UNIVERSITÀ DEGLI STUDI DI NAPOLI
PARTHENOPE

Artificial Intelligence

Intelligent Agents

LESSON 2

prof. Antonino Staiano

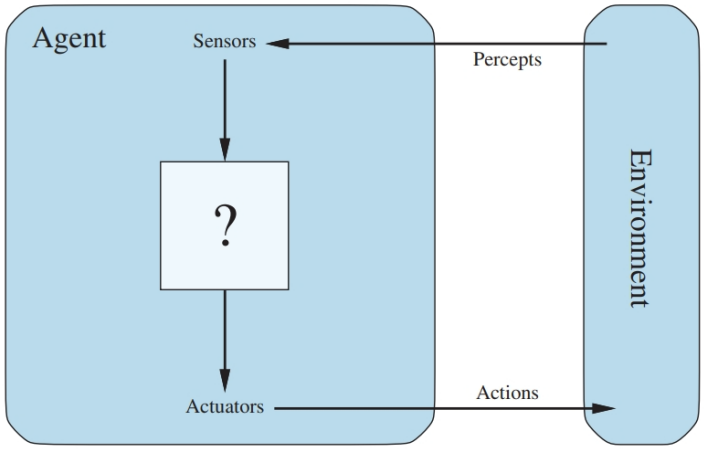
M.Sc. In ''Machine Learning e Big Data'' - University Parthenope of Naples

The Rational Agent Approach to AI

- The concept of rationality can be employed by numerous agents that work in any environment
- The aim is to use this concept to develop some design principles for building successful agents that qualify as intelligent
- Outline
  - Agents and Environments
  - Rationality
  - PEAS (Performance measure, Environment, Actuators, Sensors)
  - Environment types
  - Agent types

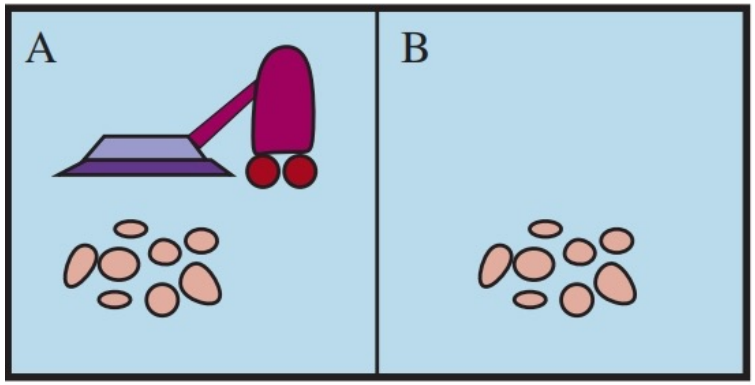
Agents and Environments

- An agent can be anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
  - Agents include humans, robots, softbots, thermostats
- The agent function maps from percept histories to actions
  - f : P* -> A
- The agent program runs on the physical architecture to produce f
  - The agent function is an abstract mathematical description
  - The agent program is a concrete implementation



Example: Vacuum-cleaner world

- The world consists of squares either dirty or clean
  - The agent starts in A



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

A vacuum-cleaner agent

- Simple agent function
  - If the current square is dirty, the suck, else 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
⋮	⋮

What is the right function?

- Can it be implemented in a small agent program?

```
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
```

Defining a good behavior

- An agent in an environment generates a sequence of actions according to its percepts
  - The sequence of actions causes the environment to go through a sequence of states
- If the sequence is desirable, then the agent performed well
- The desirability is captured by a performance measure
  - It evaluates any given sequence of environment states
  - Vacuum-cleaner agent
    - One point per square cleaned up in time T?
    - One point per clean square per time step, minus one per move?
    - Penalize for > k dirty squares?

Rationality and Rational Agent

- Depends on four elements
  - The performance measure that defines the success criterion
  - The agent's prior knowledge of the environment
  - The actions the agent can perform
  - The previous perception of the agent (up to a given time instant)
- A rational agent chooses whichever action maximizes the expected value of the performance measure given the percept sequence to date and whatever built-in knowledge it has

Example: Rational Agent

- Is the vacuum-cleaner agent rational?
- That depends, we need
  - to specify the performance measure
  - knowledge about the environment
  - to know the sensors and actuators equipping the agent
- Example
  - Performance measure: one point for each clean square per time step, over 1000 time steps
  - the environment space is known a priori, the distribution of dirt, and the initial agent location not
    - Clean squares remain and sucking cleans the current square
    - The right and left actions move the agent one square, but the agent stands still if those moves take it out of the environment
    - The only available actions are Right, Left, and Suck
    - The agent correctly perceives its location and whether that location contains dirt
  - In this case, the agent is rational

