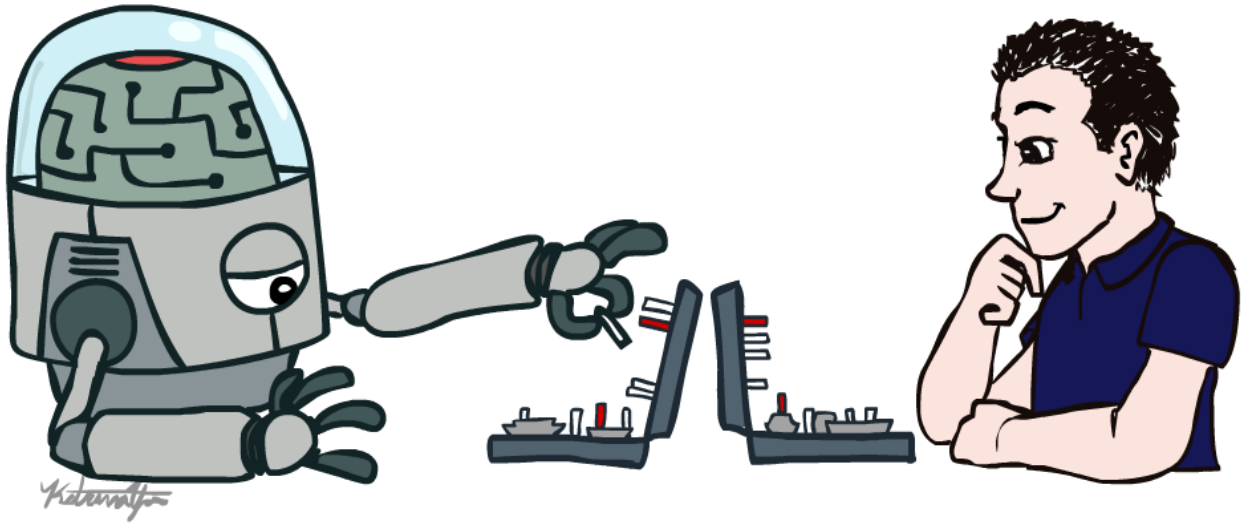
Artificial Intelligence: Basics & Applications

Intelligent Agents (Chapter 2)



Instructor: ***Mahdi Javanmardi***

Amirkabir University of Technology

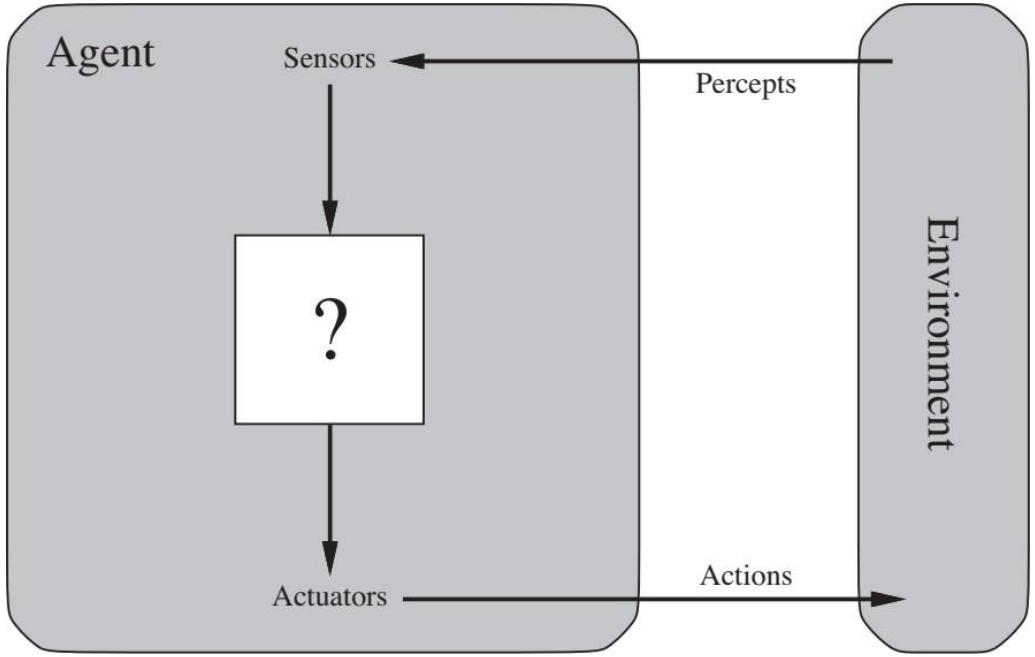
[slides adapted from Dr. Roshanfekr @aut]

Outline

- Agents and Environments
- Good Behavior: The Concept of Rationality
- The Nature of Environments
- The Structure of Agents

Agents

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



Samples of agents

- Human agent
  - Sensors: eyes, ears, and other organs for sensors
  - Actuators: hands, legs, vocal tract, and etc.
- 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 and Environments

- An agent's behavior is described by the agent function that maps any given percept sequence to an action.

