence to an action $[f: p^* \rightarrow A]$
- The **agent program** runs on the physical architecture to produce f
- **Agent = architecture + program**

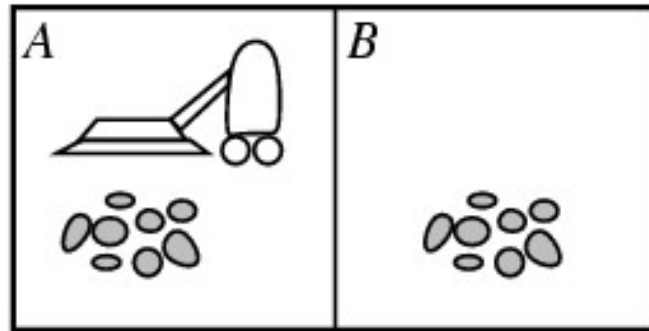




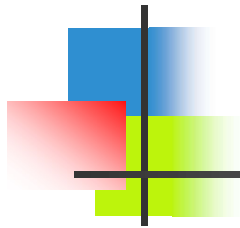
Questions

- Can there be more than one agent program that implements a given agent function?
- Given a fixed machine architecture, does each agent program implement exactly one agent function?


Vacuum-Cleaner World

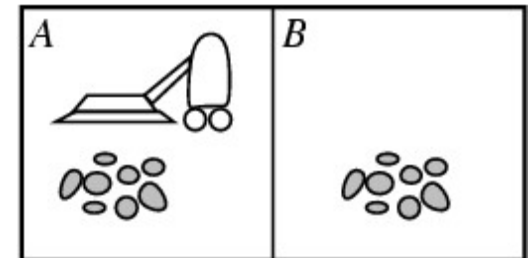


- **Percepts:** location and contents, e.g., [A, dirty]
- **Actions:** Left, Right, Suck, NoOp



A Simple Agent Function

| Percept sequence | Action |
|------------------------------------|---|
| [A, Clean] |  |
| [A, Dirty] | |
| [B, Clean] | |
| [B, Dirty] | |
| [A, Clean], [A, Clean] | |
| [A, Clean], [A, Dirty] | |
| ... | |
| [A, Clean], [A, Clean], [A, Clean] | |
| [A, Clean], [A, Clean], [A, Dirty] | |
| ... | |





Rationality

- An agent should "**do the right thing**", based on what it can perceive and the actions it can perform. The right action is the one that will cause the agent to be most successful
- **Performance measure**: An objective criterion for success of an agent's behavior
- Back to the vacuum-cleaner example
 - Amount of dirt cleaned within certain time
 - +1 credit for each clean square per unit time
- General rule: measure what one wants rather than how one thinks the agent should behave



Rational Agent

- Definition:

- 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.



Rational Agent

- Definition:
 - 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.



Vacuum-Cleaner Example

- A simple agent that cleans a square if it is dirty and moves to the other square if not
- Is it rational?
- **Assumption:**
 - performance measure: 1 point for each clean square at each time step
 - environment is known a priori
 - actions = {left, right, suck, no-op}
 - agent is able to perceive the location and dirt in that location
- Given different assumption, it might not be rational anymore



Omniscience, Learning and Autonomy

- Distinction between rationality and **omniscience**
 - expected performance vs. actual performance
- Agents can perform actions in order to modify future percepts so as to obtain useful information (**information gathering, exploration**)
- An agent can also **learn** from what it perceives
- An agent is **autonomous** if its behavior is determined by its own experience (with ability to learn and adapt)



Questions

- Given the assumption on slide 12. Describe a rational agent function for the modified performance measure that deducts one point for each movement. Does the agent program require internal state?
- Discuss possible agent designs for the cases in which clean squares can become dirty and the geography of the environment is unknown.