Rational agent

- Rational ≠ omniscient
  - Assume the most recent forecast is for rain but one did not listen to it and did not bring her umbrella. Is that rational?
  - Yes, since she did not know about the recent forecast!
  - Percepts may not supply all relevant information
- Rational ≠ clairvoyant
  - Action outcomes may not be as expected
- Hence, rational ≠ successful
  - Suppose the forecast is for no rain but one brings her umbrella and uses it to defend herself against a dog attack. Is that rational?
  - No, although successful, it was done for the wrong reason
- Agents don't have to be successful, and they don't have to know everything, just do a good job with what they know and want

Rational Agents

- Rational ->exploration, learning, autonomy
  - Exploration is needed when the rational agent gathers information from an initially unknown environment
  - Learning makes it possible for a rational agent to learn from what it perceives
  - A rational agent should be autonomous, it should learn what it can to compensate for partial or incorrect prior knowledge

Task Environment

- Once we know what rationality is, to be able to build rational agents we need to define task environments
- A task environment specification includes the performance measure, the external environment, the actuators, and the sensors (PEAS)
  - In designing an agent, the first step must always be to specify the task environment as fully as possible

PEAS

- Consider, e.g., the task of designing an automated taxi:
  - Performance measure?
    - safety, destination, profits, legality, comfort, ...
  - Environment?
    - EU streets/freeways, traffic, pedestrians, weather, . . .
  - Actuators?
    - steering, accelerator, brake, horn, speaker/display, . . .
  - Sensors?
    - video, accelerometers, gauges, engine sensors, keyboard, GPS, . . .

Internet shopping agent

- An agent could also be a software agent
  - Performance measure?
    - Price, quality, appropriateness, efficiency
  - Environment?
    - Current and future WWW sites, vendors, shippers
  - Actuators?
    - Display to the user, follow URL, fill in a form
  - Sensors?
    - HTML pages (text, graphics, scripts)

Properties of Task Environments

- Task environments vary along several significant dimensions. They can be
  - fully or partially observable
    - If the sensors detect all aspects that are relevant
    - With fully observable environments no need to maintain any internal state for keeping track of the world
    - Partially observable because of noisy and accurate sensors or missing parts of the state from sensors
    - With no sensors, the environment is unobservable
  - single-agent or multiagent
  - competitive or cooperative
    - In chess, agent A tries to maximize its performance measure which minimizes the opponent agent B's performance measure
    - In a taxi driving environment, avoiding collisions maximize the performance measures of all agents
  - deterministic or nondeterministic
    - If the next state of the environment is completely determined by the current state and the action executed by the agent

Properties of Task Environments

- Task environments vary along several significant dimensions. They can be
  - episodic or sequential
    - The agent's experience is divided into atomic episodes where in each episode the agent receives a percept and performs a single action
      - The next episode does not depend on the actions taken in the previous episodes
    - On the other hand, the current decision could affect all future decisions
  - static or dynamic
    - If the environment changes while an agent is acting, then the environment is dynamic otherwise it is static
  - discrete or continuous
    - The discrete/continuous distinction applies to the state of the environment, to the way time is handled, and to the percepts and actions of the agent
  - known or unknown
    - The distinction refers not to the environment itself but to the agent's state of knowledge about the "laws of physics" of the environment
    - In a known environment, the outcomes (or outcome probabilities if the environment is nondeterministic) for all actions are given
    - If the environment is unknown, the agent will have to learn how it works in order to make good decisions
- The hardest case is partially observable, multiagent, nondeterministic, sequential, dynamic, continuous, and unknown

Environments types

	Solitaire	Backgammon	Internet shopping	Taxi
Observable	Yes	Yes	No	No
Deterministic	Yes	No	Partly	No
Episodic	No	No	No	No
Static	Yes	Semi	Semi	No
Discrete	Yes	Yes	Yes	No
Single-agent	Yes	No	Yes(except auctions)	No

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

The Structure of Agents

- The job of AI is to design an agent program that implements the agent function
  - the mapping from percepts to actions
- We assume this program will run on some sort of computing device with physical sensors and actuators
  - It is called the agent architecture
- Thus, *agent = architecture+program*
- the program must be one that is appropriate for the architecture
- In general, the architecture
  - makes the percepts from the sensors available to the program
  - runs the program, and
  - feeds the program's action choices to the actuators as they are generated