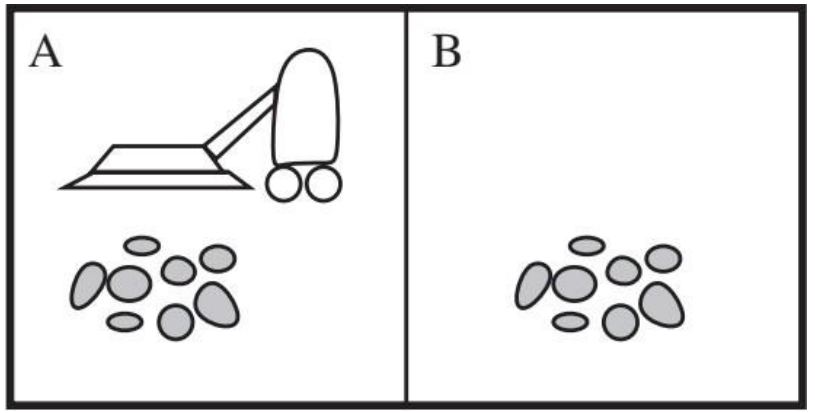
$$f: P^* \rightarrow A$$

  - Can be described by a table
    - For most agents, this would be a very large table
    - The table is, of course, an external characterization of the agent
- Internally, the agent function for an artificial agent will be implemented by an agent program

The agent function is an abstract mathematical description; the agent program is a concrete implementation, running within some physical system

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

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

Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
⋮	⋮
$[A, Clean], [A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Clean], [A, Dirty]$	$Suck$
⋮	⋮

Tabulation of the agent function

Rational agents

- "Do the right thing" based on the perception history and the actions it can perform.
- What is the right thing to do?
  - When an agent is plunked down in an environment, it generates a sequence of actions according to the percepts it receives. This sequence of actions causes the environment to go through a sequence of states. If the sequence is desirable, then the agent has performed well.
  - This notion of desirability is captured by a performance measure

Performance measure

- Evaluates the sequence of environment states
- It depends on environment states, not agent states
- Obviously, there is not one fixed performance measure for all tasks and agents
- samples of performance measure in Vacuum-cleaner agent

✗ Amount of dirt cleaned up

• One point award for each clean square at each time step and Penalty for

✓ • electricity consumption & generated noise

Rationality

- What is rational at any given time depends on four things
  - The performance measure
  - The agent's prior knowledge of the environment
  - The actions that the agent can perform
  - The agent's percept sequence to date
- 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.

Rationality(Vacuum cleaner example)

- Is this rational?
  - If dirty then suck, otherwise move to the other square
- 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
- Omniscient agent
  - An omniscient agent knows the actual outcome of its actions and can act accordingly; but omniscience is impossible in reality.
- Rationality is not the same as perfection
  - Rationality maximizes expected performance, while perfection maximizes actual performance
- Rationality does not require omniscience
  - Because the rational choice depends only on the percept sequence to date

Requirements of a rational agent

- Information gathering or exploration
  - Doing actions in order to modify future percepts to obtain useful information
  - Exploration must be undertaken by a vacuum-cleaning agent in an initially unknown environment
- Learning
  - The agent's initial configuration could reflect some prior knowledge of the environment
  - As the agent gains experience this may be modified and augmented
- Autonomy

Autonomy

- An agent is autonomous if its behavior is determined by its own experience (with ability to learn and adapt)
  - 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
  - If dirty then suck, otherwise move to the other square
    - Does it yield an autonomous agent?
  - learning to foresee occurrence of dirt in squares

Task Environment (PEAS)

- Performance measure
- Environment
- Actuators
- Sensors

In designing an agent, the first step must always be to specify the task environment as fully as possible.

