Fundamentals of Artificial Intelligence

Intelligent Agents

Agents and Environments

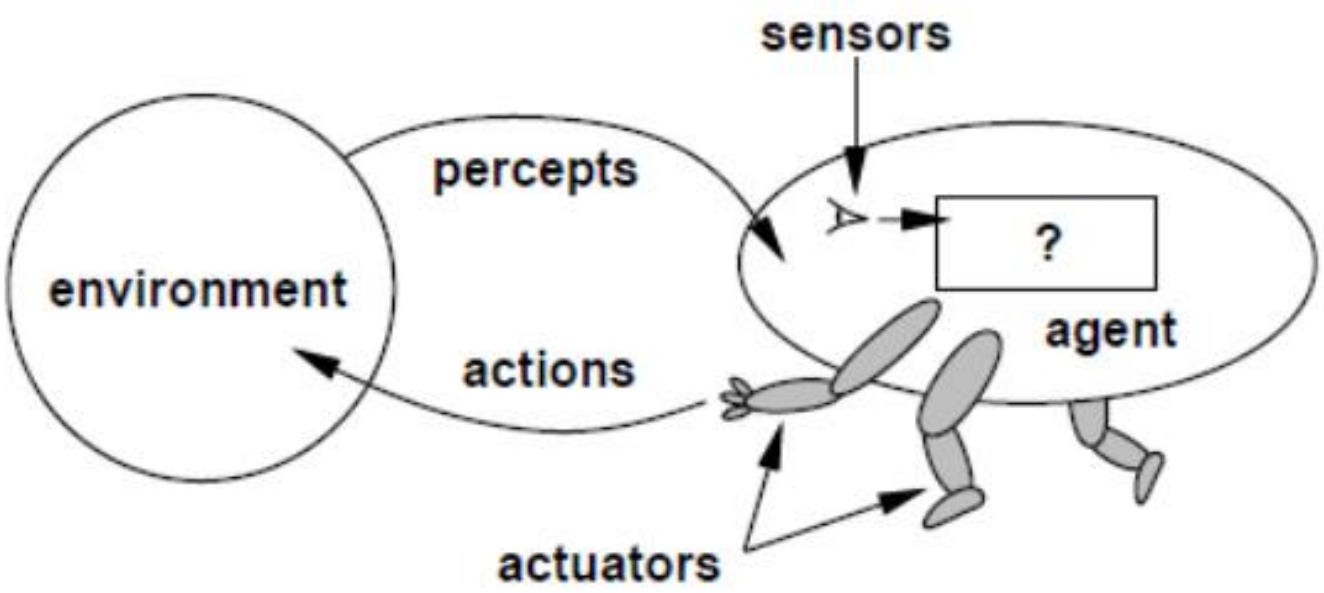
- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and acting upon that environment through **actuators**.
- The term **percept** to refer to the agent's perceptual inputs at any given instant.
- An agent's **percept sequence** is the complete history of everything the agent has ever perceived.
- An agent's behavior is described by the **agent function** that maps any given percept sequence to an action.
  - The agent function for an artificial agent will be implemented by an **agent program**.

Agents and Environments

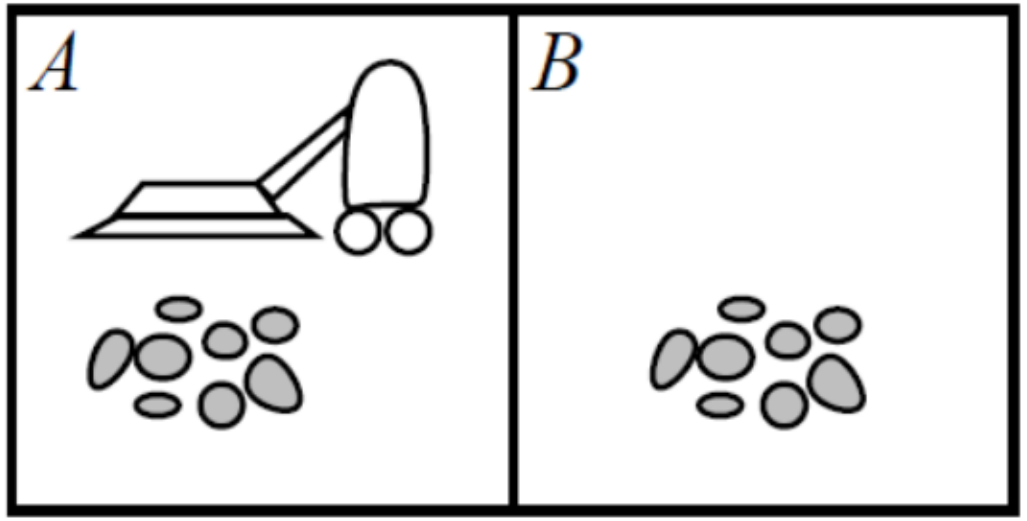
- **Agents** include humans, robots, softbots (software agents), thermostats, etc.
- The **agent function f** maps from **percept histories** to **actions**:

  $f : P^* \rightarrow A$

- The **agent program** runs on the physical architecture to produce f



Vacuum-cleaner world with two locations



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

A vacuum-cleaner agent

- Partial tabulation of a simple agent function for the vacuum-cleaner world

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

- Various vacuum-world agents can be defined by filling in the right-hand column in various ways.
  - What is the right way to fill out the table?
  - In other words, what makes an agent good or bad, intelligent or stupid?

Good Behavior: Rationality

- A **rational agent** is one that does the *right thing*. ➔ What is the *right thing*?
  - A sequence of actions causes the environment to go through a sequence of states.
  - If the sequence is desirable, then the agent has performed well.
- The notion of desirability is captured by a **performance measure** that evaluates any given sequence of ***environment states***.
- A **rational agent** chooses whichever action maximizes the *expected value* of the ***performance measure*** given the *percept sequence*.

Rationality: omniscience, learning, and autonomy

- An **omniscient** (**perfect**) agent knows the actual outcome of its actions and can act accordingly; but perfection is impossible in reality.
- **Rationality** is NOT the same as **perfection**.
  - **Rationality** maximizes *expected performance*, while **perfection** maximizes *actual performance*.
- A rational agent not only to **gather information (exploration)** but also to **learn** as much as possible from what it perceives.
- An agent relies on the prior knowledge of its designer rather than on its own percepts, we say that the agent lacks **autonomy**. A rational agent should be **autonomous.**

- **Rational ➔ exploration, learning, autonomy**

Task Environment: PEAS

- To design *a rational agent*, we must specify the **task environment**.
- The performance measure, the environment, and the agent's actuators and sensors are grouped as the **task environment**, and called as **PEAS** (Performance measure, Environment, Actuators, Sensors).

Agent Type	Performance Measure	Environment	Actuators	Sensors
Taxi driver	Safe, fast, legal, comfortable trip, maximize profits	Roads, other traffic, pedestrians, customers	Steering, accelerator, brake, signal, horn, display	Cameras, sonar, speedometer, GPS, odometer, accelerometer, engine sensors, keyboard

PEAS description of the task environment for an automated taxi

Environment types

- The **environment type** largely determines the agent design.
- **Fully observable** vs. **partially observable**:
  - If an agent's sensors give it access to the complete state of the environment at each point in time, then task environment is called as **fully observable**.
    - Fully observable environments are convenient because the agent need not maintain any internal state to keep track of the world.
  - An environment might be **partially observable** because of noisy and inaccurate sensors.
- **Single agent** vs. **multi agent**:
  - One or more agents. Solving a crossword puzzle is a single agent environment.
  - Chess is a ***competitive*** multi agent environment because an agent tries to maximize its performance while minimizing the performance of other agent.
  - Taxi-driving partially ***cooperative*** multi agent environment because avoiding collisions maximizes all agents' performances.

Environment types

- **Deterministic** vs. **stochastic**:
  - If the next state of the environment is completely determined by the current state and the action executed by the agent, then the environment is **deterministic**; otherwise, it is **stochastic**.
- **Episodic** vs. **sequential**:
  - In an **episodic task environment**, the agent's experience is divided into atomic episodes. In each episode the agent receives a percept and then performs a single action. The next episode does not depend on the actions taken in previous episodes.
  - In **sequential environments**, on the other hand, the current decision could affect all future decisions. Chess and taxi driving are sequential: in both cases, short-term actions can have long-term consequences.
- **Static** vs. **dynamic**:
  - If the environment can change while an agent is deliberating, then we say the environment is **dynamic** for that agent; otherwise, it is **static**.
- **Discrete** vs. **continuous**:
  - The discrete/continuous distinction applies to the *state* of the environment.
  - Chess is discrete; Taxi-driving continuous

Environment types: examples of task environments and their characteristics.