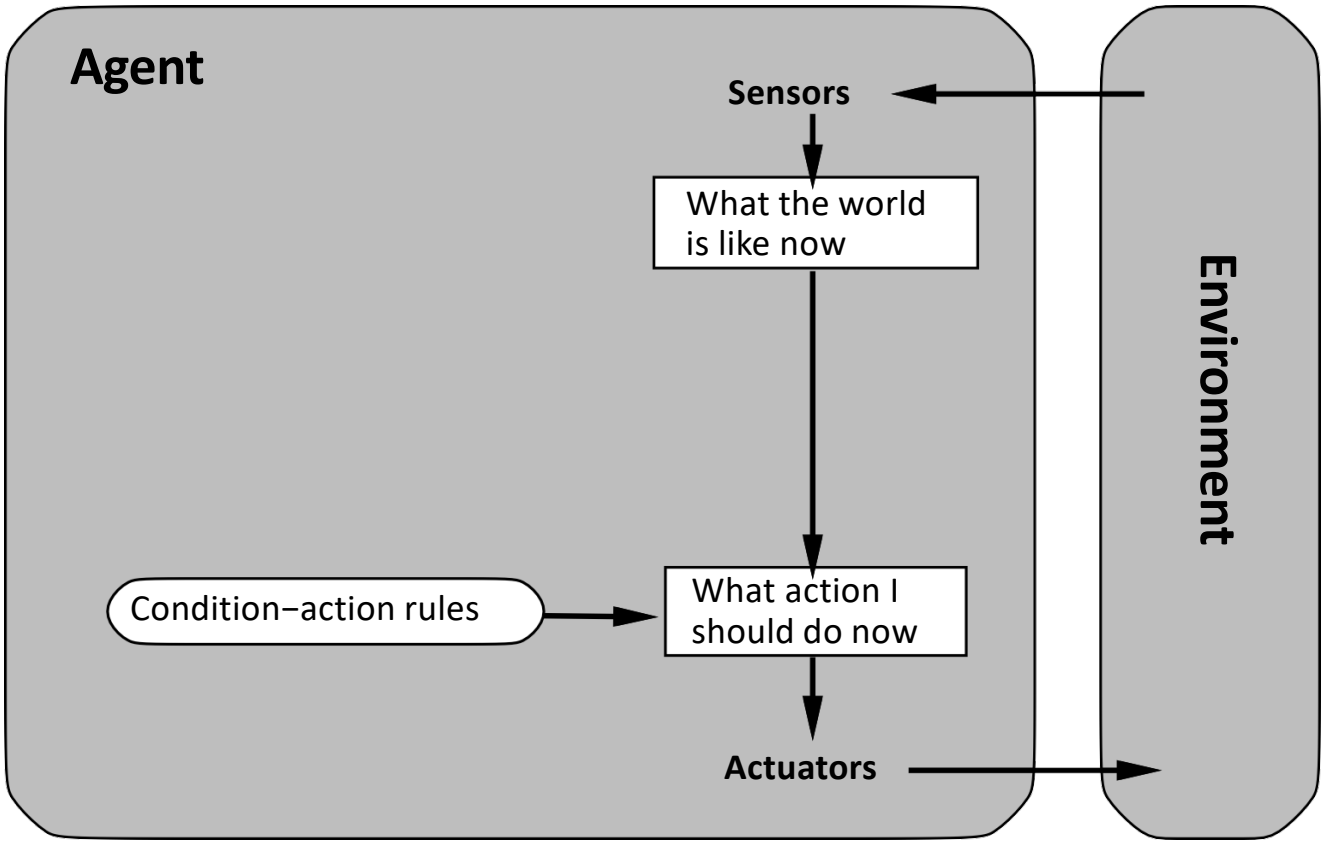
Agent types

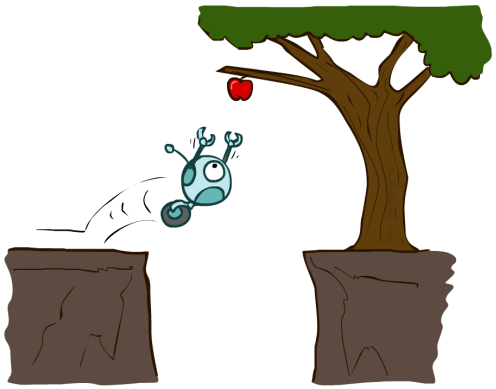
- Four basic types in order of increasing generality
  - Simple reflex agents
  - Reflex agents with state
  - Goal-based agents
  - Utility-based agents
- Each kind of agent program combines components in particular ways to generate actions
- All these can be turned into learning agents

Reflex Agents

- It operates by simply reacting to environmental stimuli in a pre-programmed way
- Reflex agents are the simplest type of agents, and they are not capable of learning or making decisions based on their beliefs and desires
- Reflex agents operate using a set of rules called a condition-action rule, which maps environmental conditions to actions
- When a reflex agent receives input from its sensors that satisfies a condition in its set of rules, it performs the corresponding action without any further decision-making

Simple reflex agents





Example

```
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
```

Example: A Thermostat Agent

- A simple reflex agent designed to control the temperature in a room
- Set of rules:
  - If the temperature is too high, turn on the air conditioning
  - If the temperature is too low, turn on the heater
- It simply reacts to the current temperature reading and performs the appropriate action based on a pre-programmed set of rules
- Reflex agents are limited in their capabilities, but they can be useful for performing simple tasks that require a quick and automatic response
  - controlling a thermostat,
  - turning on a light when someone enters a room or detecting obstacles in a robot's path
- However, more complex tasks require agents with greater decision-making and learning capabilities, such as model-based agents, goal-based agents, or learning agents