Outline

- Last class, introduced AI and rational agent
- Today's class, focus on intelligent agents
 - Agent and environments
 - Nature of environments influences agent design
 - Basic “skeleton” agent designs



PEAS

- Specifying the task environment is always the first step in designing agent
- **PEAS:**
 - **P**erformance, **E**nvironment, **A**ctuators, **S**ensors



Taxi Driver Example

| Performance Measure | Environment | Actuators | Sensors |
|--|---|---|--|
| safe, fast, legal, comfortable trip, maximize profits | roads, other traffic, pedestrians, customers | steering, accelerator, brake, signal, horn, display | camera, sonar, speedometer, GPS, odometer, engine sensors, keyboard, accelerator |

DARPA urban challenge 07:

<http://www.youtube.com/watch?v=SQFEmR50HAK>

Medical Diagnosis System

| Performance Measure | Environment | Actuators | Sensors |
|---|--------------------------|--|---|
| healthy patient, minimize costs, lawsuits | patient, hospital, staff | display questions, tests, diagnosis, treatments, referrals | keyboard entry of symptoms, findings, patient's answers |



Mushroom-Picking Robot

| Performance Measure | Environment | Actuators | Sensors |
|--|------------------------------------|----------------------|-----------------------------|
| Percentage of good mushrooms in correct bins | Conveyor belt with mushrooms, bins | Jointed arm and hand | camera, joint angle sensors |





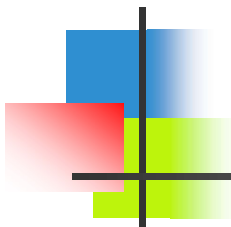
Properties of Task Environments

- **Fully observable** (vs. **partially observable**):
 - An agent's sensors give it access to the complete state of the environment at each point in time
- **Deterministic** (vs. **stochastic**):
 - next state of the env. determined by current state and the agent's action
 - If the environment is deterministic except for the actions of other agents, then the environment is **strategic**
- **Episodic** (vs. **sequential**):
 - Agent's experience is divided into atomic "episodes"
 - Choice of action in each episode depends only on the episode itself



Properties of Task Environments

- **Static** (vs. **dynamic**):
 - The environment is unchanged while an agent is deliberating
 - **Semidynamic** if the environment itself doesn't change with time but the agent's performance score does
- **Discrete** (vs. **continuous**):
 - A limited number of distinct, clearly defined percepts and actions
- **Single agent** (vs. **multiagent**):
 - An agent operating by itself in an environment
 - Competitive vs. cooperative



Examples

| Task Environment | Oberservable | Deterministic | Episodic | Static | Discrete | Agents |
|---------------------------|--------------|---------------|------------|---------|----------|--------|
| <i>Crossword puzzle</i> | fully | deterministic | sequential | static | discrete | single |
| <i>Chess with a clock</i> | fully | strategic | sequential | semi | discrete | multi |
| <i>Taxi driver</i> | partially | stochastic | sequential | dynamic | conti. | multi |
| <i>mushroom-picking</i> | partially | stochastic | episodic | dynamic | conti. | single |

- The environment type largely determines the agent design
- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent



Exercises

- Develop PEAS description for the following task environment:
 - Robot soccer player
 - Shopping for used AI books on the Internet
- Analyze the properties of the above environments



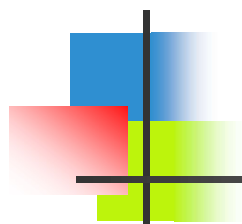
True/False Questions

- An agent that senses only partial information about the state cannot be perfectly rational.
- Suppose an agent selects its action uniformly at random from the set of possible actions. There exists a deterministic task environment in which this agent is rational.
- It is possible for a given agent to be perfectly rational in two distinct task environments.
- A perfectly rational poker-playing agent never loses.



Outline

- Last class, introduced AI and rational agent
- Today's class, focus on intelligent agents
 - Agent and environments
 - Nature of environments influences agent design
 - **Basic "skeleton" agent designs**



Agent = Architecture + Program

- The job of AI is to design the **agent program** that implements the **agent function** mapping percepts to actions
- Aim: find a way to implement the **rational agent function** concisely
- Same skeleton for agent program: it takes the **current percept** as input from the sensors and returns an action to the actuators



Agent Program vs. Agent Function

- Agent program takes the current percept as input
 - Nothing is available from the environment
- Agent function takes the entire percept history
 - To do this, remember all the percepts