- The real world is (of course) partially observable, stochastic, sequential, dynamic, continuous, multi-agent

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

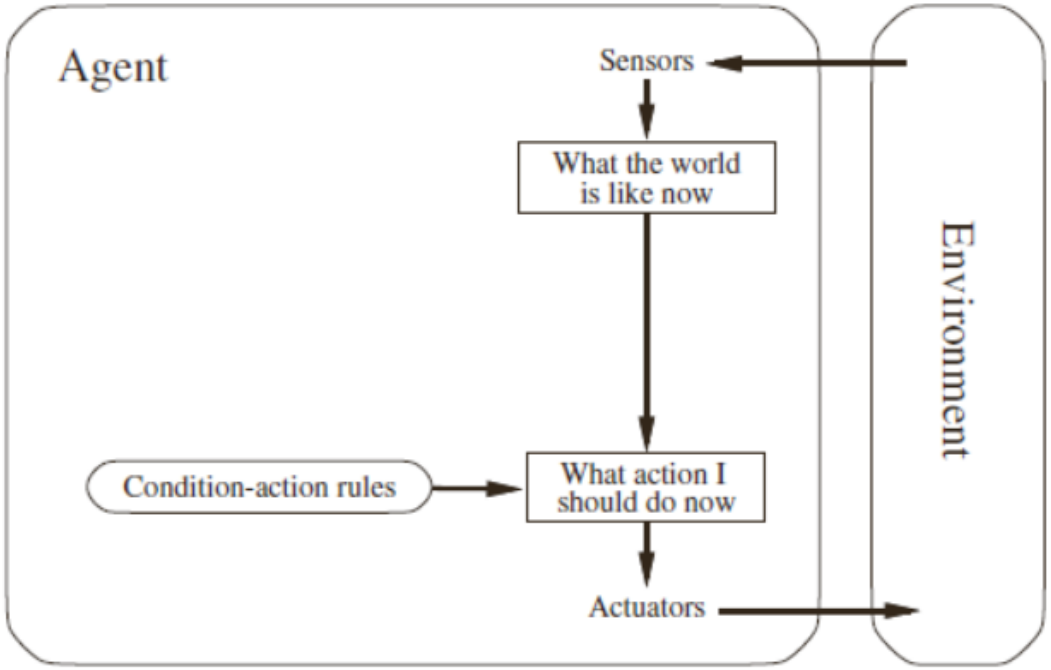
Agent types

- Four basic types in order of increasing generality:
  - simple reflex agents
  - reflex agents with state
  - goal-based agents
  - utility-based agents

- All these can be turned into learning agents

Simple reflex agents

- **Simple reflex agents** select actions on the basis of the *current* percept, ignoring the rest of the percept history.



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*

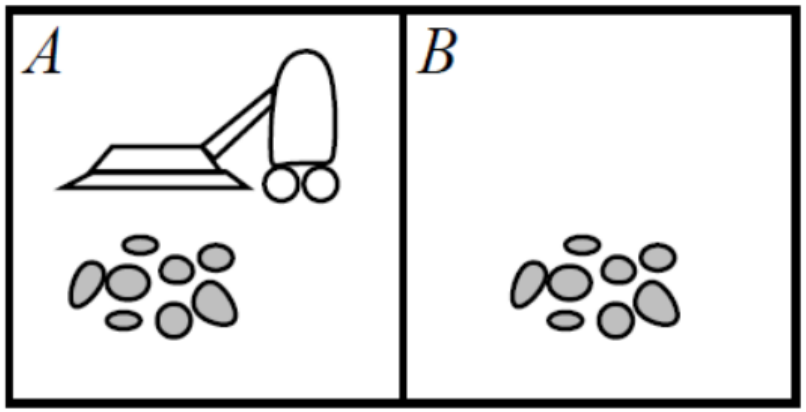
Simple reflex agent in the two-state vacuum environment

