

# Artificial Intelligence

## AI 2002



### Chapter # 02

**Dr. Hafeez Ur Rehman**

**(Email: [hafeez.urrehman@nu.edu.pk](mailto:hafeez.urrehman@nu.edu.pk))**

# Rational Agents (Chapter 2)



**Act Rationally Approach**

# Today's Outline

- ❑ Agents and Environments
- ❑ Autonomy, Rationality & Omniscience
- ❑ PEAS (Performance measure, Environment, Actuators, Sensors)
- ❑ Environment types
- ❑ Agent types

# Agent Definition (1)

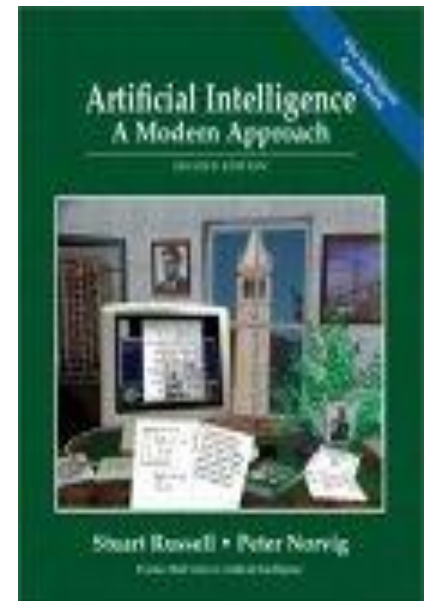
- An agent is an entity which is:
  1. *Situated in some environment.*
  2. *Autonomous*, in the sense that it can act without direct intervention from humans or other software processes, and controls over its own actions and internal state.
  3. *Flexible* which means:
    - *Responsive (reactive)*: agents should perceive their environment and respond to changes that occur in it;
    - *Proactive*: agents should not simply act in response to their environment, they should be able to **exhibit opportunistic, goal-directed behavior** and take the initiative when appropriate;
    - *Social*: agents should be able to interact with humans or other artificial agents

*“A Roadmap of agent research and development”,  
N. Jennings, K. Sycara, M. Wooldridge (1998)*

# Agent Definition (2)

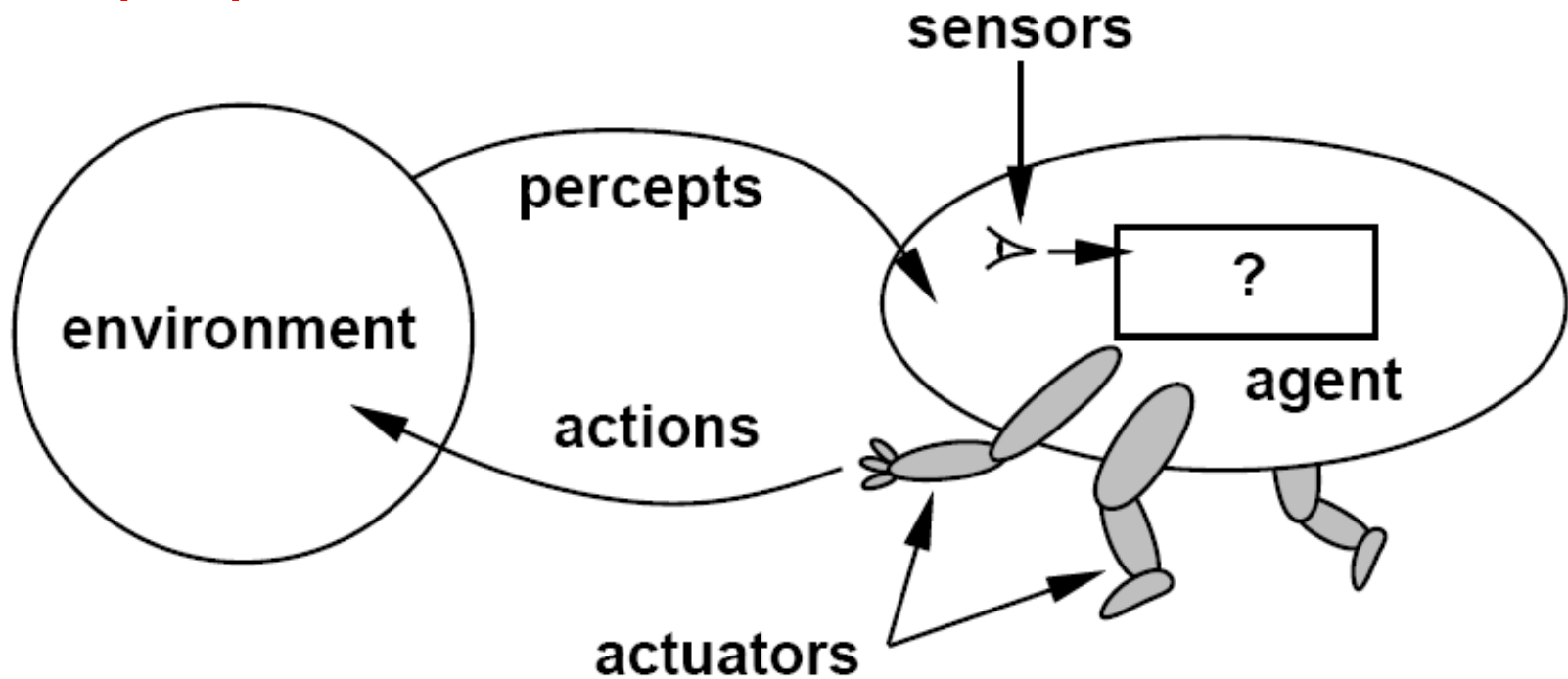
- *"An agent is anything that can be viewed as **perceiving** its **environment** through sensors and **acting** upon that **environment** through effectors/actuators."*

