Intelligent Agents

CE417: Introduction to Artificial Intelligence
Sharif University of Technology
Spring 2016

Soleymani

"Artificial Intelligence: A Modern Approach", Chapter 2

Outline

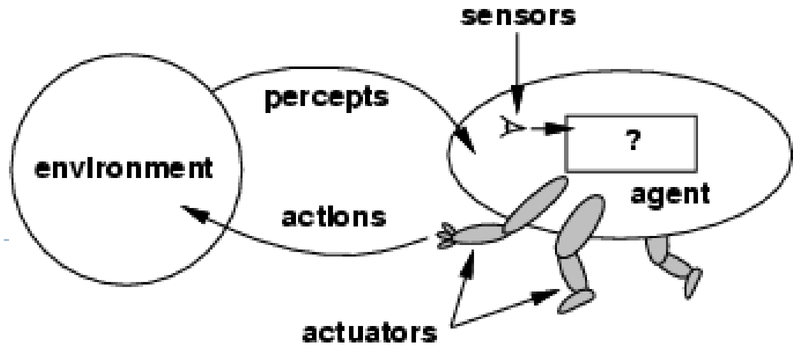
- Agents and environments
- Rationality
- PEAS (Performance measure, Environment, Actuators, Sensors)
- Environment types
- Agent types

Agents

- An agent is anything that can be viewed as
  - **Sensors**: perceive environment
  - **Actuators**: act upon environment

- Samples of agents
  - Human agent
    - Sensors: eyes, ears, and other organs for sensors
    - Actuators: hands, legs, vocal tract, and other movable or changeable body parts
  - Robotic agent
    - Sensors: cameras and infrared range finders
    - Actuators: various motors
  - Software agents
    - Sensors: keystrokes, file contents, received network packages
    - Actuators: displays on the screen, files, sent network packets

Agents & environments



- Agent behavior can be described as an agent function that maps entire perception histories to actions:

$$f: P^* \rightarrow A$$

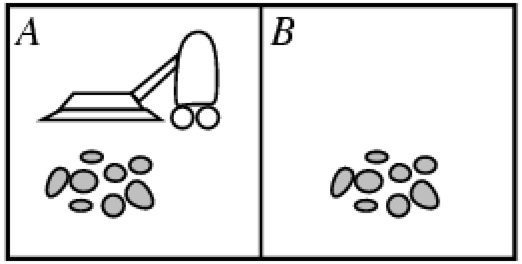
Percept sequence to date

Action set

- The agent program runs on the physical architecture to produce *f*
  - Program is a concrete implementation of agent function
  - Architecture includes sensors, actuators, computing device

agent = architecture + program

Vacuum-cleaner world



- Percepts: location and dirt/clean status of its location
  - e.g., [A,Dirty]

- Actions: *Left*, *Right*, *Suck*, *NoOp*

A vacuum-cleaner agent

Tabulation of the agent function

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
…	…
[A, Clean], [A, Clean], [A, Clean]	Right
[A, Clean], [A, Clean], [A, Dirty]	Suck
…	…

One simple rule implementing the agent function:

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

Rational agents

- "**do the right thing**" based on the perception history and the actions it can perform.

- **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.

Performance measure

- Evaluates the sequence of environment states
- Vacuum-cleaner agent: samples of performance measure
  - ✗ Amount of dirt cleaned up
  - ☑ One point award for each clean square at each time step
    - Penalty for electricity consumption & generated noise
    - Mediocre job or periods of high and low activation?

Rational agents (vacuum cleaner example)

- Is this rational? <u>If dirty then suck, otherwise move to the other square</u>
  - Depends on
    - Performance measure, e.g., Penalty for energy consumption?
    - Environment, e.g., New dirt can appear?
    - Actuators, e.g., No-op action?
    - Sensors, e.g., Only sense dirt in its location?

Rationality vs. Omniscience

- Rationality is distinct from omniscience (all-knowing with infinite knowledge, impossible in reality)

- Doing actions in order to modify future percepts to obtain useful information
  - information gathering or exploration (important for rationality)
    - e.g., eyeballs and/or neck movement in human to see different directions

Autonomy

- An agent is <u>autonomous</u> if its behavior is determined by its own experience (with ability to <u>learn</u> and <u>adapt</u>)
  - Not just relies only on prior knowledge of designer
  - Learns to compensate for partial or incorrect prior knowledge
    - Benefit: changing environment
    - Starts by acting randomly or base on designer knowledge and then learns form experience
  - Rational agent should be autonomous

- Example: vacuum-cleaner agent
  - <u>If dirty then suck, otherwise move to the other square</u>
    - Does it yield an autonomous agent?
  - learning to foresee occurrence of dirt in squares