PEAS Samples…

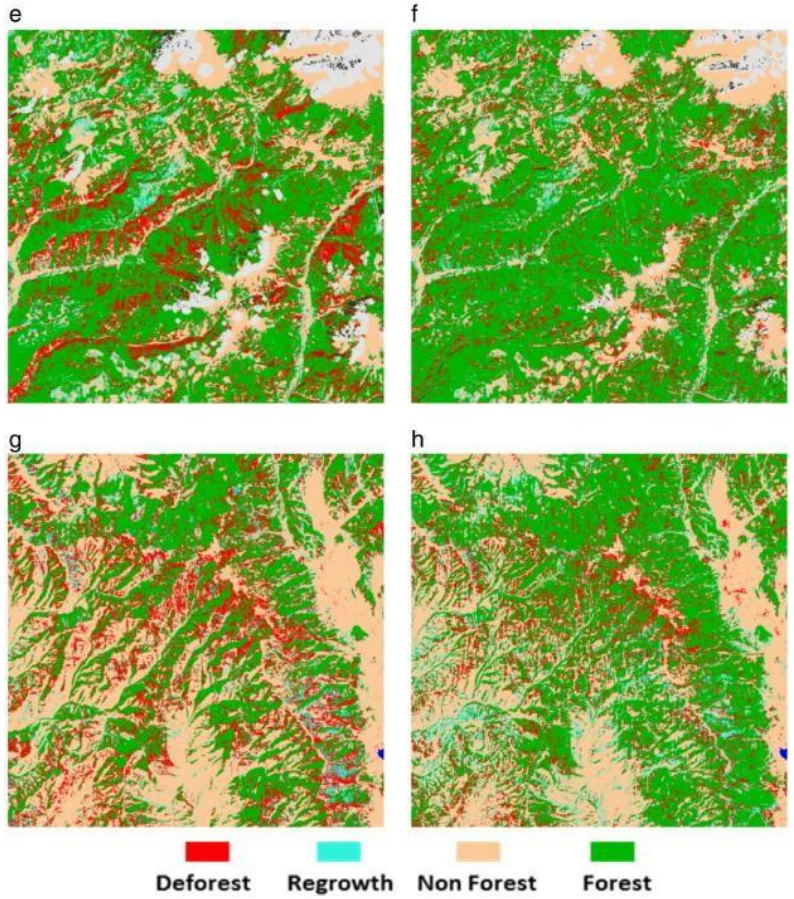
- Agent: Automated taxi driver
  - Performance measure
    - Safe, fast, legal, comfortable trip, maximize profits, …
  - Environment
    - Roads, other traffic, pedestrians, customers, …
  - Actuators
    - Steering wheel, accelerator, brake, signal, horn, display
  - Sensors
    - Cameras, sonar, speedometer, 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…

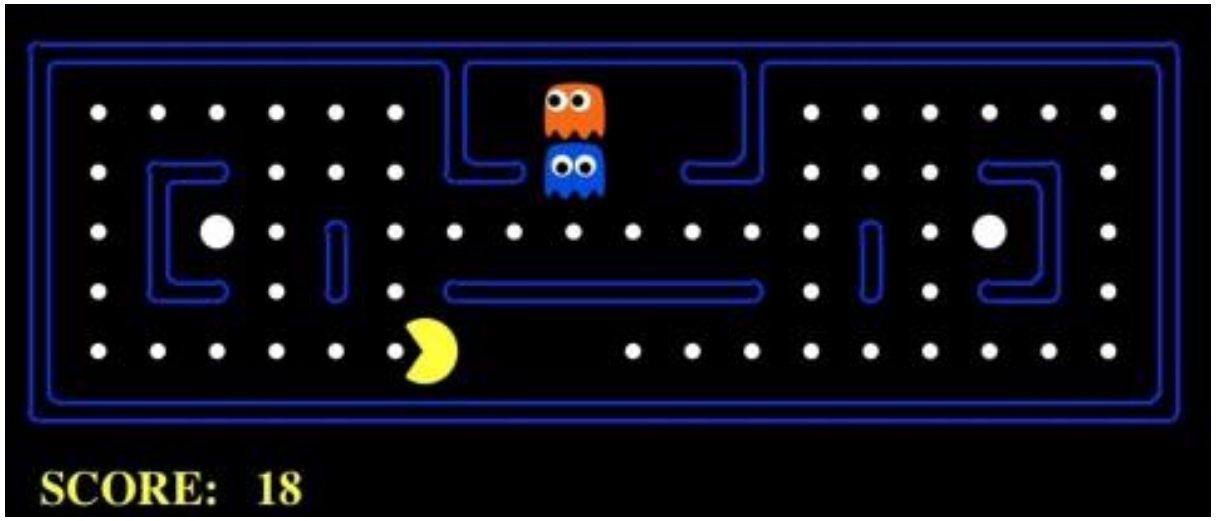
- Agent: Part picking robot
  - Performance measure
    - Percentage of parts in correct bins
  - Environment
    - Conveyor belt with parts, bins
  - Actuators
    - Jointed arm and hand
  - Sensors
    - Camera, joint angle sensors