Russell & Norvig



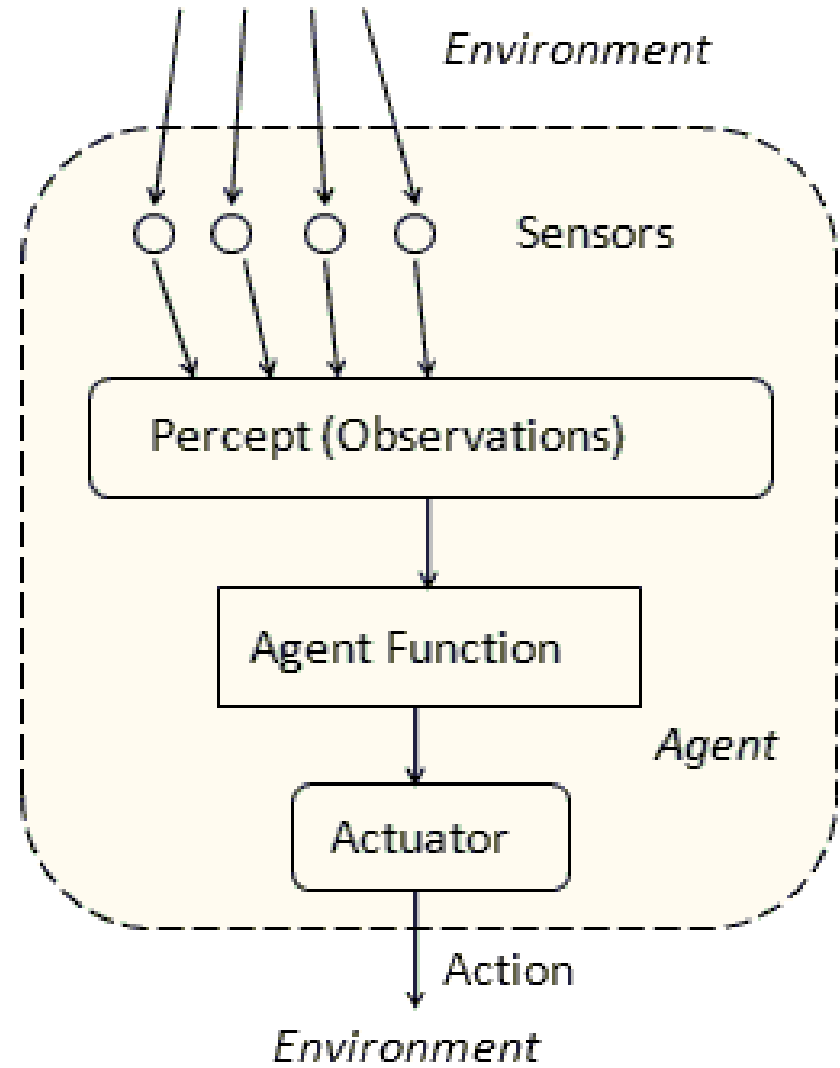
# Agent Visualization

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

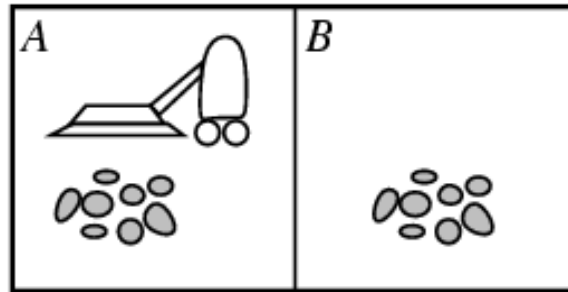


# Agent Function & Program

- The **agent function** maps from **percept histories** to actions:  
$$[f: \mathcal{P}^* \rightarrow \mathcal{A}]$$
- The **agent program** is an implementation of  $f$  that runs on the physical **architecture** to produce  $f$
- **agent = architecture + program**