Table-Driven Agent

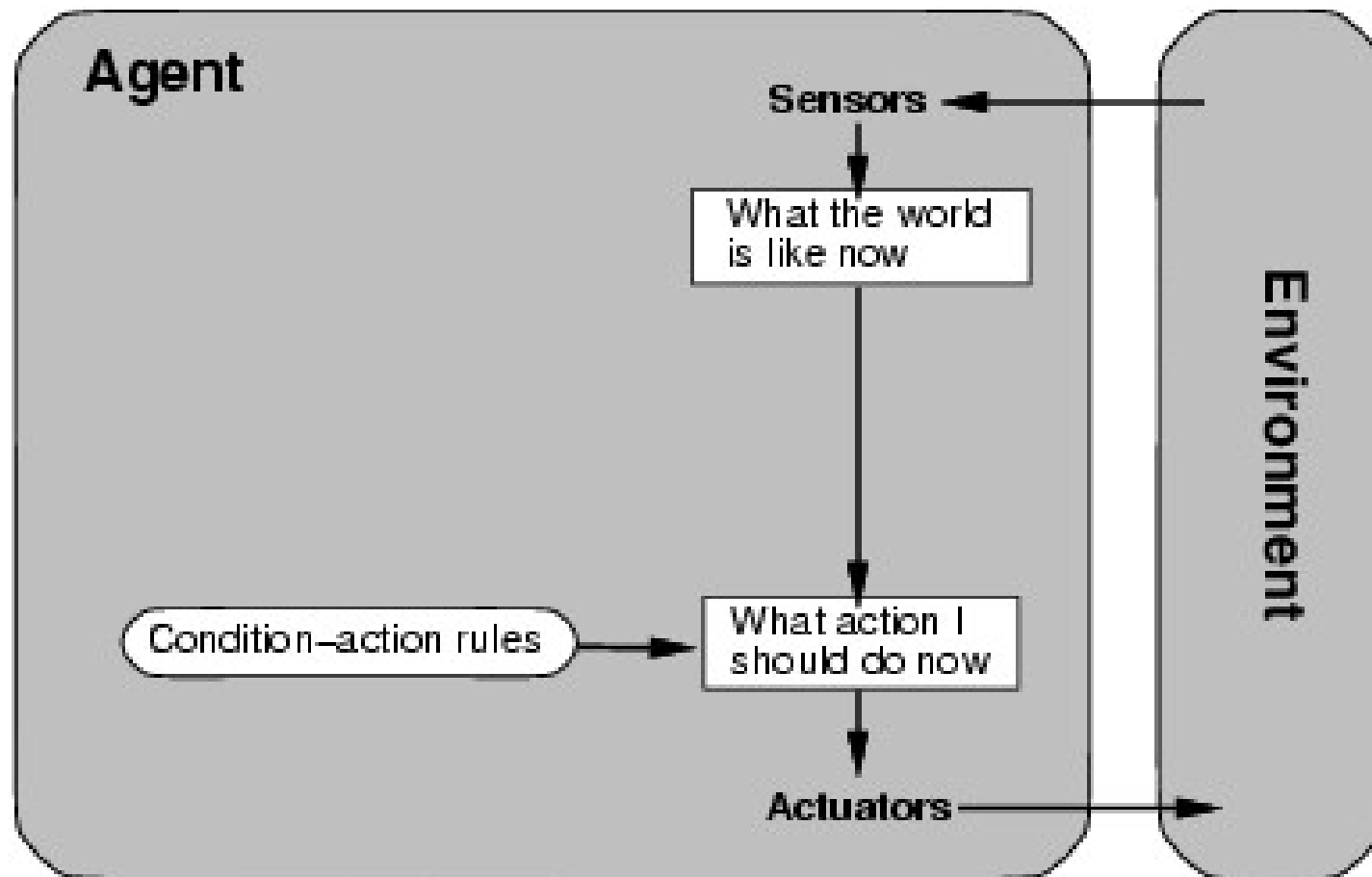
- Designer needs to construct a table that contains the appropriate action for every possible percept sequence
- Drawbacks?
 - huge table
 - take a long time to construct such a table
 - no autonomy
 - Even with learning, need a long time to learn the table entries

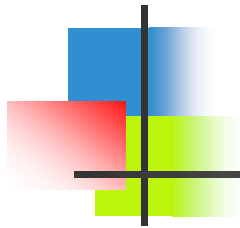


Five Basic Agent Types

- Arranged in order of increasing generality:
 - Simple reflex agents
 - Model-based reflex agents
 - Goal-based agents
 - Utility-based agents; and
 - Learning agents

Simple Reflex Agent





Pseudo-Code

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

static: *rules*, a set of condition–action rules

state \leftarrow INTERPRET-INPUT(*percept*)

rule \leftarrow RULE-MATCH(*state*, *rules*)

action \leftarrow RULE-ACTION[*rule*]