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

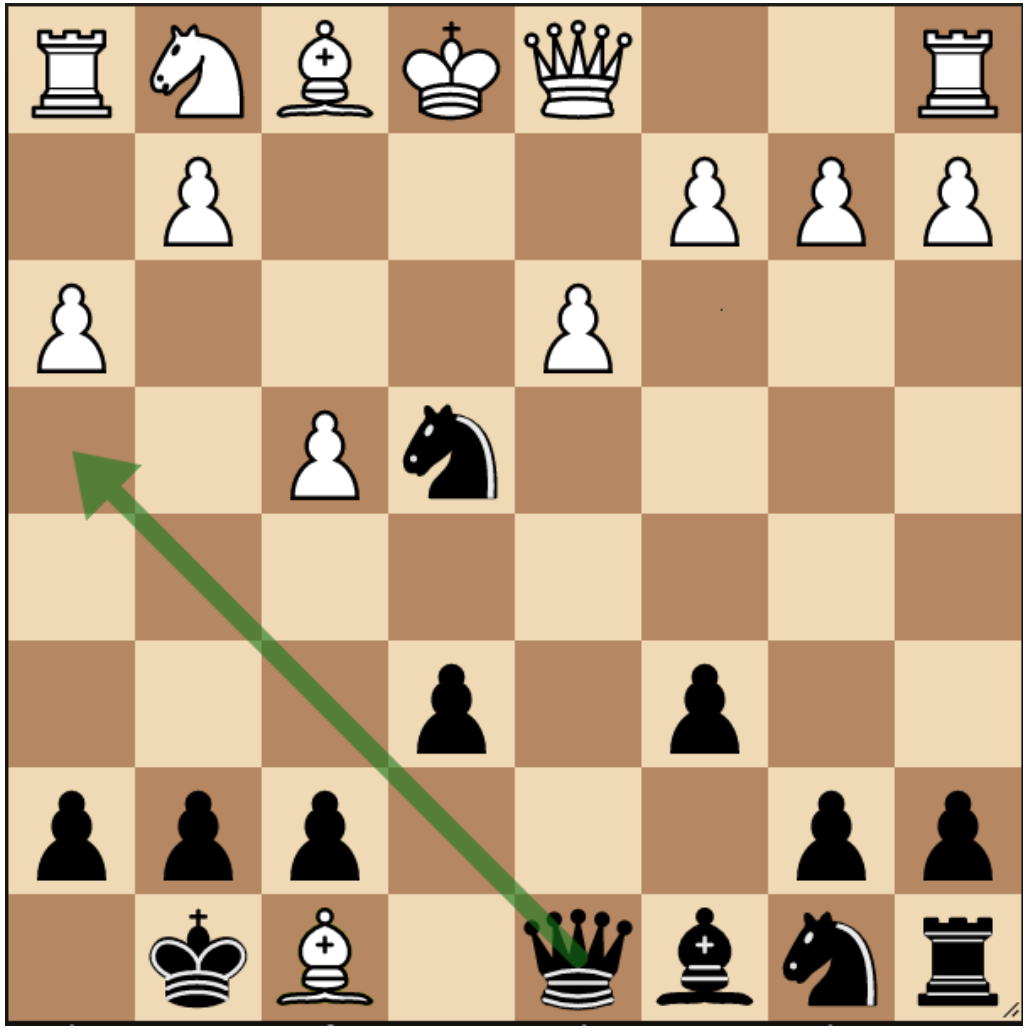


Environment types

- 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
- 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

Environment types - Example

- Find the best move for black



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
- 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 outcome 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
  - If the environment is unknown, the agent will have to learn how it works in order to make good decisions
  - An unknown environment can be fully observable
- The real world
  - partially observable, multi-agent, stochastic, sequential, dynamic, continuous, (and unknown)
  - Hardest type of environment
- The environment type largely determines the agent design

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

agent = architecture + program

- Agent program
  - The job of AI is to design an agent program that implements the agent function
  - Agent program takes just the current percept as input
    - Agent needs to remember the whole percept sequence, if requiring it
      - internal state
- Architecture
  - We assume the program will run on some sort of computing device with physical sensors and actuators

Agent Program Types

- Table-driven agent
- Basic types of agent program in order of increasing generality
  - Simple reflex agents
  - Model-based reflex agents
  - Goal-based agents
  - Utility-based agents
  - Learning-based agents

Table-driven agent

Function TABLE-DRIVEN_AGENT(*percept*) **returns** an action

persistent: *percepts*, a sequence initially empty

table, a table of actions, indexed by percept sequence, initially fully specified

append *percept* to the end of *percepts*

action ← LOOKUP(*percepts*, *table*)

return *action*

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 …

Table-driven agent

- Benefits
  - Easy to implement
- Drawbacks
  - Space
    - $\sigma_{t=1}^{T} |P|^t$ (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 not necessarily using a table
  - AI intends to find programs producing rational behavior (to the extent possible) from a smallish program instead of a vast table