# Example: Vacuum-cleaner world



- **Percepts:** location and contents, e.g., [A,Dirty]
- **Actions:** *Left, Right, Suck, NoOp*
- **Agent's function** → *look-up table*
  - *For many agents this is a very large table*

Percept sequence	Action
[A, Clean]	<i>Right</i>
[A, Dirty]	<i>Suck</i>
[B, Clean]	<i>Left</i>
[B, Dirty]	<i>Suck</i>
[A, Clean], [A, Clean]	<i>Right</i>
[A, Clean], [A, Dirty]	<i>Suck</i>
⋮	⋮



# Other Agent Examples...

- **Human agent:**
  - eyes, ears, and other organs for **sensors**;
  - hands, legs, mouth, and other body parts for **actuators**
- **Robotic agent:**
  - cameras and infrared range finders for **sensors**
  - various motors for **actuators**

# Rational agents

- **Rationality:** What is rational at any given time depends on four things i.e.
  1. **Performance measure** that defines success
  2. Agents **prior knowledge** of **environment**
  3. **Actions** that agent can perform
  4. 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.

# Autonomy in Agents

The **autonomy** of an agent is the extent to which its behaviour is determined by **its own experience**, rather than knowledge of the designer.

- Extremes
  - No autonomy – ignores environment/data
  - Complete autonomy – must act randomly/no program
- Example: baby learning to crawl
- Ideal: design agents to have some autonomy
  - Possibly become more autonomous with experience

# Rationality vs Omniscience

- Rational is different from **omniscience**
  - Percepts may not supply all relevant information
  - E.g., in card game, don't know cards of others.
- Rational is different from being perfect
  - Rationality maximizes **expected outcome**, while perfection maximizes **actual outcome**.
- To **design** a rational agent you need to specify its task environment.

# Specifying the Task Environments

- **Task Environment** are essentially the **problems** to which **rational agents** are the **solutions**.
- In designing an agent the first step must always be to specify the **task environment** as fully as possible a.k.a PEAS (Performance measure, Environment, Actuators, Sensors)
- **Example (Automated Taxi):** Consider, e.g., the task of designing an automated taxi driver agent:
  1. **Performance measure:** Safe, fast, legal, comfortable trip, maximize profits
  2. **Environment:** Roads, other traffic, pedestrians, customers
  3. **Actuators:** Steering wheel, accelerator, brake, signal, horn
  4. **Sensors:** Cameras, sonar, speedometer, GPS, odometer, engine sensors, keyboard

# PEAS - Example # 02

**Agent:** Part-Picking Robot

- 1. Performance measure:** Percentage of parts in correct bins
- 2. Environment:** Conveyor belt with parts, bins
- 3. Actuators:** Jointed arm and hand
- 4. Sensors:** Camera, joint angle sensors



# Environment types

- Fully observable vs. partially observable
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Single agent vs. multi-agent
- Known vs. unknown

# Fully observable vs. partially observable

- Do the agent's sensors give it access to the complete state of the environment?
  - For any given world state, are the values of all the variables known to the agent?



VS.





# Examples



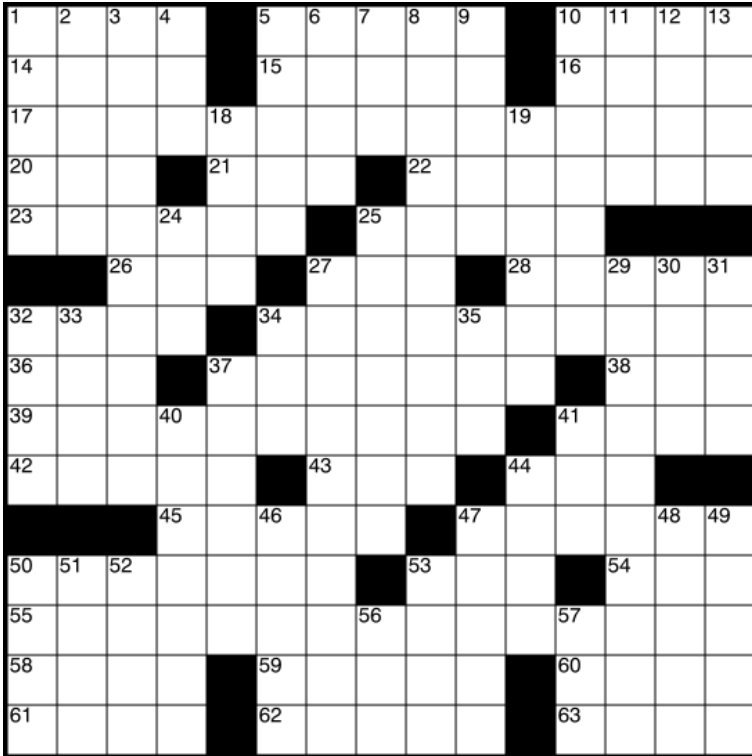
**Poker Game (Partially)**



**Backgammon (Fully)**

A task environment is effectively fully observable if the **sensors detect all aspects that are *relevant* to the choice of action**; relevance, in turn, depends on the performance measure.