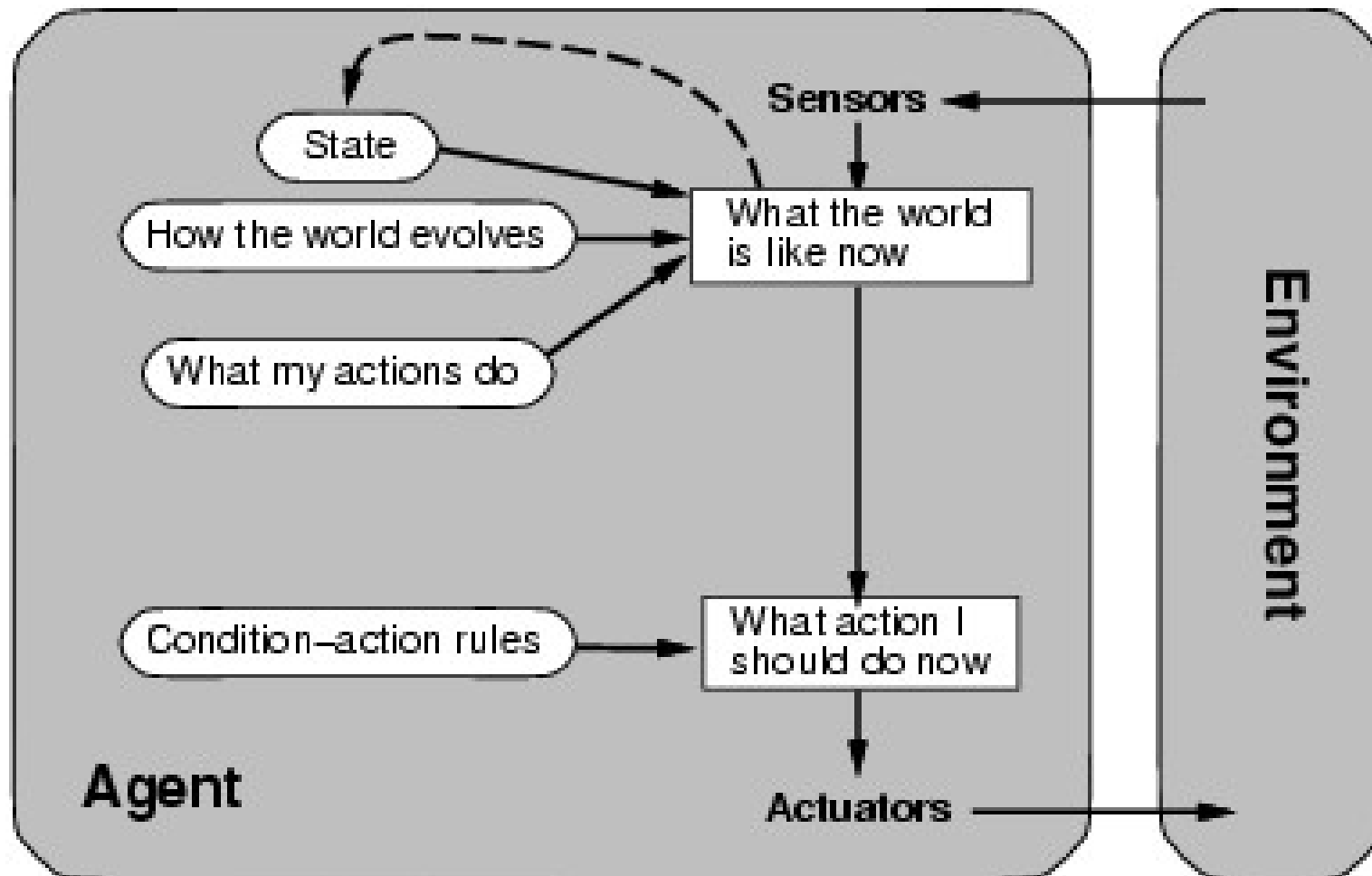
return *action*

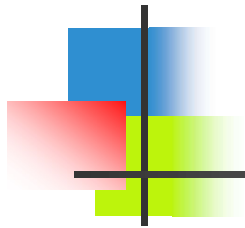
Example: write a simple reflex agent for the vacuum cleaner example