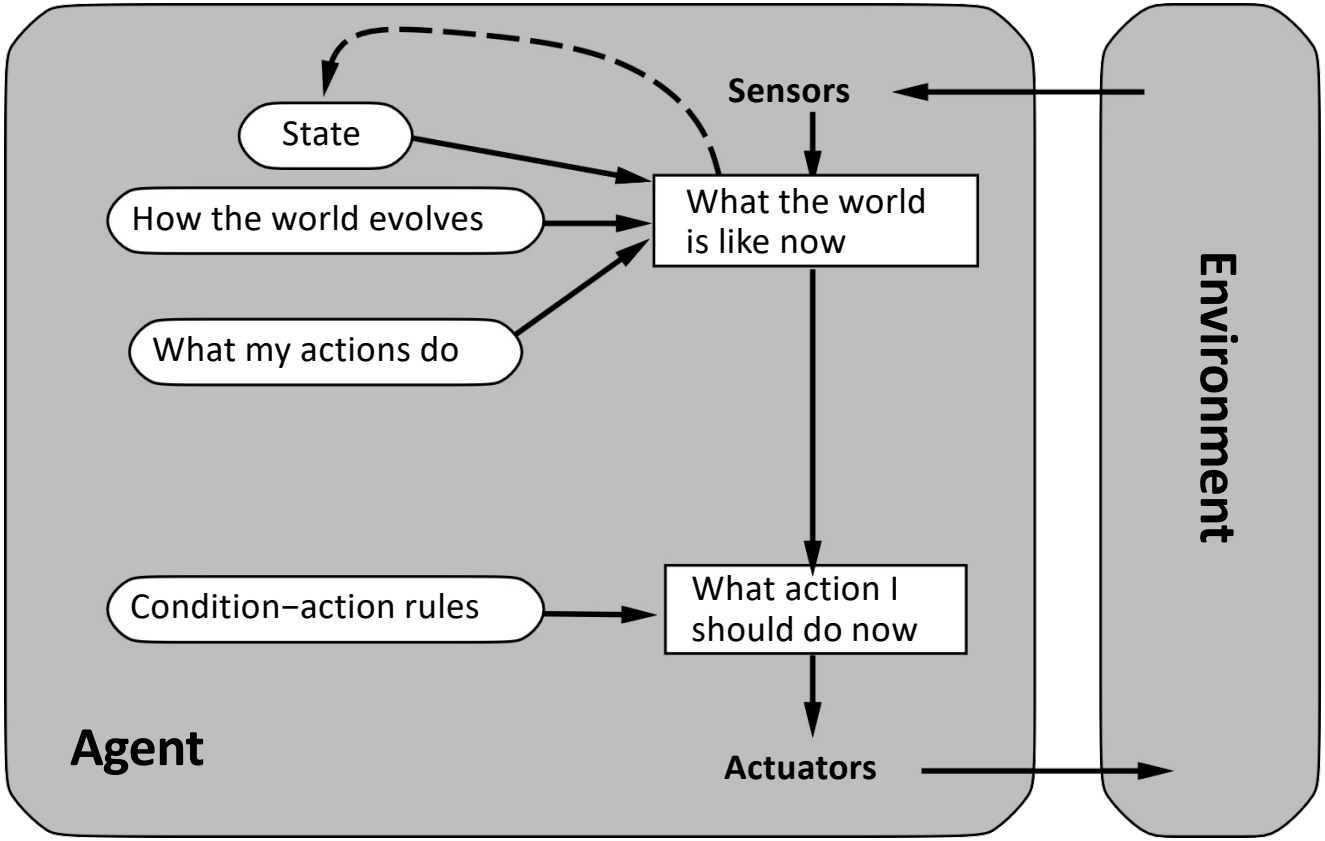
Reflex Agents with State

- A reflex agent with state is a type of AI agent that uses a set of predefined rules or reflexes, as well as an internal representation of the current state of the environment, to take actions
- This representation, called the "state", is used to make more informed decisions by considering the current and past states of the environment
- Example
  - A reflex agent with state might react to the presence of an obstacle in its path by considering the distance to the obstacle and the presence of other obstacles nearby to make a more informed decision about which direction to turn
  - To handle partial observability the agent should keep track of the part of the world it can't see, that is, the agent should maintain an internal state that depends on the percept history and reflects some of the unobserved aspects of the current state
    - For the braking problem, the internal state is just the previous frame from the camera, allowing the agent to detect when two red lights at the edge of the vehicle go on or off simultaneously