Huge tables of square roots before 1970s

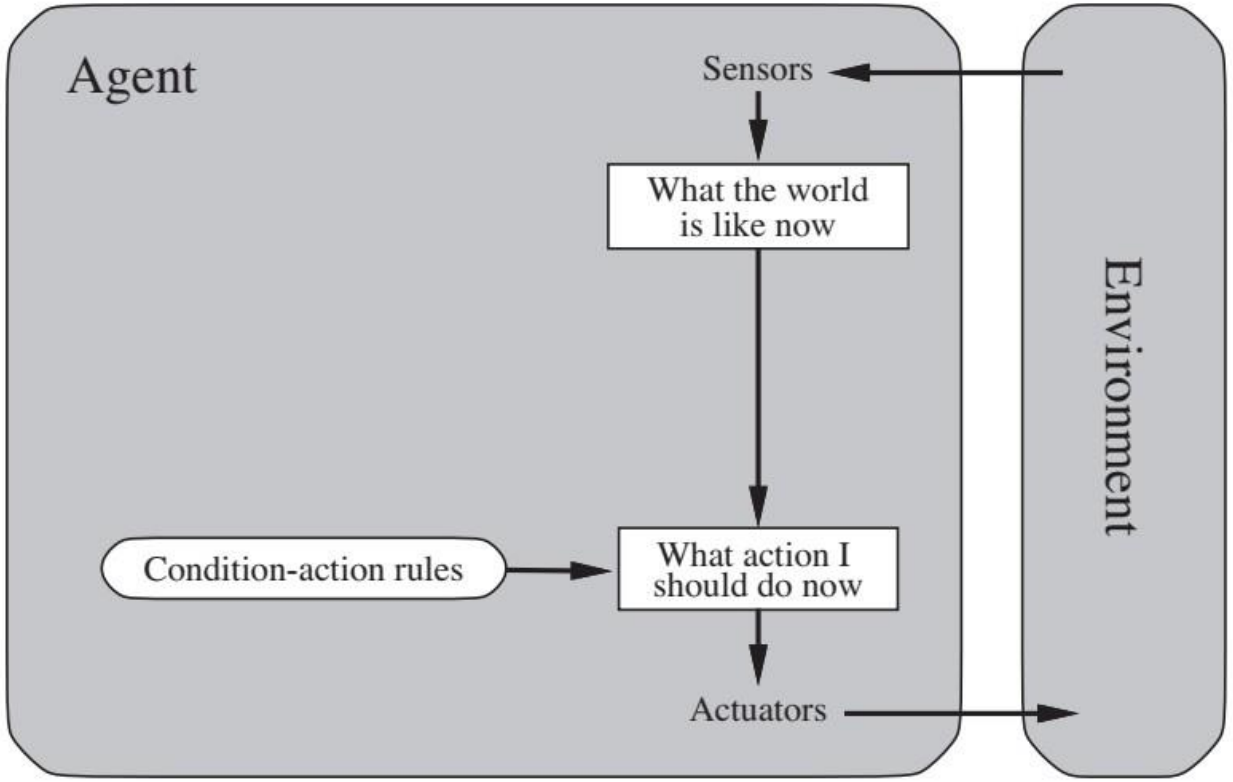
1.0	1.00000
1.1	1.04898
1.2	1.09565
1.3	1.14056
...	

```
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
   persistent: rules, a set of condition-action rules

   state ← INTERPRET-INPUT(percept)
   rule ← RULE-MATCH(state, rules)
   action ← rule.ACTION
   return action
```



Simple Reflex Agents

- Select actions on the basis of the current percept ignoring the rest of the percept history
  - If car-in-front-is-braking then initiate-braking
  - Blinking reflex
- Vacuum agent example

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

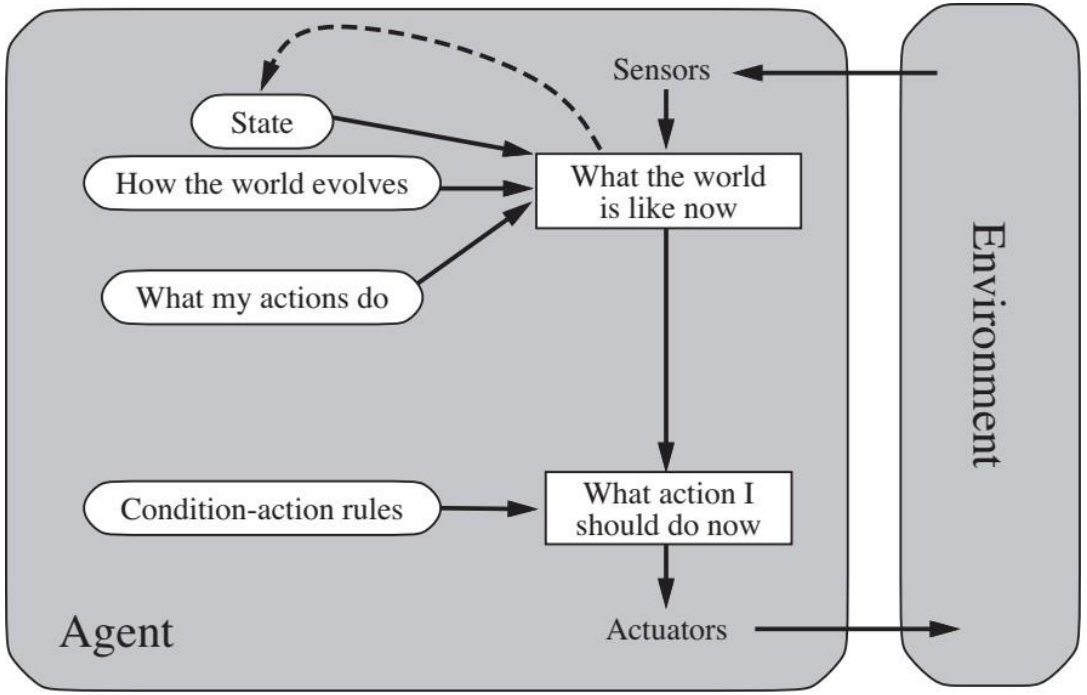
  - Cuts down the number of possibilities from 4^T to just 4