# Examples



**Cross Word  
(Fully)**



**Part Picking Robot  
(Partially)**

# Deterministic vs. stochastic

- Is the next state of the environment completely determined by the current state and the agent's action?
  - Is the transition model deterministic (unique successor state given current state and action) or stochastic (distribution over successor states given current state and action)?
  - **Strategic:** the environment is deterministic except for the actions of other agents



VS.

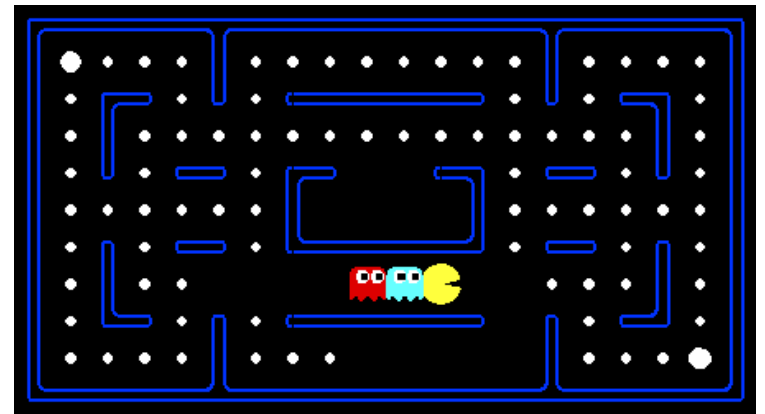


# Episodic vs. sequential

- Is the agent's experience divided into unconnected episodes, or is it a coherent sequence of observations and actions?
  - Does each problem instance involve just one action or a series of actions that change the world state according to the transition model?



VS.

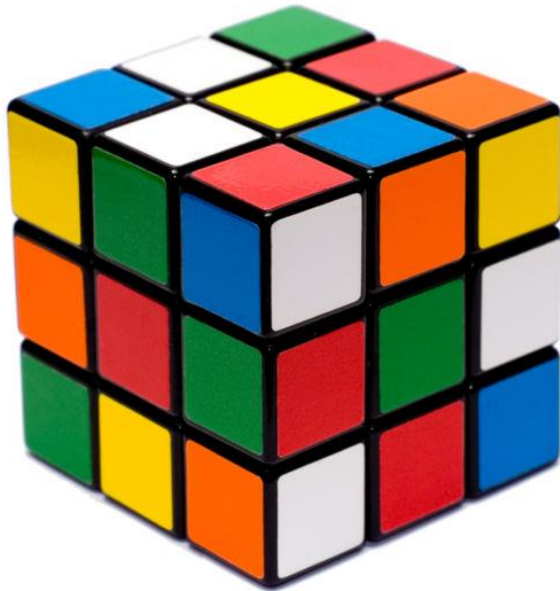


## Episodic (vs. sequential):

- Is the choice of current action
  - Dependent on previous actions?
  - If not, then the environment is episodic
- In non-episodic environments:
  - Agent has to plan ahead:
    - Current choice will affect future actions

# Static vs. dynamic

- Is the world changing, while the agent is thinking?
  - **Semidynamic:** the environment does not change with the passage of time, but the agent's performance score does



VS.





# Discrete vs. continuous

- Does the environment provide a fixed number of distinct percepts, actions, and environment states?
  - Are the values of the state variables discrete or continuous?
  - Percepts and actions of the agent are discrete or continuous?
  - Time can also evolve in a discrete or continuous fashion



VS.