Task Environment (PEAS)

- Performance measure
- Environment
- Actuators
- Sensors

PEAS Samples…

- Agent: 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, display
  - Sensors: Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

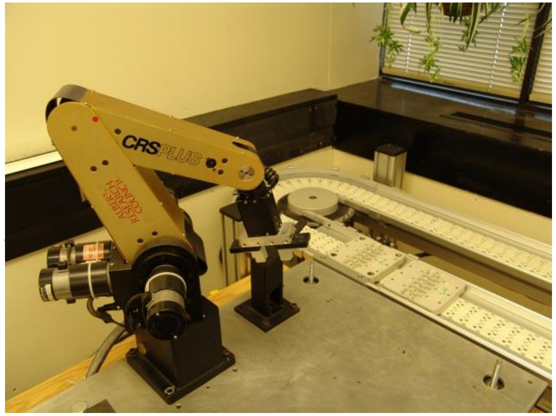
PEAS Samples…

- Agent: Medical diagnosis system
  - Performance measure: Healthy patient, minimize costs
  - Environment: Patient, hospital, staff
  - Actuators: Screen display (questions, tests, diagnoses, treatments, referrals)
  - Sensors: Keyboard (entry of symptoms, findings, patient's answers)

PEAS Samples…

- Satellite image analysis system
  - Performance measure: Correct image categorization
  - Environment: Downlink from orbiting satellite
  - Actuators: Display of scene categorization
  - Sensors: Color pixel array

PEAS Samples…



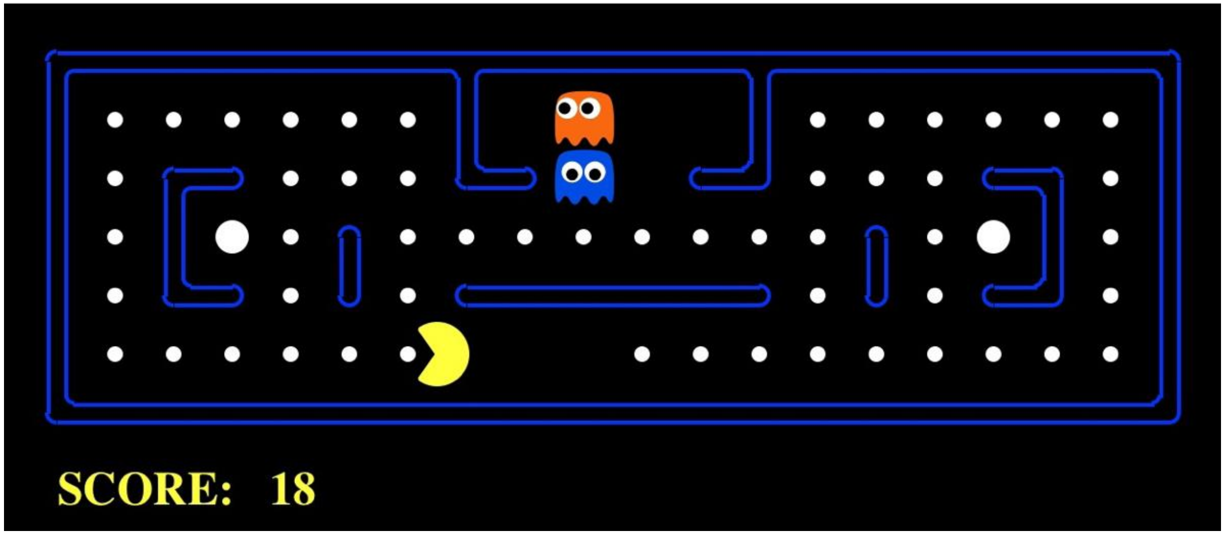
- Agent: Part picking robot
  - Performance measure: Percentage of parts in correct bins
  - Environment: Conveyor belt with parts, bins
  - Actuators: Jointed arm and hand
  - Sensors: Camera, joint angle sensors

PEAS Samples…

- Agent: Interactive English tutor
  - Performance measure: Maximize student's score on test
  - Environment: Set of students
  - Actuators: Screen display (exercises, suggestions, corrections)
  - Sensors: Keyboard

PEAS Samples…

- Agent: Pacman
  - Performance measure: Score, lives
  - Environment: Maze containing white dots, four ghosts, power pills, occasionally appearing fruit
  - Actuators: Arrow keys
  - Sensors: Game screen



Environment types

- Fully observable (vs. partially observable): Sensors give access to the complete state of the environment at each time
  - Sensors detect all aspects relevant to the choice of action
  - Convenient (need not any internal state)
  - Noisy and inaccurate sensors or missing parts of the state from sensors cause partially observability