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*

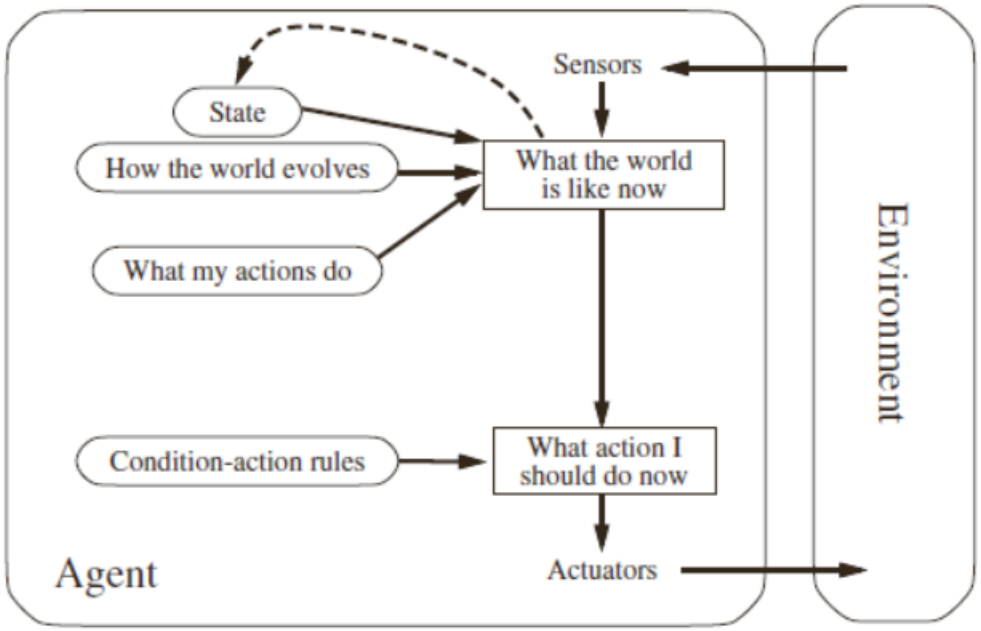
- This program implements the agent function in the following table.

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*
⋮	⋮



Model-based reflex agents

- A **model-based reflex agent** keeps track of the current state of the world, using an internal model. It then chooses an action in the same way as the reflex agent.

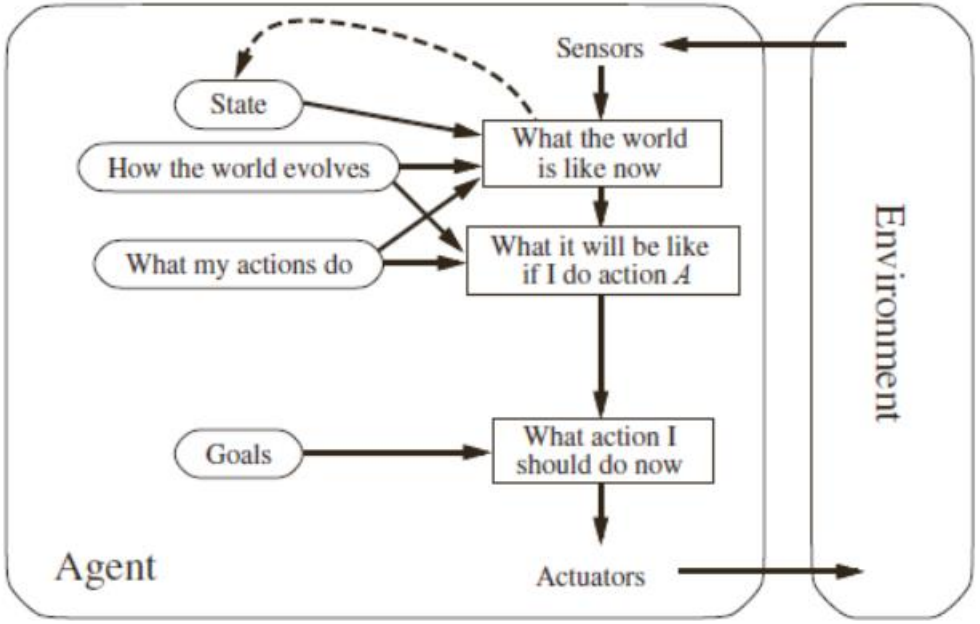


function MODEL-BASED-REFLEX-AGENT(*percept*) **returns** an action
  persistent: *state*, the agent's current conception of the world state
    model, a description of how the next state depends on current state and action
    rules, a set of condition–action rules
    action, the most recent action,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