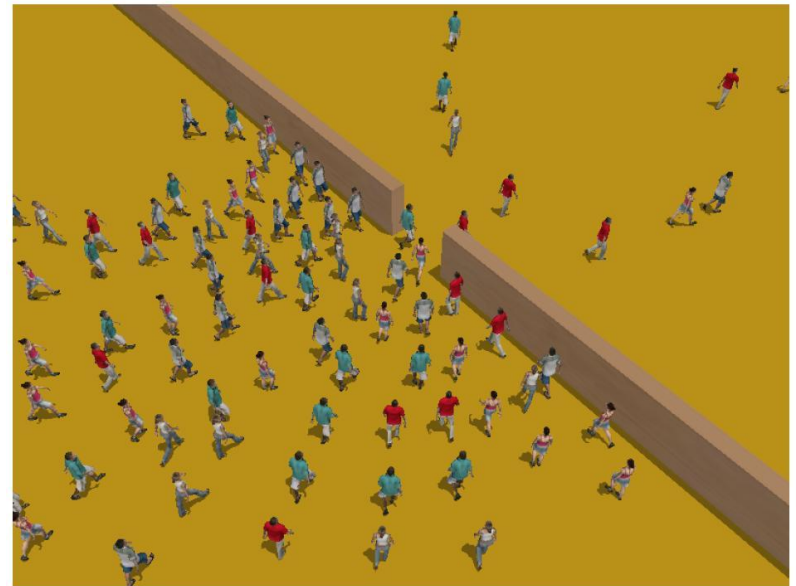


# Single-agent vs. multiagent

- Is an agent operating by itself in the environment?



vs.





# Known vs. unknown

- Are the rules of the environment (laws of physics, transition model and rewards associated with states etc.) known to the agent?
  - Strictly speaking, not a property of the environment, but of the agent's state of knowledge



Monopoly game

vs.



A new video game

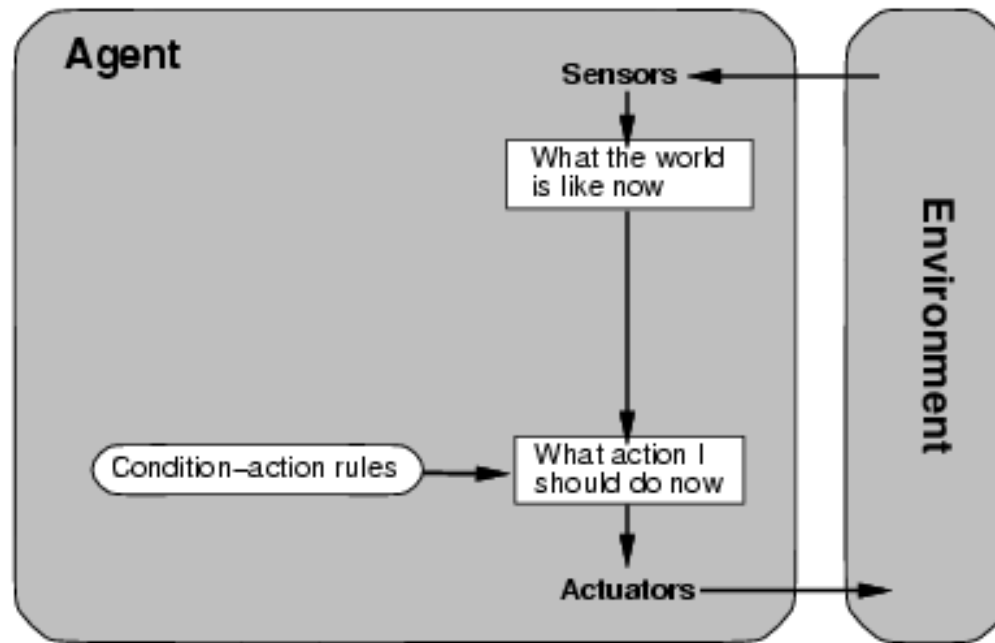
# Summary

	Observable	Deterministic Episodic		Static	Discrete	Agents
<b>Cross Word</b>	Fully	Deterministic	Sequential	Static	Discrete	Single
<b>Poker</b>	Partially	Stochastic	Sequential	Static	Discrete	Multi
<b>Backgammon</b>	Fully	Stochastic	Sequential	Static	Discrete	Multi
<b>Taxi driver</b>	Partially	Stochastic	Sequential	Dynamic	Conti	Multi
<b>Part picking robot</b>	Partially	Stochastic	Episodic	Dynamic	Conti	Single
<b>Image analysis</b>	Fully	Deterministic	Episodic	Semi	Conti	Single

# Agent Types

- Five basic types in order of increasing generality:
  - 1. Simple reflex agents**
  - 2. Reflex agents with state/model**
  - 3. Goal-based agents**
  - 4. Utility-based agents**
  - 5. Learning based Agents**

# Simple Reflex Agents



```
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
```

# Simple Reflex Agents

- Simple but very limited intelligence.
- **Action does not depend on percept history, only on current percept.**
- Therefore no memory requirements.
- **PROBLEM: Infinite loops if environment partially observable**
  - Suppose vacuum cleaner does not observe location. What do you do given location = clean? Left of A or right of B -> infinite loop.
  - Agent will behave like fly buzzing around window or light.
  - **Possible Solution:** Randomize action (flip a coin to decide action).
- For complex problems e.g., Chess
  - Lookup table (not a good idea in general)
    - $35^{100}$  entries required for the entire game