Environment types

- Deterministic (vs. stochastic): Next state can be completely determined by the current state and the executed action
  - If the environment is deterministic except for the actions of other agents, then the environment is strategic (we ignore this uncertainty)
  - Partially observable environment could appear to be stochastic.
  - Environment is uncertain if it is not fully observable or not deterministic

Environment types

- Single agent (vs. multi-agent):
  - Crossword puzzle is a single-agent game (chess is a multi-agent one)
  - Is B an agent or just an object in the environment?
    - B is an agent when its behavior can be described as maximizing a performance measure whose value depends on A's behavior.
  - Multi-agent: competitive, cooperative
    - Randomized behavior and communication can be rational

- Discrete (vs. continuous): A limited number of distinct, clearly defined states, percepts and actions, time steps
  - Chess has finite number of discrete states, and discrete set of percepts and actions while Taxi driving has continuous states, and actions

Environment types

- Episodic (vs. sequential): The agent's experience is divided into atomic "episodes“ where the choice of action in each episode depends only on the episode itself.
  - E.g., spotting defective parts on an assembly line (independency)
  - In sequential environments, short-term actions can have long-term consequences
    - Episodic environment can be much simpler

- Static (vs. dynamic): The environment is unchanged while an agent is deliberating.
  - Semi-dynamic: if the environment itself does not change with the passage of time but the agent's performance score does.
  - Static (cross-word puzzles), dynamic (taxi driver), semi-dynamic (clock chess)

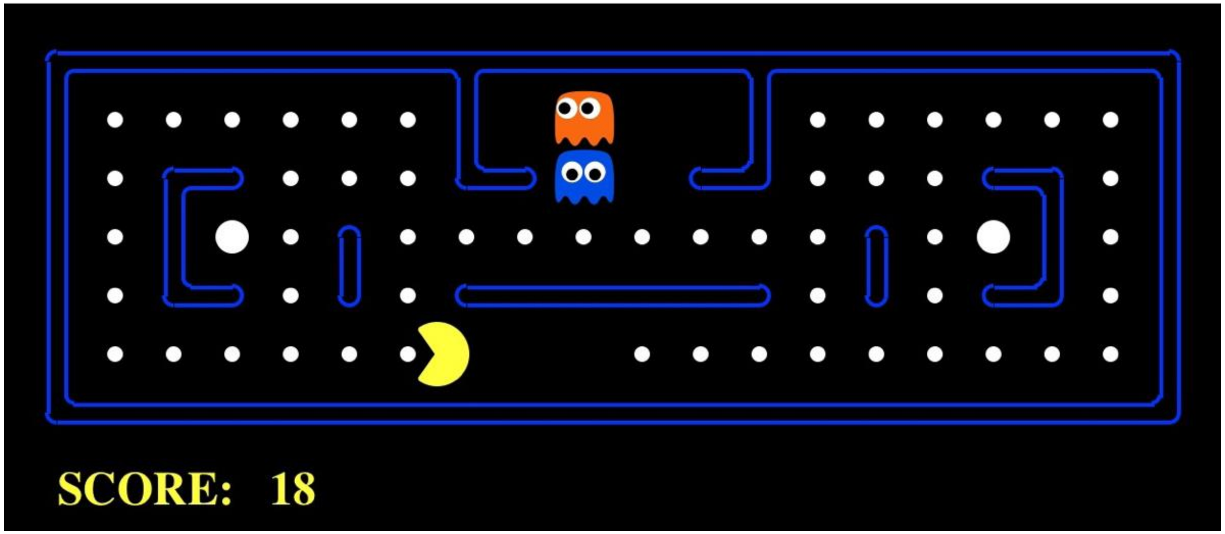
Environment types

- Known (vs. unknown): the outcomes or (outcomes probabilities for all actions are given.
  - It is not strictly a property of the environment
    - Related to agent's or designer's state of knowledge about "laws of physics" of the environment

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

Pacman game

- Fully observable?
- Single-agent?
- Deterministic?
- Discrete?
- Episodic?
- Static?
- Known?