- **Infinite loops** are often unavoidable for simple reflex agent operating in partially observable environments
 - No location sensor
- **Randomization** will help
- A randomized simple reflex agent might outperform a deterministic simple reflex agent
- Better way: keep track of the part of the world it can't see now
 - Maintain internal states

Model-Based Reflex Agent



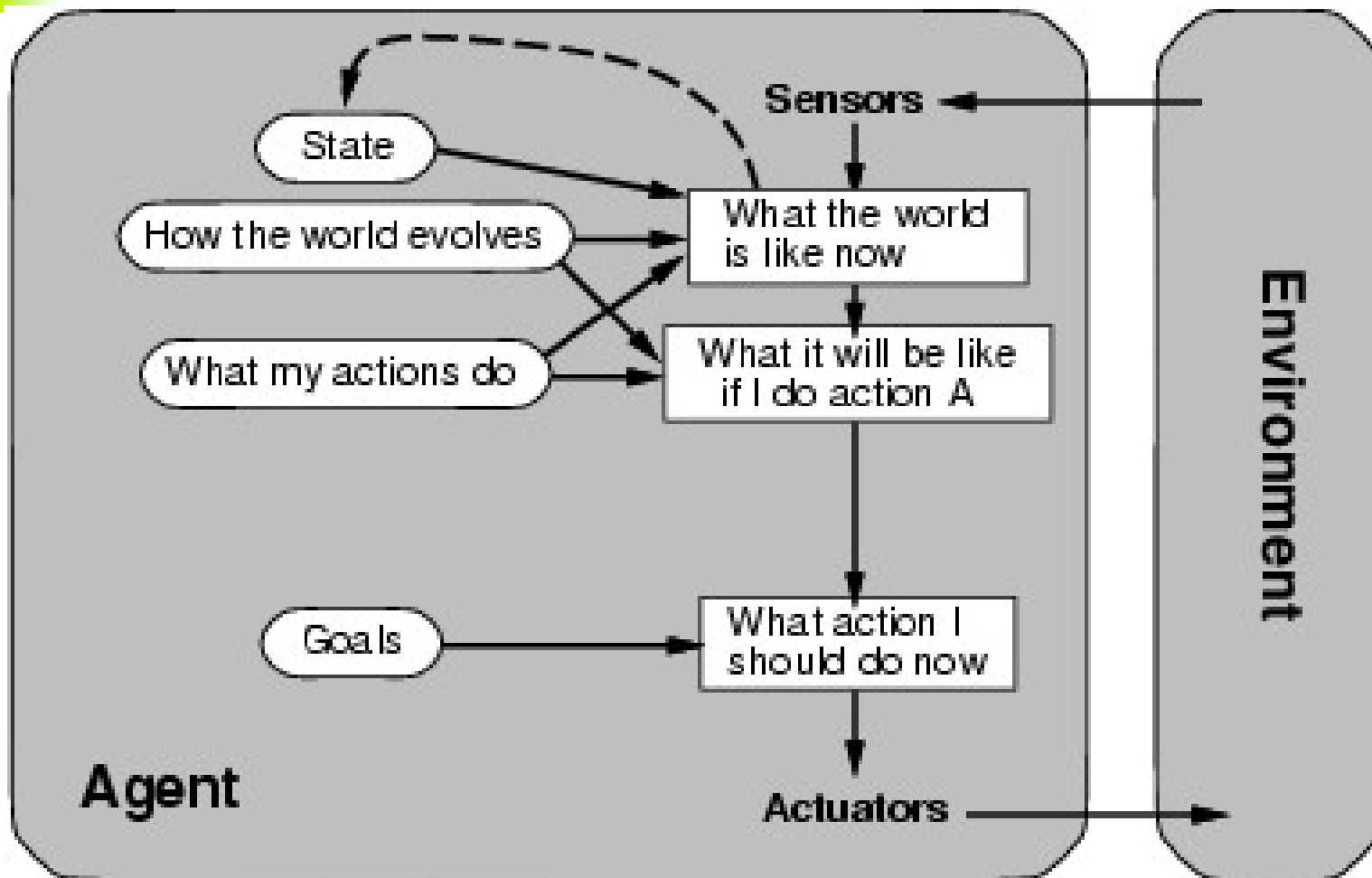


