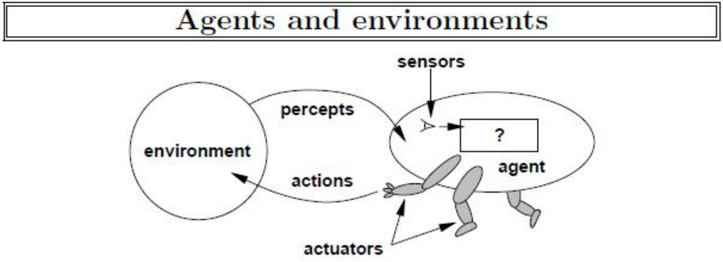
Introduction to Artificial Intelligence

Dr. Tapas Kumar Mishra

Assistant Professor
Department of Computer Science and Engineering
SRM University
Amaravati-522502, Andhra Pradesh, India
Email: tapaskumar.m [at] srmap [dot] edu [dot] in

Agents and Environments



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

Agent and Sensors

Agent: Human
Sensors: Eyes, Ears and other Organs
Effectors: Hands, Legs, Mouth and other body parts
Agent: Robotic
Sensors: Cameras and Infrared Range Finders
Effectors: Motors

- A **Rational Agent** is one that does the right thing. We will say that the **right action** is the one that will cause the agent to be **most successful**.

Deciding how and when to evaluate the agent's success

how

Consider the case of an agent that is supposed to **vacuum a dirty floor**. A performance measure would be the **amount of dirt cleaned up in a single eight-hour shift**. A more sophisticated performance measure would factor in the **amount of electricity consumed and the amount of noise generated** as well. A third performance measure might give **highest marks** to an agent that not only **cleans the floor** quietly and efficiently, but also finds time to go **windsurfing** at the weekend.

---

when

If we measured **how much dirt** the agent had cleaned up in the first hour of the day, we would be **rewarding** those agents that **start fast** (even if they do little or no work later on) and **punishing** those that **work consistently**.

Agents and Environments

- An **omniscient agent** knows the **actual outcome** of its actions, and can act accordingly; but omniscience is **impossible in reality**.
- **Example: Crossing the road with no traffic.** If I face an **accident** (due to some reason), then "**Was I irrational to cross the street**"? Crossing the street was rational because most of the time the crossing would be successful.
- What is rational at any given time depends on **four** things:
  - The performance measure that defines **degree of success**.
  - Everything that the agent has perceived so far. We will call this **complete perceptual history** the **percept sequence**.
  - What the agent knows about the **environment**?
  - The **actions** that the agent can perform.

Ideal Rational Agent

For each possible percept sequence, an **ideal rational agent** should do whatever **action** is expected to **maximize** its **performance measure**, on the basis of the **evidence** provided by the **percept sequence** and whatever **built-in knowledge** the agent has.

Ideal Mapping

Mapping from **percept sequences** to **actions**
Specifying which action an agent ought to take in response to any given percept sequence provides a design for an ideal agent.

Percept x	Action z
1.0	1.000000000000000
1.1	1.048808848170152
1.2	1.095445115010332
1.3	1.140175425099138
1.4	1.183215956619923
1.5	1.224744871391589
1.6	1.264911064067352
1.7	1.303840481040530
1.8	1.341640786499874
1.9	1.378404875209022
⋮	⋮

```
function SQRT(x)
  z ← 1.0                          /* initial guess */
  repeat until |z^2 - x| < 10^-15
              z ← z - (z^2 - x)/(2z)
  end
  return z
```

- $z \leftarrow \frac{z-(z^2-x)}{2z}$
- If the agent's actions are based completely on **built-in knowledge**, such that it AUTONOMY need pay no attention to its **percepts**, then we say that the agent lacks **autonomy**.
- **Example:** The clock's owner would be going to **Australia** at some particular date, then a mechanism could be built in to adjust the clock hands automatically by **six hours** at just the right time. Intelligence seems to belong to the **clock's designer** rather than to the **clock itself**.

Ideal Mapping

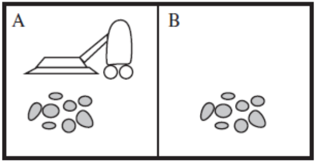


Figure 2.2 A vacuum-cleaner world with just two locations.

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
...	...