Environment types

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi driving	Partially	Multi	Stochastic.	Sequential	Dynamic	Continuous
Medical diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive. English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Environment types

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi driving	Partially	Multi	Stochastic.	Sequential	Dynamic	Continuous
Medical diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive. English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Structure of agents

- An agent is completely specified by the agent function (that maps percept sequences to actions)
  - One agent function or small equivalent class is rational

- Agent program implements agent function (focus of our course)
  - Agent program takes just the current percept as input
    - Agent needs to remember the whole percept sequence, if requiring it (internal state)

Agent Program Types

- Lookup table
- Basic types of agent program in order of increasing generality:
  - Simple reflexive
  - Model-based reflex agents
  - Goal-based agents → We mainly focus on this type in our course
  - Utility-based agents
  - Learning-based agents

Look Up Table Agents

```
function TABLE-DRIVEN-AGENT(percept) returns an action
    static:   percepts, a sequence, initially empty
              table, a table of actions, indexed by percept sequences, initially fully specified
    append percept to the end of percepts
    action <--LOOKUP(percepts, table)
    return action
```

- Benefits:
  - Easy to implement
- Drawbacks:
  - space ($\sum_{t=1}^{T} |P|^{t}$; P : set of possible percepts, T: lifetime)
    - For chess at least 10^{150} entries while less than 10^{80} atoms in the observable universe
  - the designer have not time to create table
  - no agent could ever learn the right table entries from its experience
  - how to fill in table entries?

Agent program

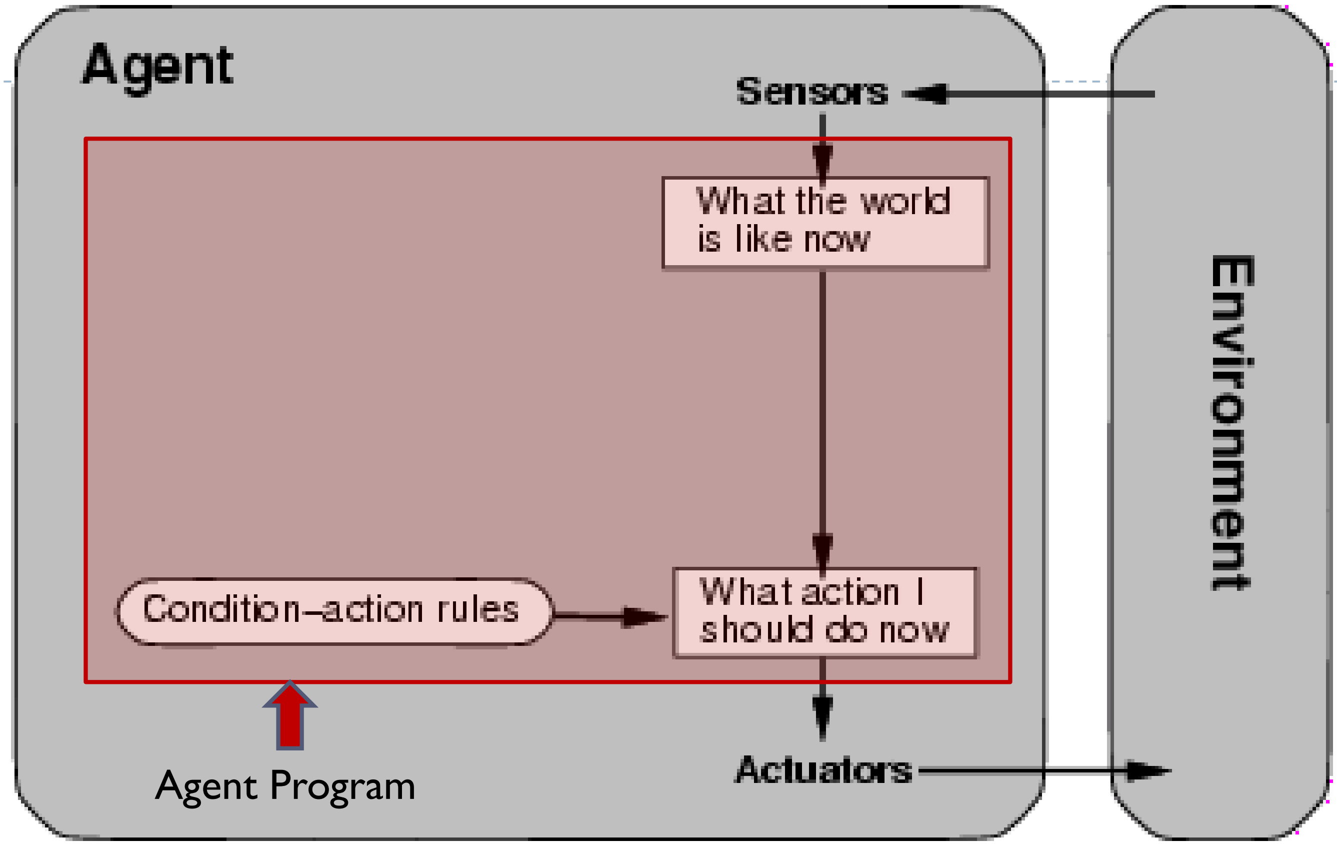
- Mapping is <u>not necessarily</u> using a table.
  - AI intends to find programs producing rational behavior (to the extent possible) from a <u>smallish program instead of a vast table</u>
  - Can AI do for general intelligent behavior what Newton did for square roots?