# States: Beyond Reflexes

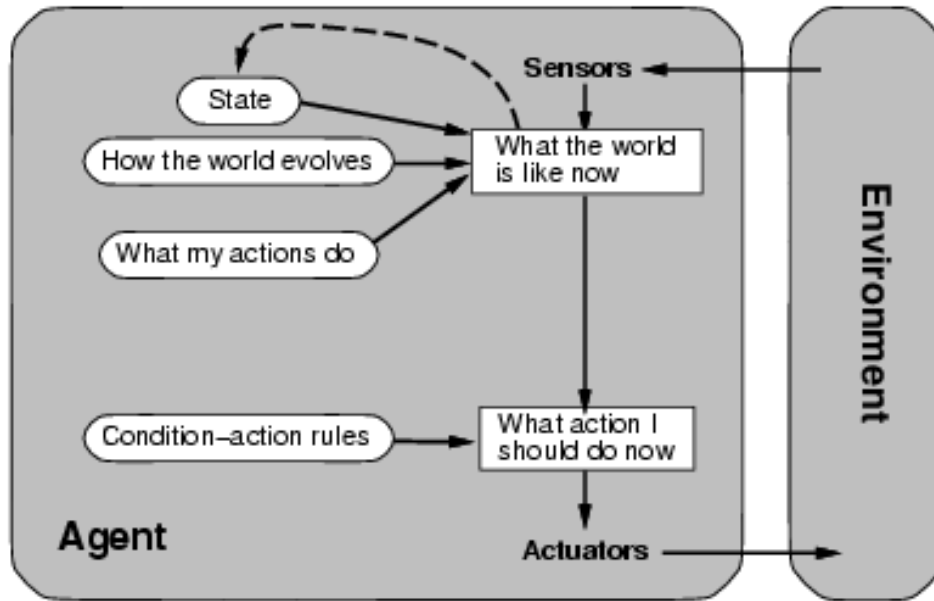
- Recall the **agent function** that maps from **percept histories** to actions:

$$[f: \mathcal{P}^* \rightarrow \mathcal{A}]$$

- An agent program can implement an agent function by maintaining an **internal state**.
- The internal state can contain information about the state of the external environment.
- The state depends on the history of percepts and on the history of actions taken:

$$[f: \mathcal{P}^*, \mathcal{A}^* \rightarrow \mathcal{S} \rightarrow \mathcal{A}] \text{ where } \mathcal{S} \text{ is the set of states.}$$

# Model-based reflex agents



1. Know how world evolves
  - Overtaking car gets closer from behind
2. How agents actions affect the world
  - Wheel turned clockwise takes you right

```
function REFLEX-AGENT-WITH-STATE(percept) returns action
  static: state, a description of the current world state
         rules, a set of condition-action rules

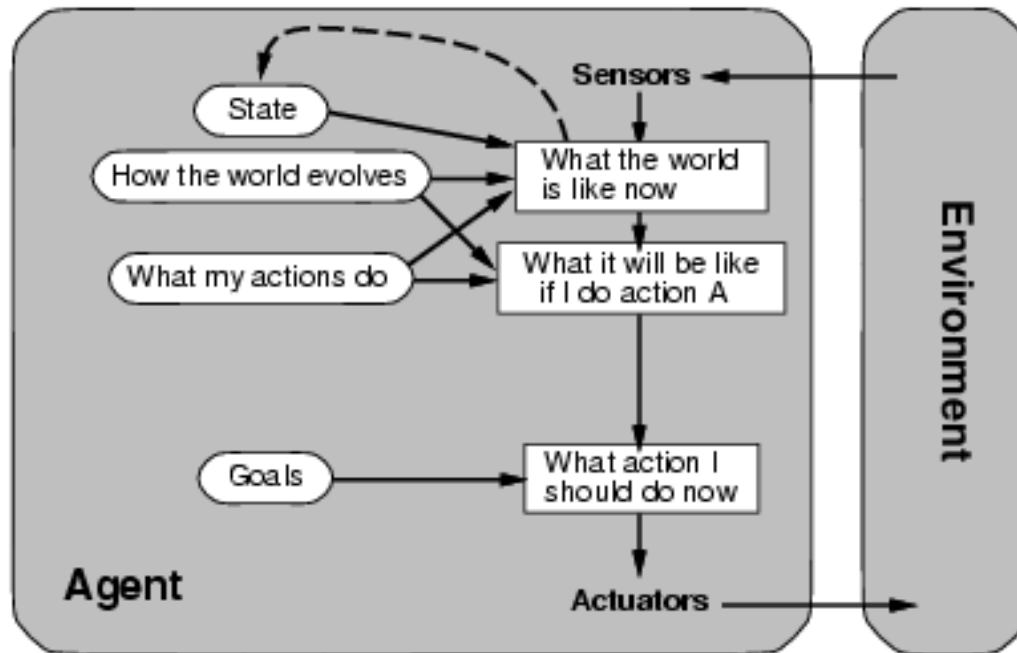
  state ← UPDATE-STATE(state, percept)
  rule ← RULE-MATCH(state, rules)
  action ← RULE-ACTION[rule]
  state ← UPDATE-STATE(state, action)
  return action
```