Simple Reflex Agents

- Simple, but very limited intelligence
- Works only if the correct decision can be made on the basis of the current percept (fully observability)
- Infinite loops in partially observable environment
  - Escape from infinite loops is possible if the agent can randomize its actions

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



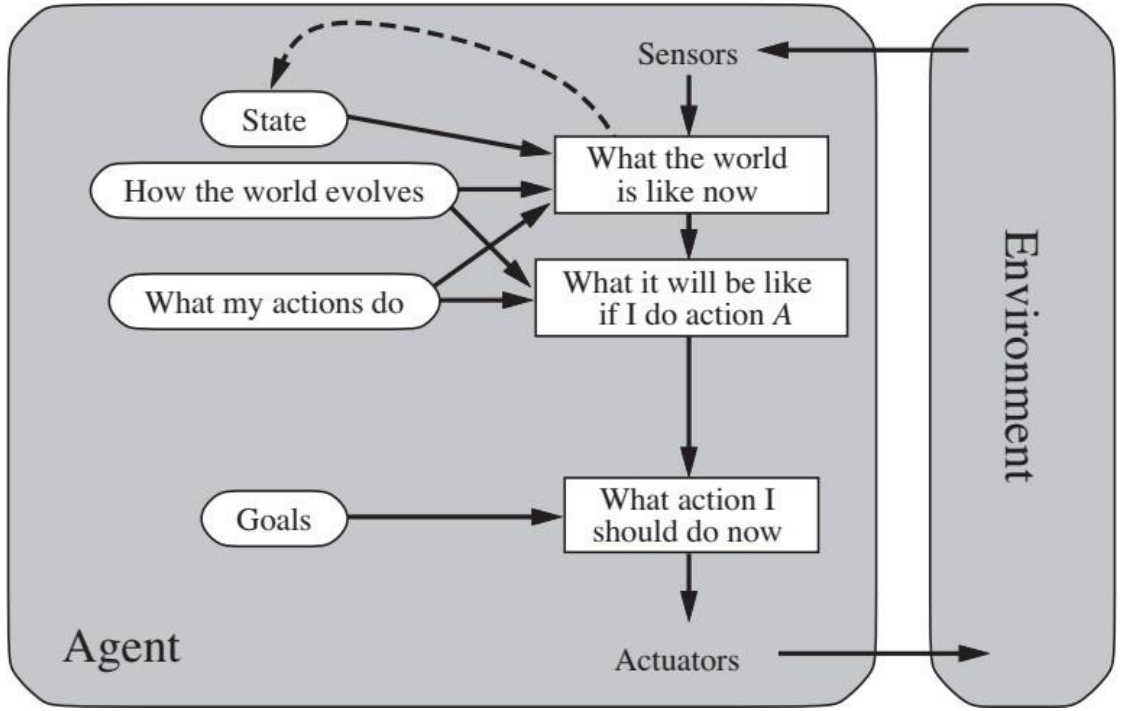
Model-based reflex agents

```
function MODEL-BASED-REFLEX-AGENT(percept) returns an action
   persistent: state, the agent's current conception of the world state
               model, a description of how the next state depends on current state and action
               rules, a set of condition–action rules
               action, the most recent action, initially none

   state ← UPDATE-STATE(state, action, percept, model)
   rule ← RULE-MATCH(state, rules)
   action ← rule.ACTION
   return action
```