Huge tables of square roots before 1970s

1.0	1.00000
1.1	1.04898
1.2	1.09565
1.3	1.14056
...	

```
function SQRT( double X )
{
    double r = 1.0 ;
    while ( fabs( r * r - x ) > 10^-8 )
        r = r - ( r * r - x ) / 2r ;
    return r ;
}
```

Simple Reflex Agents



function SIMPLE-REFLEX-AGENT(*percept*) **returns** an action
    static: *rules,* a set if condition-action rules
    state <-- INTERPRET_INPUT(*percept*)
    rule <-- RULE_MATCH(*state, rules*)
    action <-- RULE_ACTION[*rule*]
    return *action*

Simple Reflex Agents

- Select actions on the basis of the **current percept** ignoring the rest of the percept history

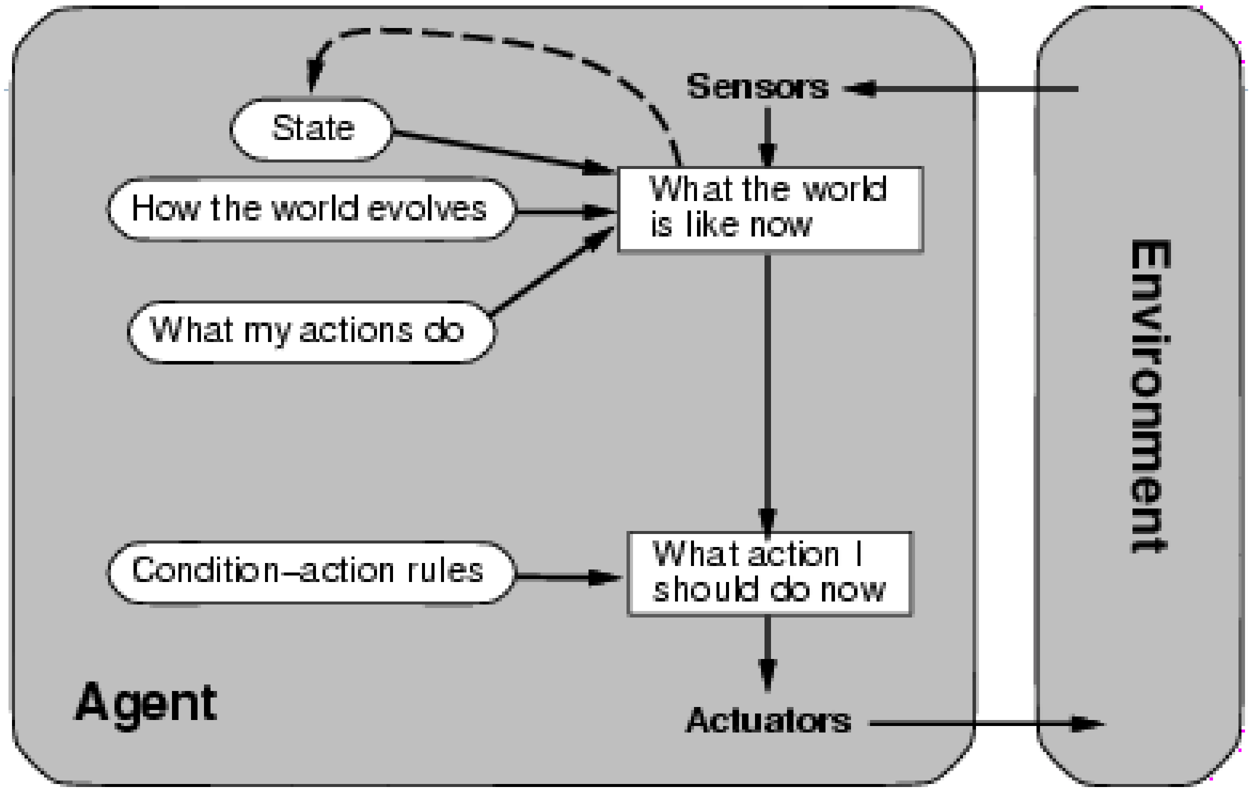
```
function REFLEX-VACUUM-AGENT([location,status]) returns an action
    if status = Dirty then return Suck
    else if location = A then return Right
    else if location = B then return Left
```

- If <u>*car-in-front-is-braking*</u> then <u>*initiate-braking*</u>
- Blinking reflex

Simple Reflex Agents

- Simple, but very limited intelligence
- Works only if the correct decision can be made on the basis of the current percept (<u>fully observability</u>)
- Infinite loops in partially observable environment