Pseudo-Code

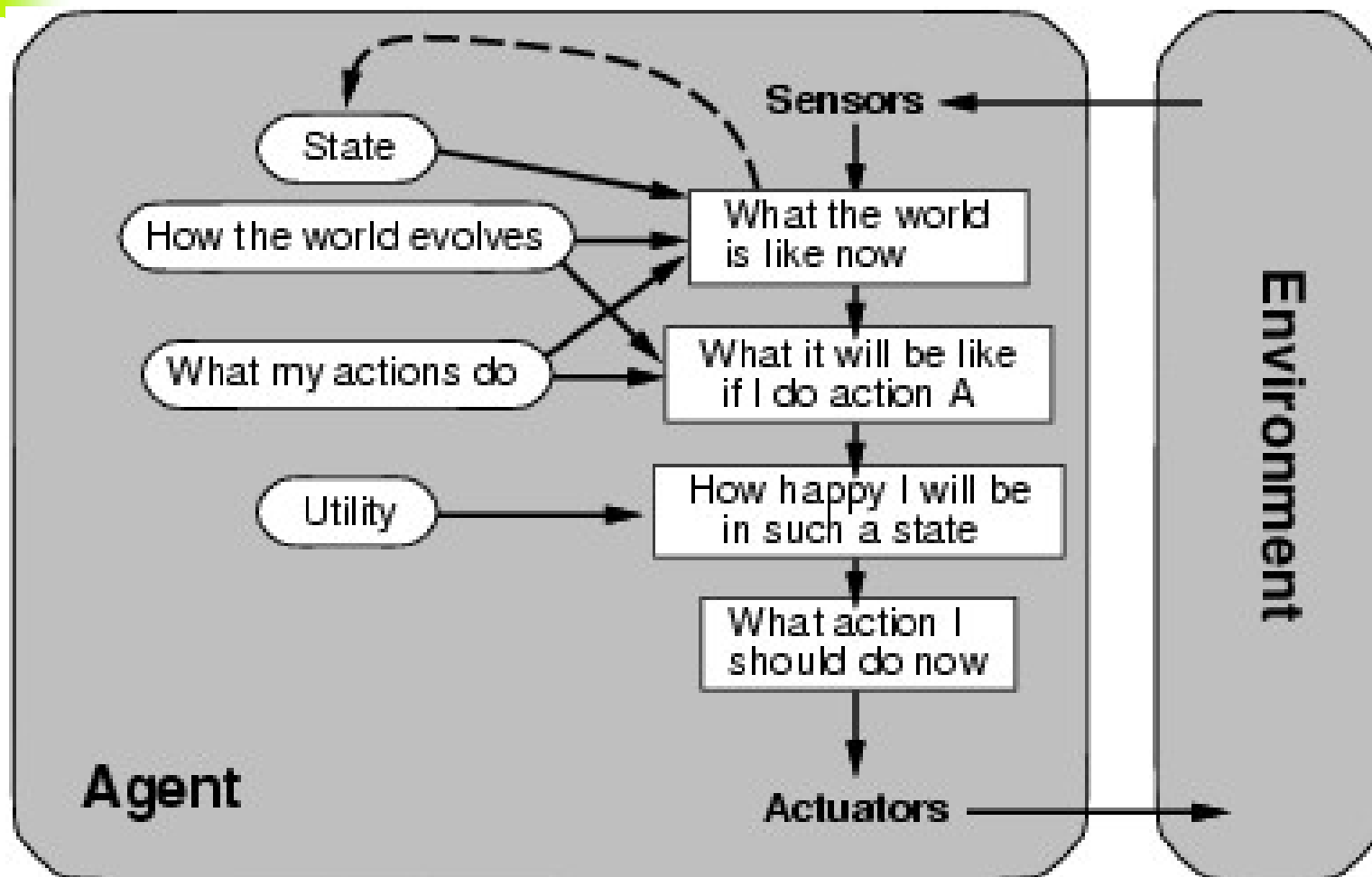
function REFLEX-AGENT-WITH-STATE(*percept*) **returns** an action
 static: *state*, a description of the current world state
 rules, a set of condition–action rules
 action, the most recent action, initially none

 state \leftarrow UPDATE-STATE(*state*, *action*, *percept*)
 rule \leftarrow RULE-MATCH(*state*, *rules*)
 action \leftarrow RULE-ACTION[*rule*]
 return *action*