Model-Based Reflex Agents

- To update the internal state information during the time, the agent program must encode two kinds of knowledge
  1. some information about how the world that changes over time
     1. the effects of the agent's actions and
     2. how the world evolves independently of the agent
  - For example, when the agent turns the steering wheel clockwise, the car turns to the right, and when it's raining the car's cameras can get wet
  - This knowledge about "how the world works" is called a transition model of the world
  2. some information about how the state of the world is reflected in the agent's percepts
  - For example, when the car in front initiates braking, one or more illuminated red regions appear in the forward-facing camera image
  - This kind of knowledge is called a sensor model
- Together, the transition model and the sensor model allow an agent to keep track of the state of the world to the extent possible given the limitations of the agent's sensors
- An agent that uses such models is called a model-based agent

Model-based Reflex agents with state



Example

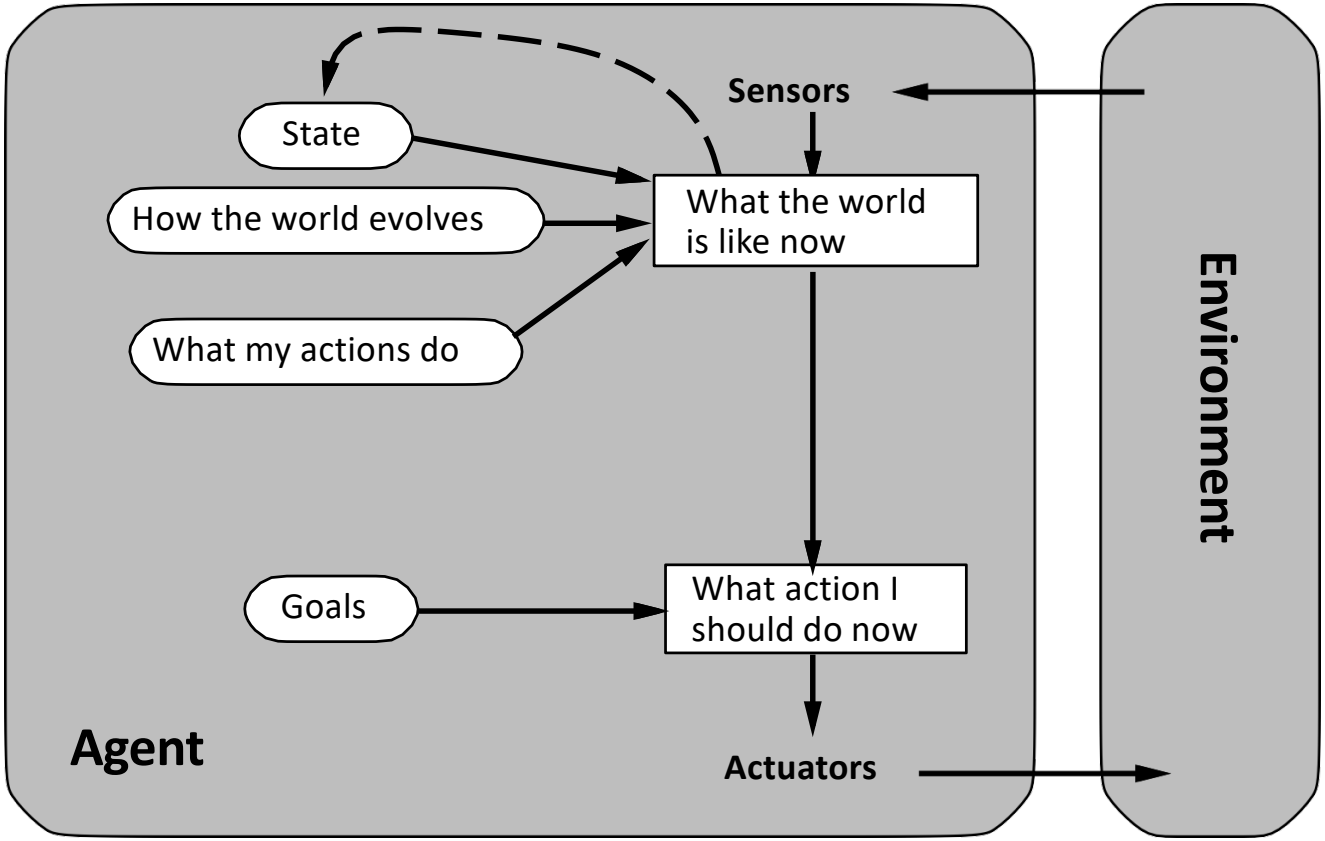
```
function MODEL-BASED-REFLEX-AGENT(percept) returns an action
  persistent: state, the agent's current conception of the world state
              transition_model, a description of how the next state depends on