Model-based reflex agents



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 <-- UPDATE_INPUT(*state, action, percept*)
    rule <-- RULE_MATCH(*state, rules*)
    action <-- RULE_ACTION[*rule*]
    return *action*

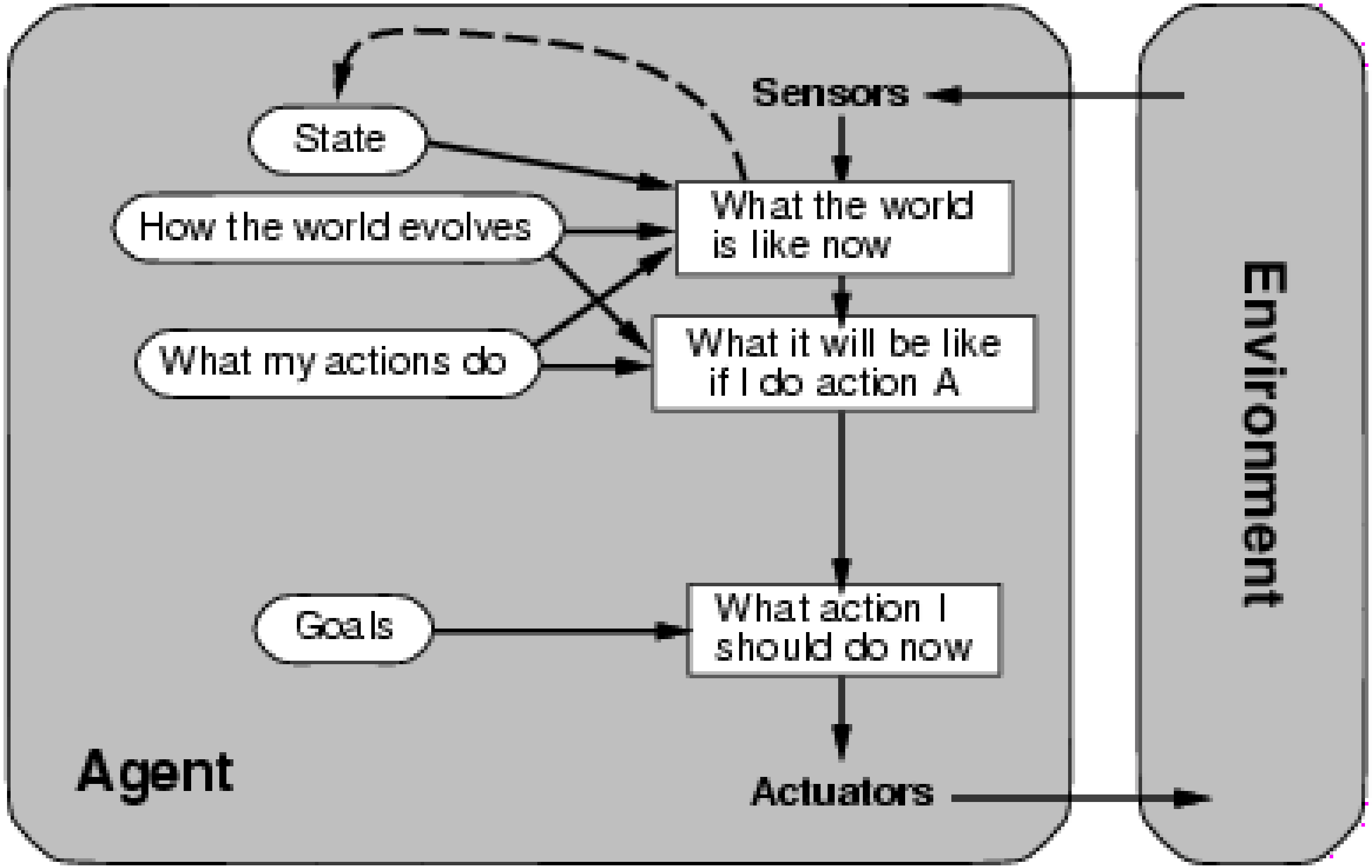
Model-based reflex agents

- Partial observability
  - Internal state (based on percept history)
    - reflects some unobserved aspects of the current state

- Updating the internal state information requires two kinds of knowledge
  - Information about how the world evolves (independent of agent)
  - Information about how the agent's own actions affects the world

- Only determine the best guess for the current state of a partially observable environment

Goal-based agents



Goal-based agents

- Knowing about the current state is not always enough to decide what to do

- Situations that are desirable must be specified (goal)