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

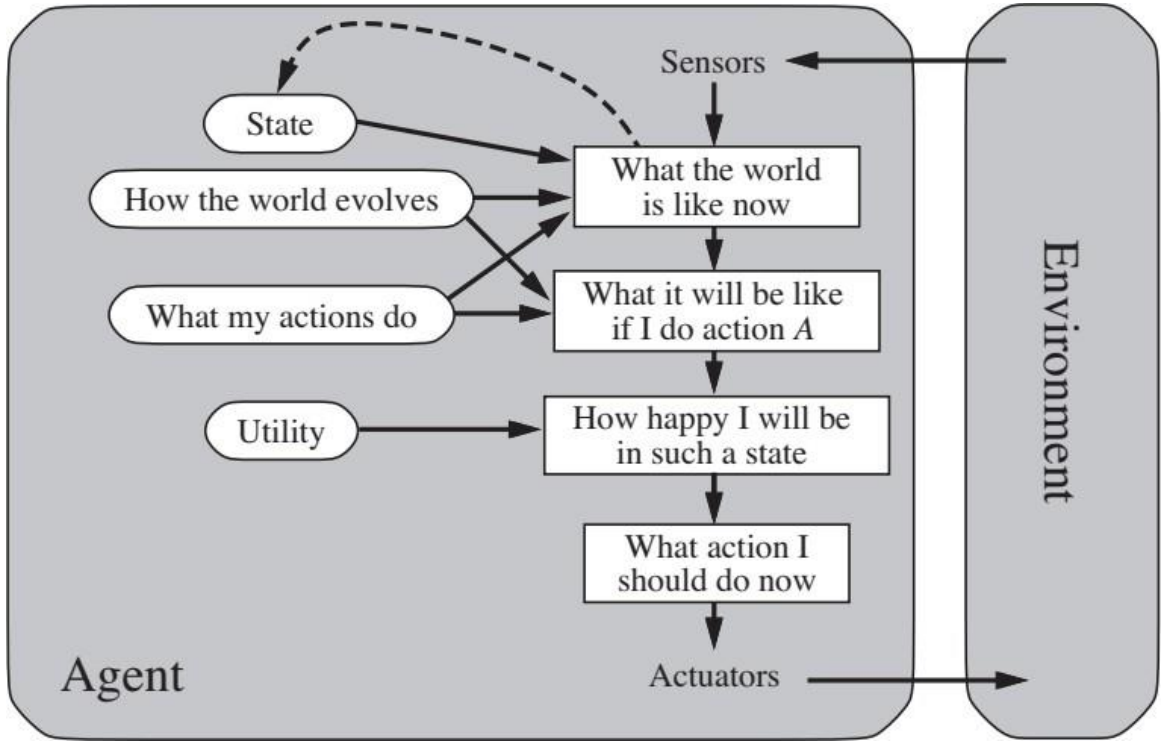


Goal-based agents vs. Reflex-based agents

- 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

- Goals alone are not enough to generate high-quality behavior in most environments.
  - Goals just provide a crude binary distinction between "happy" and "unhappy" states.
  - How to compare different world states according to exactly how happy they would make the agent.
- We use the term "utility" instead of "happy"