# Goal-based agents

- Knowing state and environment? Enough?
  - Taxi can go left, right, straight
- Have a goal
  - A destination to get to
- Uses knowledge about a goal to guide its actions
  - E.g., Search, planning
- Modified brake behaviour in case of rain



# Goal-based agents

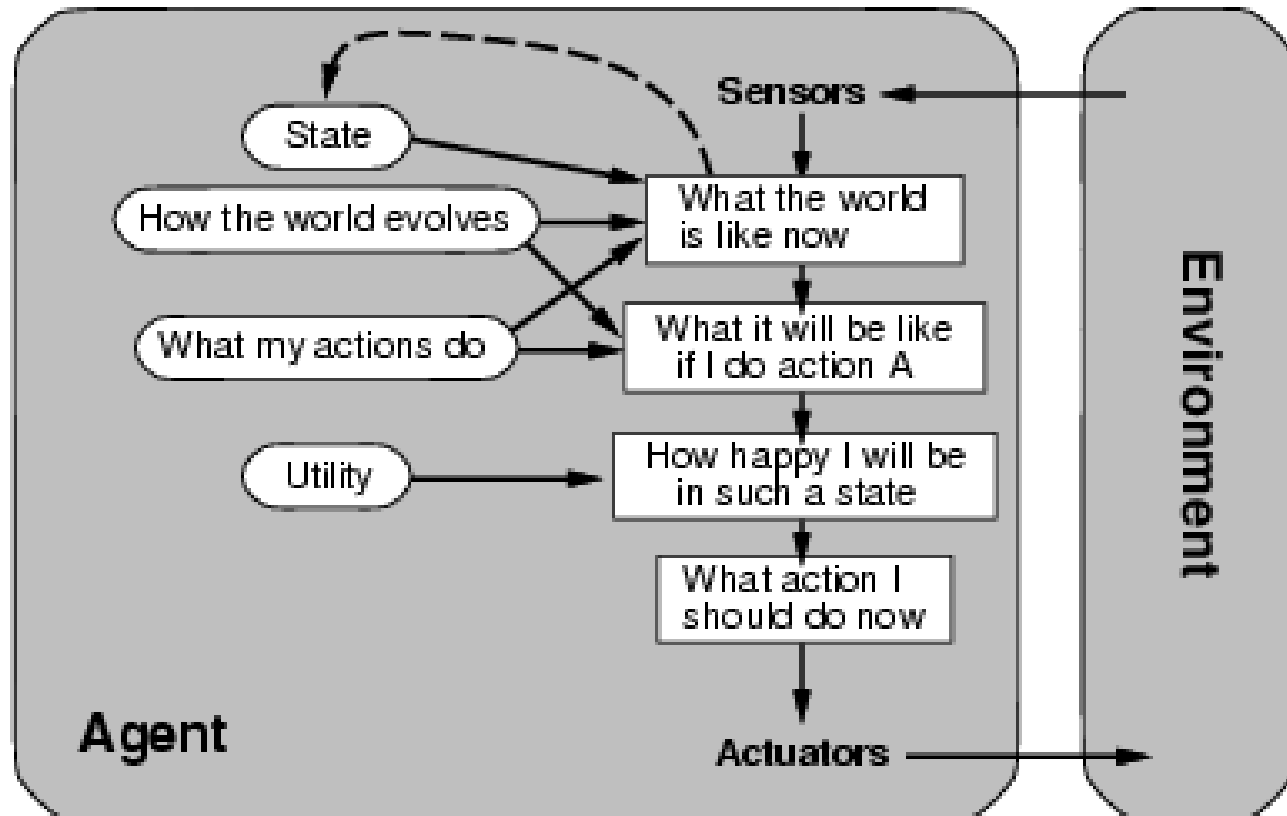


- Reflex agent breaks when it sees brake lights. Goal based agent reasons
  - Brake light -> car in front is stopping -> I should stop -> I should use brake