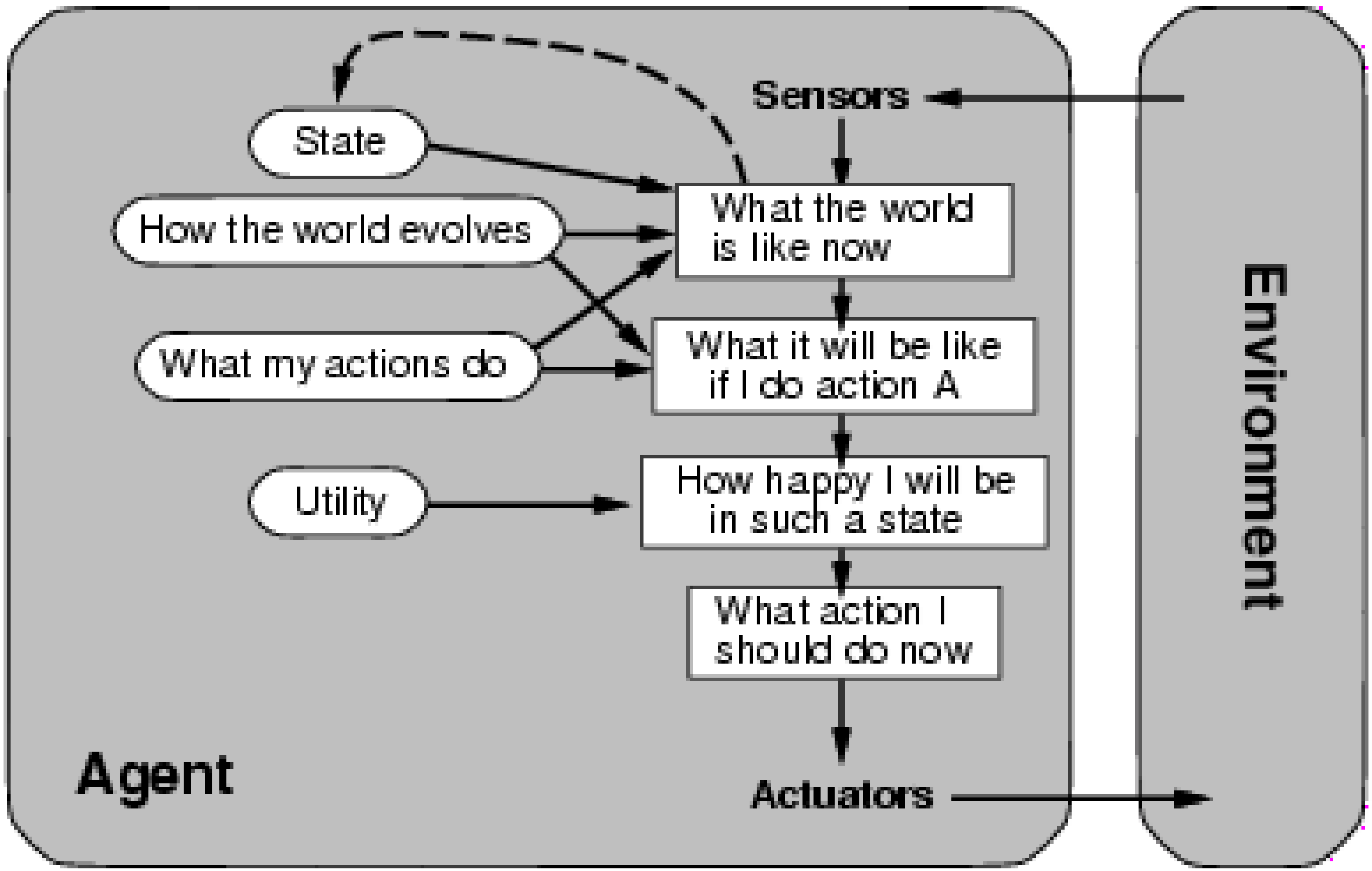
- Usually requires search and planning
  - to find action sequences achieving goal

Goal-based agents vs. reflex-based agents

- Consideration of future

- Goal-based agents may be less efficient but are more flexible
  - Knowledge is represented explicitly and can be changed easily
    - Example: going to a new destination
      - Goal-based agent: specifying that destination as the goal
      - Reflexive agent: agent's rules for when to turn and when to go straight must be rewritten