                    the current state and action
              sensor_model, a description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

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

Figure 2.12 A model-based reflex agent. It 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.

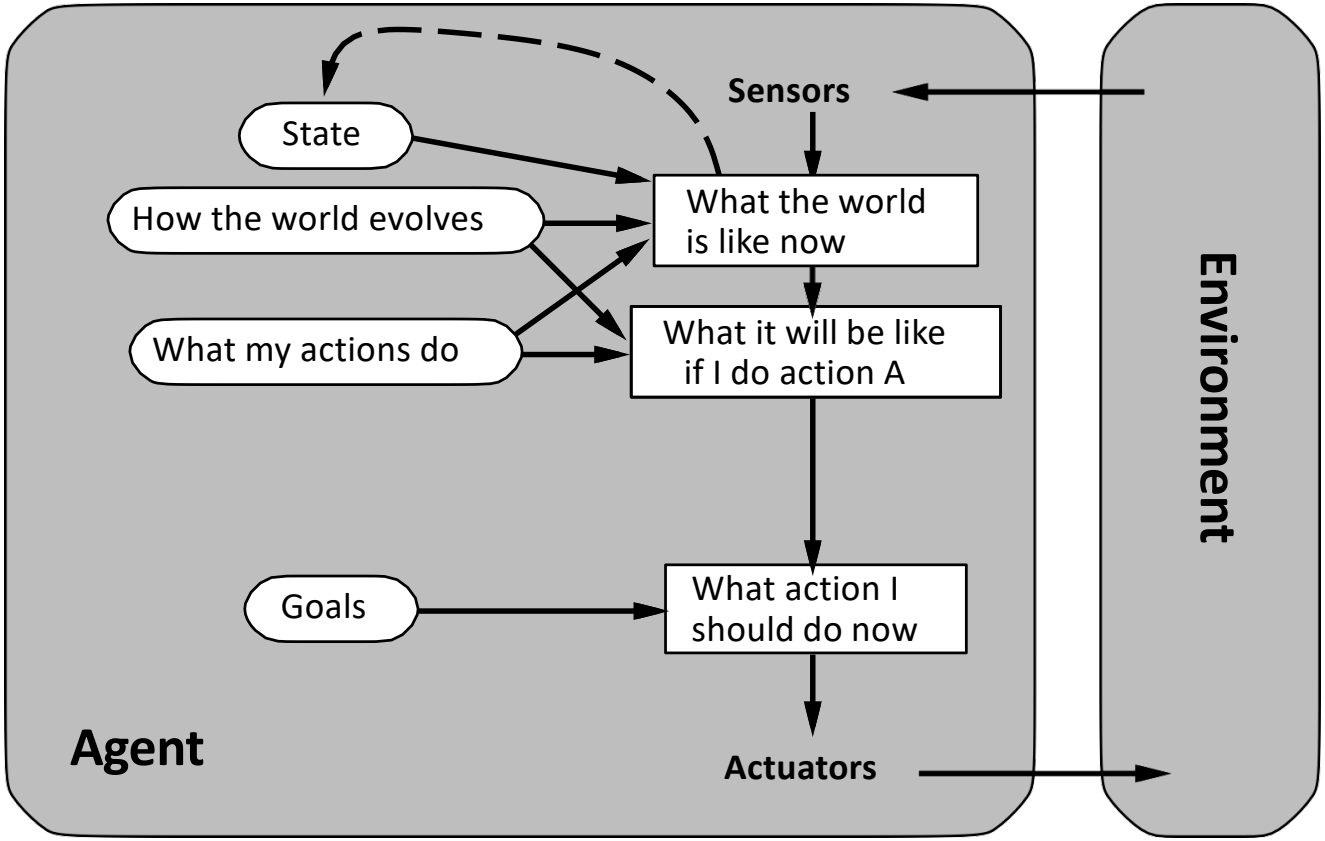
Goal-based

- The agent needs the current state description and a goal, information that describes situations that are desirable
- The agent program can combine this with the model to choose the actions that achieve the goal
  - Search is a subfield of AI devoted to finding action sequences that achieve the agent's goals
- This kind of decision-making is different from the condition-action rules
  - it involves consideration of the future
  - In reflex agents, this information is not explicitly represented because the built-in rules map directly from perceptions to actions

Model-based agent



Goal-based agent



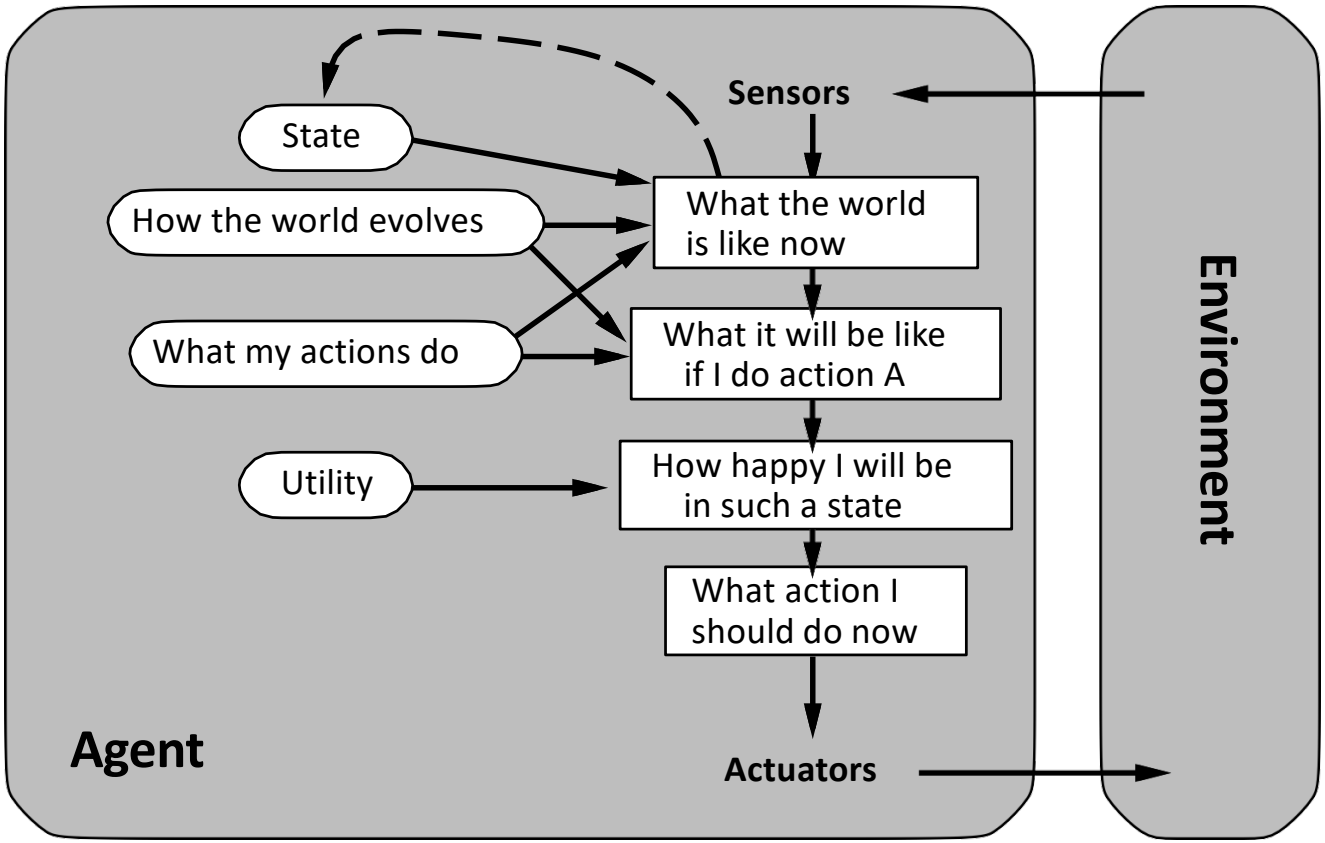
Goal-based vs Reflex Agents

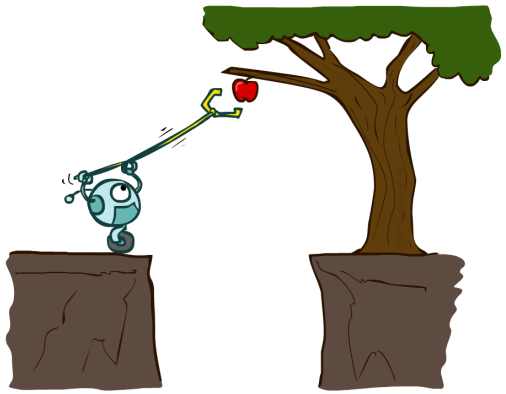
- A goal-based agent might seem less efficient, but it is more flexible due to its explicit representation of knowledge, which allows for easier modifications
- Example
  - Changing the behavior of a goal-based agent to reach a different destination is as simple as specifying that new destination as the goal