Table: Partial tabulation of a simple agent function

Structure of Intelligent Agents

- The relationship among agents, architectures, and programs can be summed up as follows.
- agent = architecture + program

Structure of Intelligent Agents

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry

Nature of Environments

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

Figure 2.6 Examples of task environments and their characteristics.

Why not just look up the answers?

Why not just look up the answers?

Let us start with the simplest possible way we can think of to write the agent program-**a lookup table**.
It is instructive to consider why this proposal is **doomed to failure**:

1. The table needed for something as simple as an agent that can only **play chess** would be about 35^{100} entries.
2. It would take quite a long time for the **designer** to **build the table**.
3. The agent has **no autonomy** at all, because the calculation of best actions is entirely built-in. So if the environment changed in **some unexpected way**, the agent would be **lost**.
4. Even if we gave the agent a **learning mechanism** as well, so that it could have a **degree of autonomy**, it would take forever to learn the right value for all the table entries.

Designing an Automated Taxi Driver

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

Figure 2.4 PEAS description of the task environment for an automated taxi.

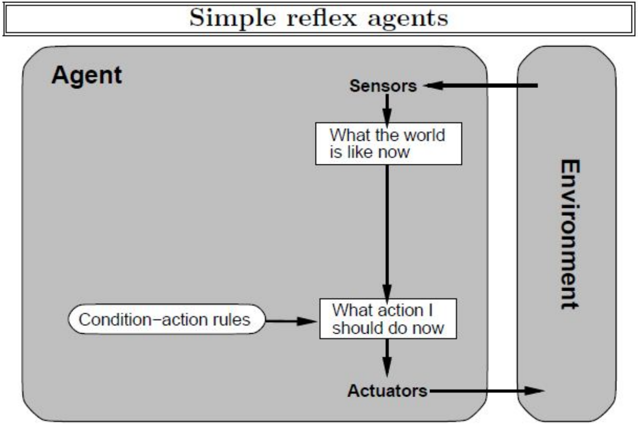
- Desirable qualities include getting to the **correct destination**; minimizing **fuel consumption** and wear and tear; minimizing the **trip time** and/or cost; minimizing **violations of traffic laws** and disturbances to other drivers; maximizing **safety and passenger comfort**; maximizing **profits**.
- Will it always be driving on the **right**, or might we want it to be flexible enough to drive on the left in case we want to operate taxis in **Britain** or **Japan**?

Types of Agent Program

1. Simple Reflex Agents
2. Model-Based Reflex Agents
3. Goal-Based Agent
4. Utility-Based Agents

- Learning Agents

Simple Reflex Agents



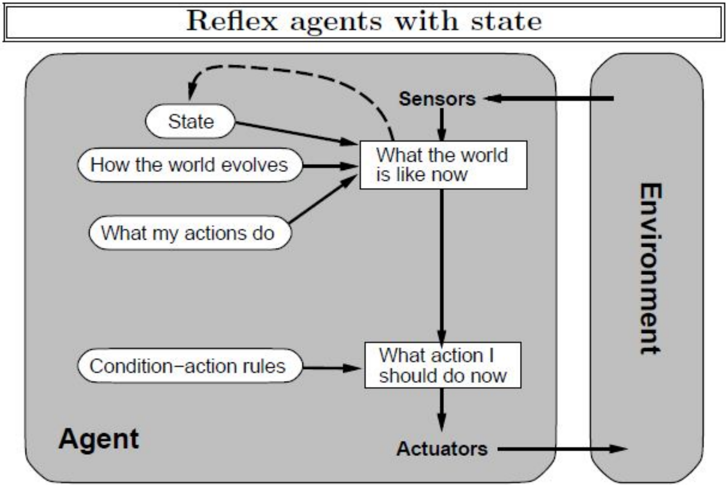
- if **car-in-front-is-braking** then **initiate-braking** //condition-action rules
- **Disadvantage:** It will work only if the **correct decision** can be made on the basis of the **current percept**.