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

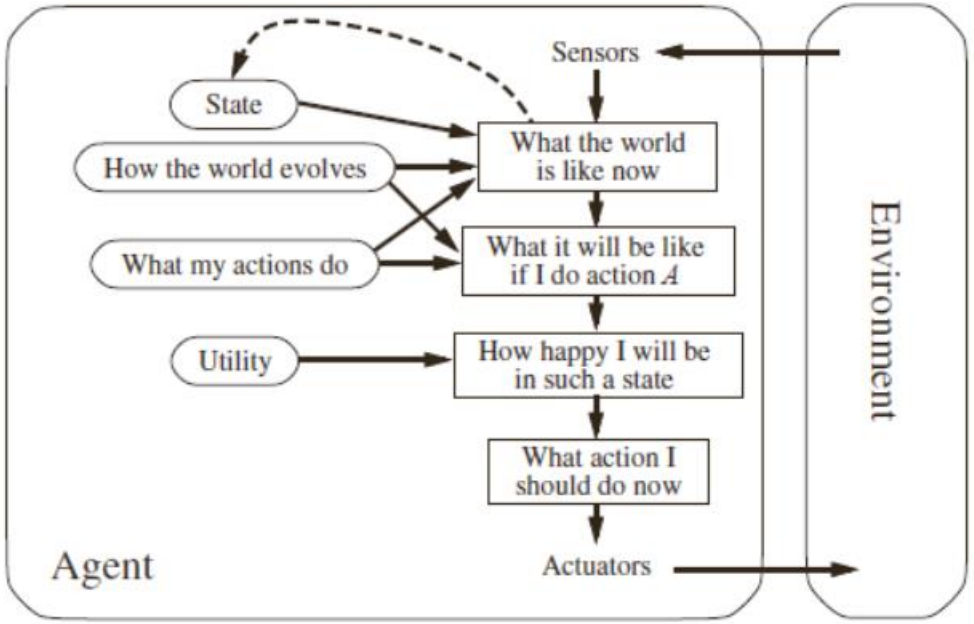
Goal-based agents

- A **goal-based agent** keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.



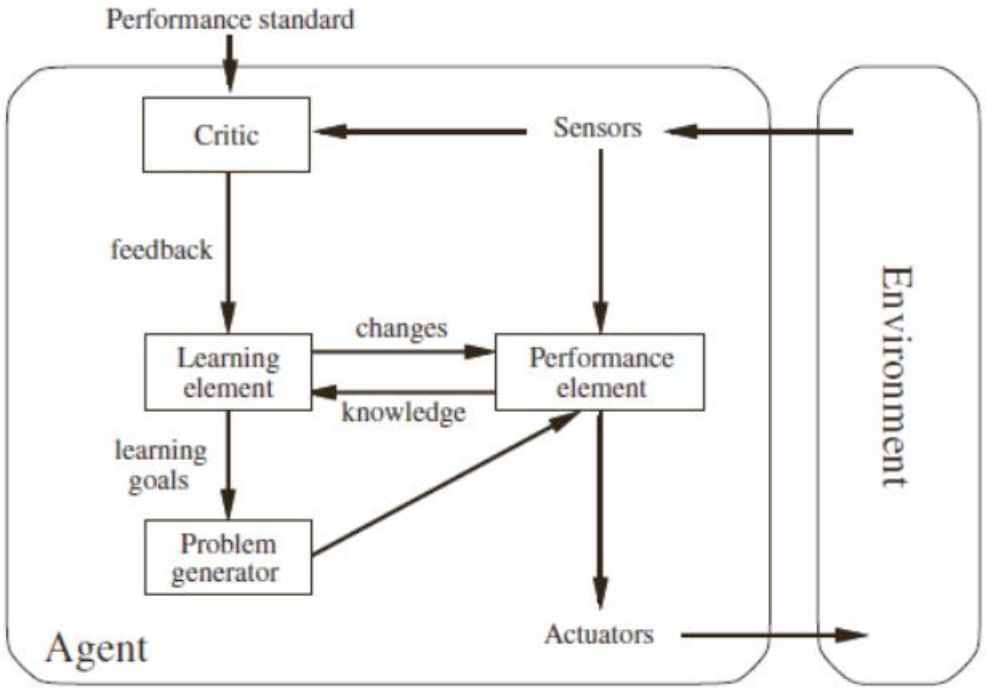
Utility-based agents

- A **utility-based agent** uses a model of the world, along with a utility function that measures its preferences among states of the world.
- Then it chooses the action that leads to the best expected utility, where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.



Learning agents

- **learning element** is responsible for making improvements,
- **performance element** is responsible for selecting external actions.
- The learning element uses feedback from **critic** on how agent is doing and determines how performance element should be modified to do better in future.
- **problem generator** is responsible for suggesting actions that will lead to new and informative experiences.



Intelligent Agents: Summary

- **Agents** interact with **environments** through **actuators** and **sensors**
- The **agent function** describes what the agent does in all circumstances
- The **performance measure** evaluates the environment sequence
- A **rational agent** maximizes ***expected performance***
- **Agent programs** implement (some) agent functions
- **PEAS** descriptions define task environments
- **Environments** are categorized along several dimensions:
  - observable? deterministic? episodic? static? discrete? single-agent?
- Several basic **agent architectures** exist:
  - reflex, reflex with state, goal-based, utility-based