Utility-based agents



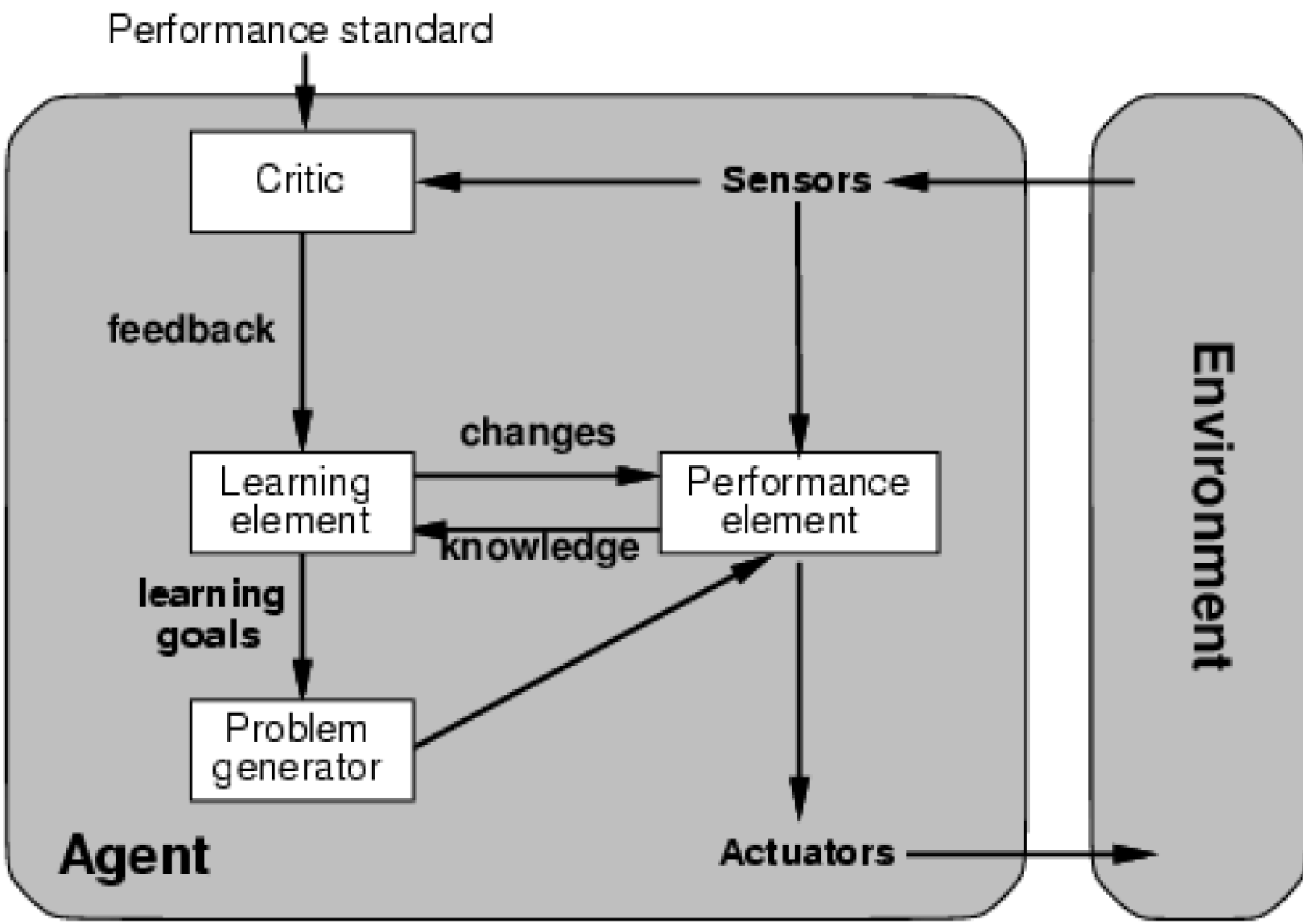
Utility-based agents

- More general performance measure than goals
  - How happy would each world state make the agent?
  - Utility function is an internalization of performance measure

- Advantages
  - Like goal-based agents show flexibility and learning advantages
  - Can trade-off conflicting goals (e.g. speed and safety)
  - Where none of several goals can be achieved with certainty
    - likelihood of success can be weighted by importance of goals

- Rational utility-based agent chooses the action that maximizes the expected utility of action outcomes
  - Many chapters of AIMA book is about this
    - Handling uncertainty in partially observable and stochastic environments

Learning Agents



Learning Agents

- Create state-of-the-art systems in many areas of AI
- Four conceptual components
  - Performance element: selects actions based on percepts (considered as entire agent before)
  - Learning element: makes improvements by modifying "knowledge" (performance element) based on critic feedback
  - Critic: feedbacks on how the agents is doing
  - Problem generator: suggests actions leading to new and informative experiences

- Performance standard
  - Fixed, out of the agent
  - Percepts themselves do not provide indication of success
    - Distinguishes part of the incoming percept as a reward or penalty

Learning Agents

- Learning element is based on performance element (i.e., agent design)

- The learning element can make changes to any of the "knowledge" components of previous agent diagrams
  - To bring components into closer agreement with feedback (yielding better overall performance)