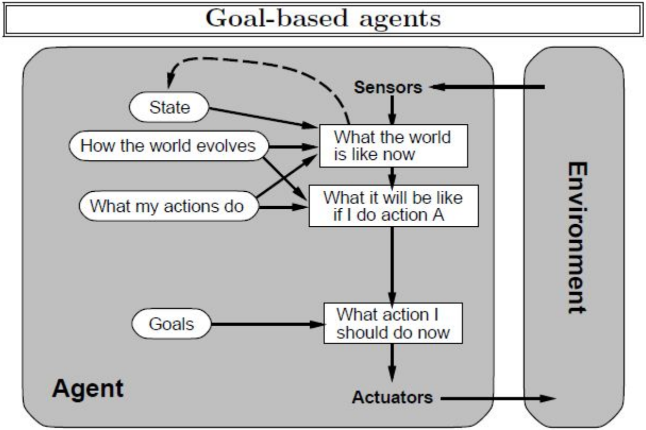
Simple Reflex Agents

- **Actuators** or **Effectors** available to an **automated taxi** will be more or less the same as those available to a **human driver**: control over the **engine** through the **accelerator** and control over **steering** and **braking**.

Simple Reflex Agents vs Reflex Agents with State

- Simple Reflex Agents: It works by finding a rule whose condition matches the **current situation** (as defined by the **percept**) and then doing the action associated with the rule.
- Reflex Agents with State: It works by finding a rule whose condition matches the **current situation** (as defined by the **percept** and the **stored internal state**) and then doing the action associated with the rule.


Reflex agents with state
Sensors
State
How the world evolves
What my actions do
What the world is like now
Condition−action rules
What action I should do now
Agent
Actuators
Environment

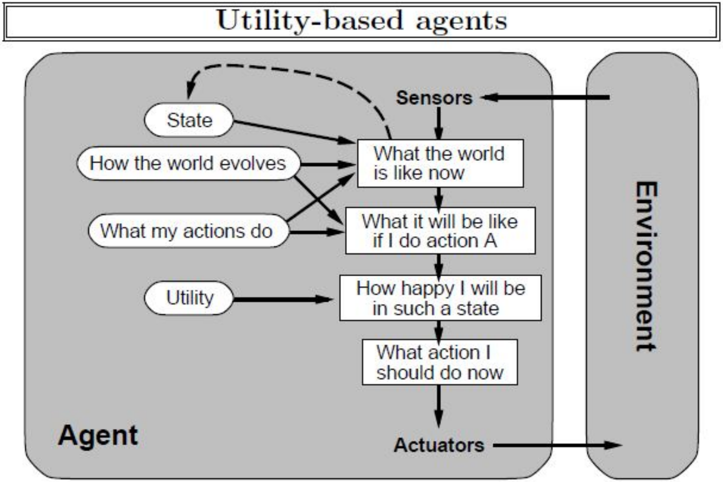


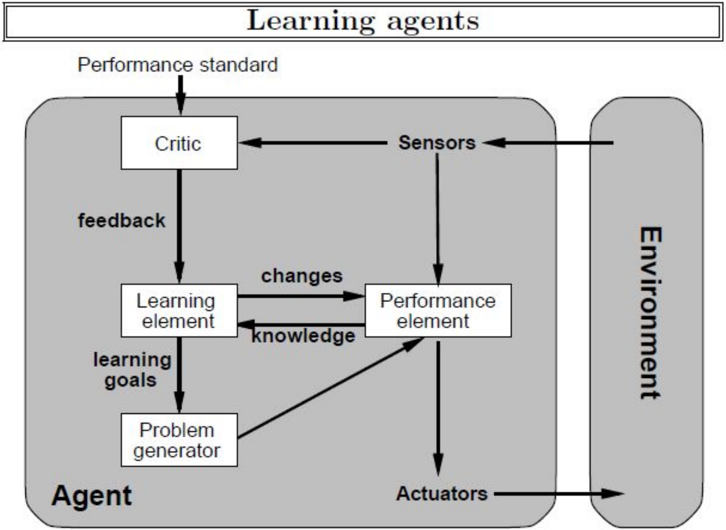
- **Example:** At a road junction, the taxi can turn **left**, **right**, or **go straight** on (**What it will be like if I do action A**).

A Model-Based, Utility-Based Agent

- There are many action sequences that will get the taxi to its **destination**, thereby achieving the **goal**, but some are **quicker**, **safer**, **more reliable** or **cheaper** than others.
- When there are **conflicting goals**, only some of which can be achieved (for example, **speed** and **safety**), the **utility function** specifies the appropriate **trade-off**.
- 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 up against the **importance of the goals**.

A Model-Based, Utility-Based Agent




Learning agents
Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Agent
Actuators
Environment

Thank You!