Goal-Based Agent



Utility-Based Agent

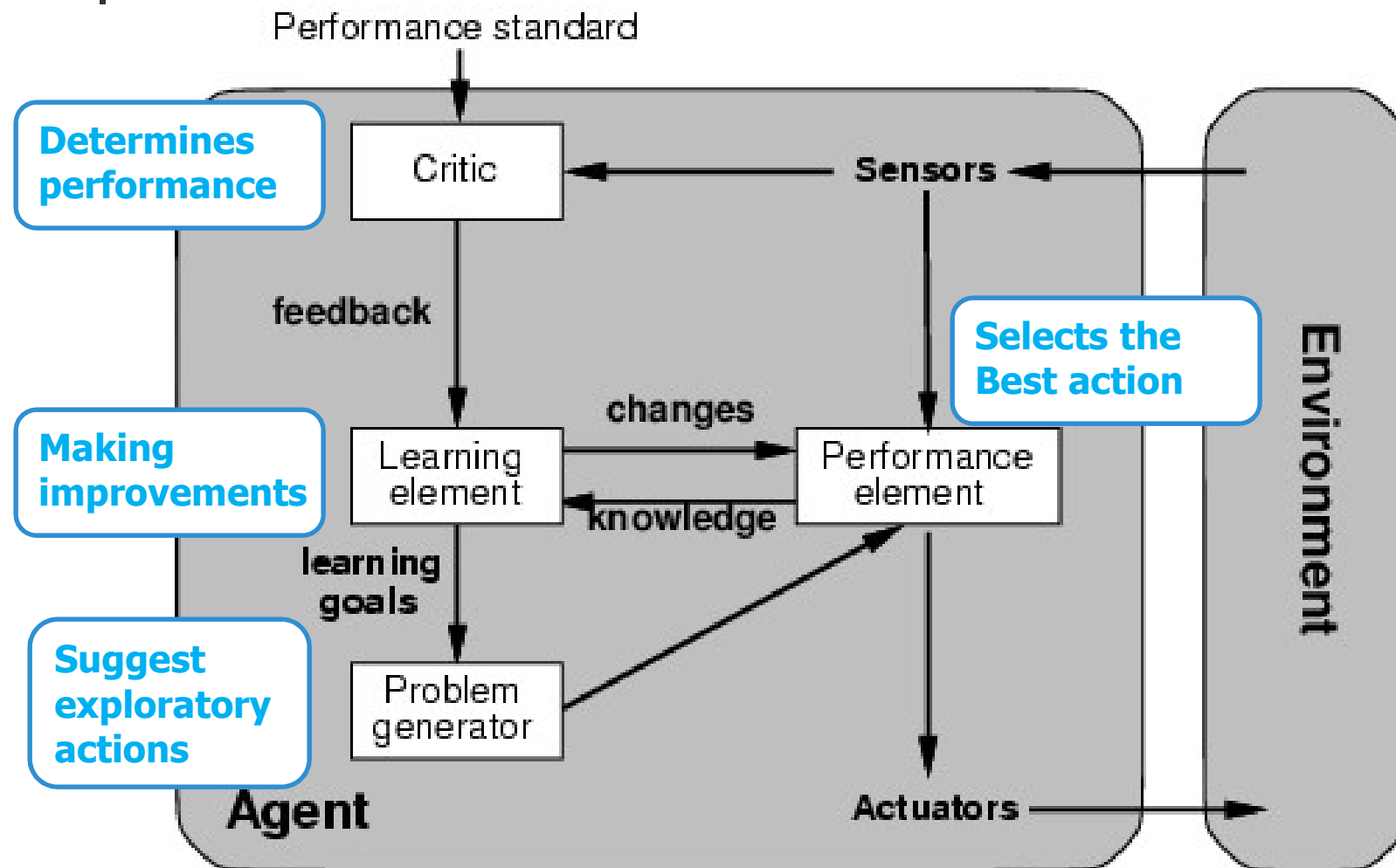




Utility Function

- Utility function maps a state or a sequence of states onto a real number \rightarrow degree of happiness
- Conflicting goals
 - Speed and safety
- Multiple goals

Learning Agent





Exercise

- Select a suitable agent design for:
 - Robot soccer player
 - Shopping for used AI books on the Internet



Summary

- Agent, agent function, agent program
 - Rational agent and its performance measure
 - PEAS
 - Five major agent program skeletons
-
- Next, solving problems by searching