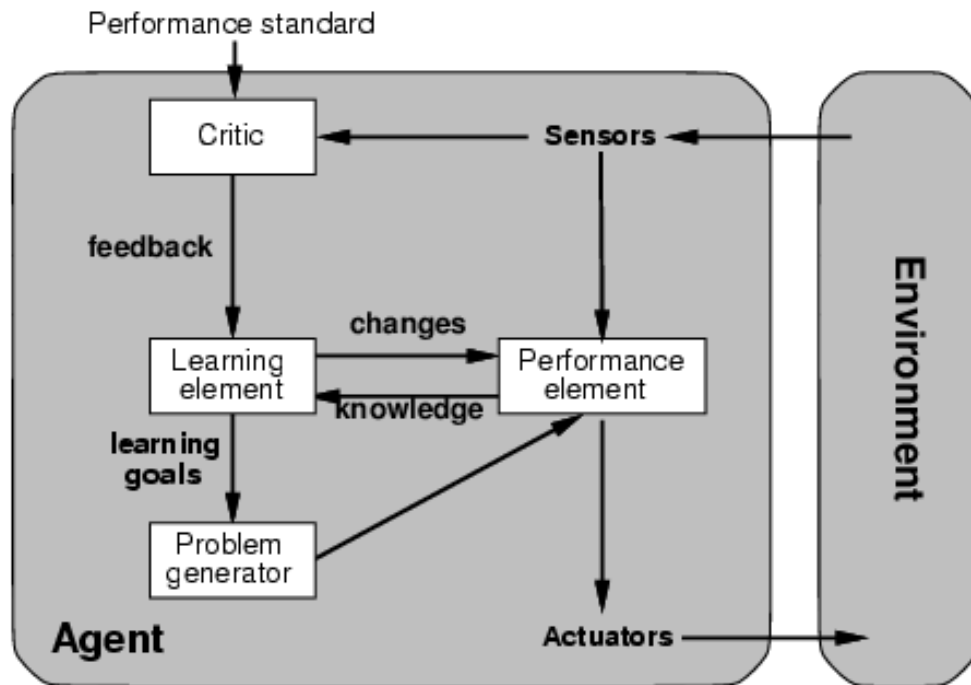
# Utility-based agents

- Goals are not always enough
  - Many action sequences get taxi to destination
  - Consider other things. **How fast, how safe.....**
- A **Utility function** maps a state onto a real number which describes the associated degree of “**happiness**”, “**goodness**”, “**success**”.
- Where does the utility measure come from?
  - Economics: money.
  - Biology: number of offspring.
  - Your life?

# Utility-based agents

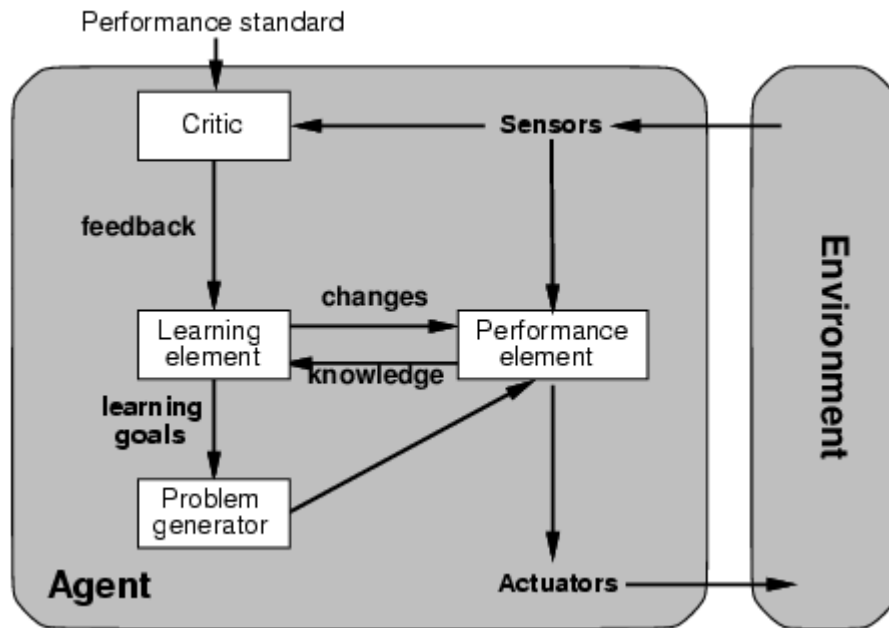


# Learning agents



- **Performance element** is what was previously the **whole agent**
  - Input sensor
  - Output action
- **Learning element**
  - Modifies performance element.

# Learning agents



- Critic: how the agent is doing
  - Input: checkmate?
  - Fixed
- Problem generator
  - Tries to solve the problem differently instead of optimizing.
  - Suggests **exploring** new actions -> new problems.

# Learning agents(Taxi driver)

- Performance element
  - How it currently drives
- Taxi driver Makes quick left turn across 3 lanes
  - **Critics observe shocking language by passenger** and other drivers and informs bad action
  - Learning element tries to **modify performance elements** for future
  - Problem generator **suggests** experiment out something called Brakes on different Road conditions
- Exploration vs. Exploitation
  - Learning experience can be costly in the short run
  - shocking language from other drivers
  - Less tip
  - Fewer passengers

End of Lecture