  - In contrast, the rules of a reflex agent that determine when to turn or go straight only apply to a specific destination and must be entirely replaced to navigate to a different location

Utility-based Agent

- Goals alone are not enough to generate high-quality behavior in most environments
- A more general performance measure should allow comparing different states of the world according to the utility they make the agent gain
- A performance measure assigns a score to any given sequence of environment states
  - An agent's utility function is an internalization of the performance measure
  - if the internal utility function and the external performance measure agree, an agent that chooses actions to maximize its utility will be rational according to the external performance measure

Utility-based agent





Utility-based vs Goal-based Agents

- A utility-based agent has many advantages in terms of flexibility and learning letting to take a rational decision when goals are inadequate
  - when there are conflicting goals (e.g., speed and safety) and only some of which can be achieved
    - A utility function enables specifying a suitable tradeoff
  - when there are multiple goals for the agent to pursue, none of which can be achieved with certainty
    - The utility provides a way to weigh the likelihood of success against the importance of the goals

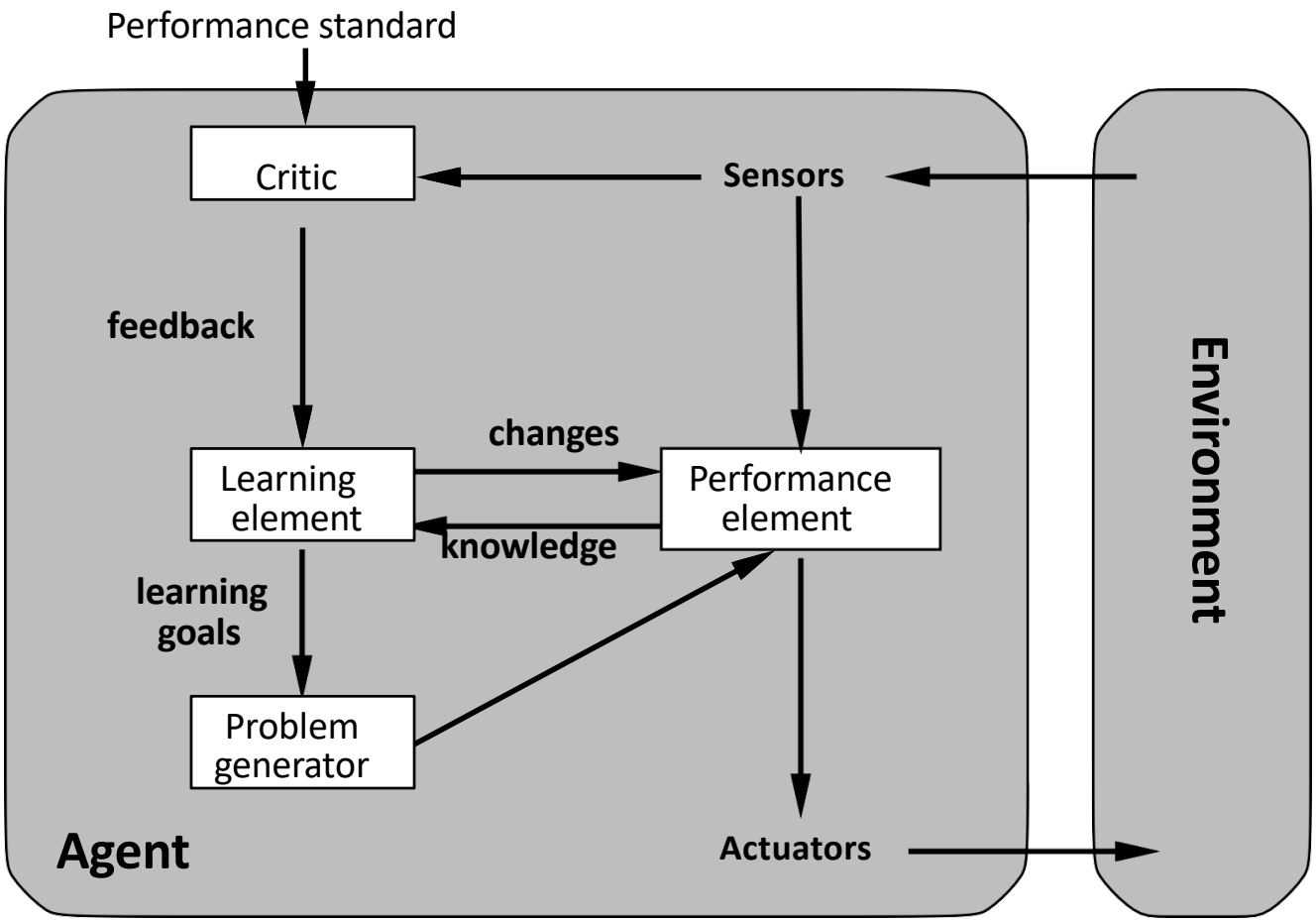
Expected Utility

- Since many real-world domains are partially observable and nondeterministic, the decision-making process operates under uncertainty
- A rational utility-based agent chooses the action that maximizes the expected utility of the action results
  - In other words, the utility expected to derive, on average, given the probability and utility of each outcome
- A utility-based agent must model and keep track of its environment, tasks that involve perception, representation, reasoning, and learning

Learning-based Agents

- If an agent needs to act in an unknown environment, learning is the key to a possible success
  - It allows an agent to become more competent than its initial knowledge alone
- A learning agent has four conceptual elements
  - Learning element
    - Responsible for making improvements
  - Performance element
    - Responsible for selecting external actions
  - Critic
    - How the agent is doing and how the performance element should be modified to do better
  - Problem generator
    - Suggests actions leading to new experiences

Learning agent



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's sequence
- A perfectly 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