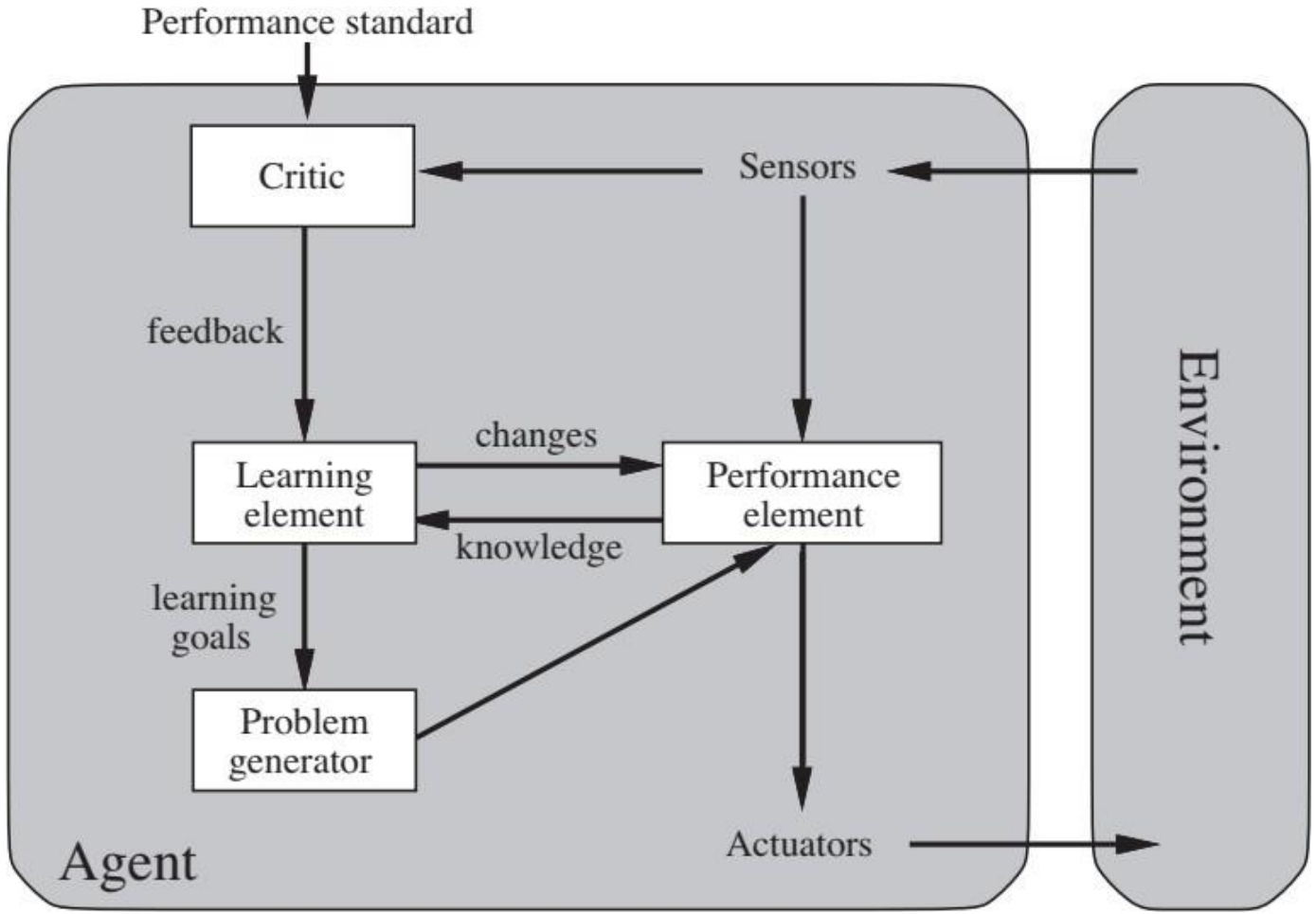
Utility-based agents

- Utility function
  - An agent's utility function is essentially an internalization of the performance measure.
    - performance measure assigns a score to any given sequence of environment states
- If the internal utility function and the external performance measure are in agreement, then an agent that chooses actions to maximize its utility will be rational according to the external performance measure.
  - This is not the only way to be rational
- Like goal-based agents, a utility-based agent has many advantages in terms of flexibility and learning.

Utility-based agents

- In two kinds of cases, goals are inadequate but a utility-based agent can still make rational decisions
  - When there are conflicting goals, only some of which can be achieved, the utility function specifies the appropriate tradeoff.
    - For example, speed and safety
  - When there are several goals that the agent can aim for, none of which can be achieved with certainty, utility provides a way in which the likelihood of success can be weighed against the importance of the goals
- A rational utility-based agent chooses the action that maximizes the expected utility of the action outcomes
- An agent that possesses an explicit utility function can make rational decisions with a general-purpose algorithm

Learning-based agents

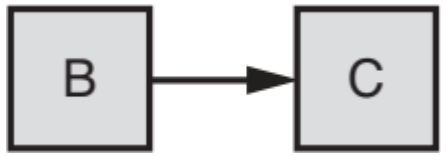


Learning-based agents

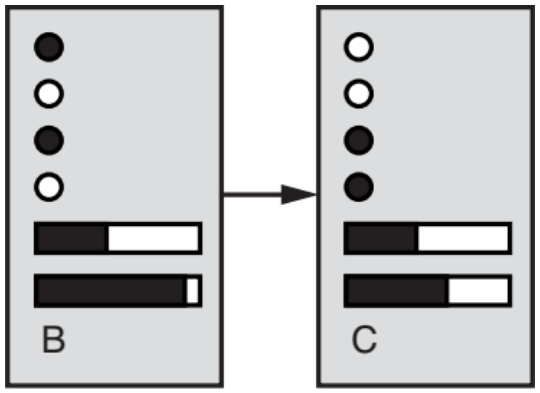
- A learning agent can be divided into four conceptual components
  - Performance element (Previously considered to be the entire agent)
    - Responsible for selecting external actions
  - Critic
    - Tells the learning element how well the agent is doing with respect to a fixed performance standard
  - Learning element
    - Responsible for making improvements
      - Modifies the performance element to do better in future based on feedbacks from the critic
  - Problem generator
    - Responsible for suggesting actions that will lead to new and informative experiences.

Representing states

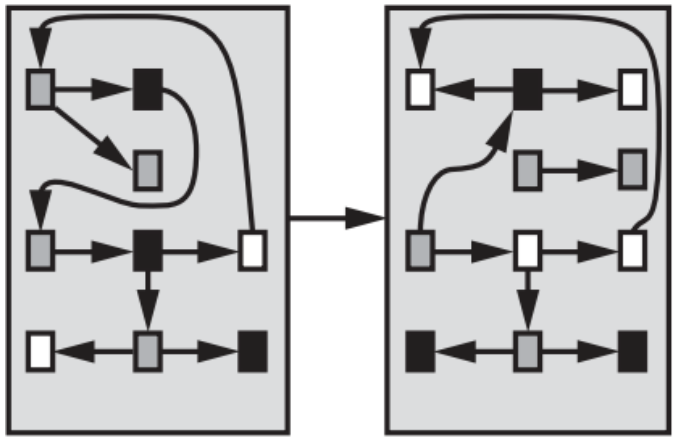
- Atomic representation
  - Each state of the world is indivisible.
  - It has no internal structure
- Factored representation
  - Splits up each state into a fixed set of variables or attributes, each of which can have a value
- Structured representation
  - Objects and their various and varying relationships can be described explicitly.



(a) Atomic



(b) Factored



(b) Structured