MODULE 1

# INTRODUCTION



## 1.1 Introduction - What is Artificial Intelligence (AI)?

Artificial intelligence is a wide-ranging branch of computer science concerned with building smart machines capable of performing tasks that typically require human intelligence.

Artificial intelligence allows machines to replicate the capabilities of the human mind.

From the development of self-driving cars to the development of smart assistants like Siri and Alexa, AI is a growing part of everyday life.

We can describe AI as the four shown in figure below

| **Thinking Humanly** | **Thinking Rationally** |
|---|---|
| "The exciting new effort to make computers think . . . *machines with minds*, in the full and literal sense." (Haugeland, 1985)<br>"[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . ." (Bellman, 1978) | "The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985)<br>"The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992) |
| **Acting Humanly** | **Acting Rationally** |
| "The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990)<br>"The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991) | "Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998)<br>"AI . . . is concerned with intelligent behavior in artifacts." (Nilsson, 1998) |

The definitions on top are concerned with thought processes and reasoning, whereas the ones on the bottom address behavior.

The definitions on the left measure success in terms of fidelity to human performance, whereas the ones on the right, measure against an ideal performance measure, which is called as rationality.

A system is rational if it does the "right thing," given what it knows.

**Acting humanly: The Turing Test approach**

The Turing Test is proposed by Alan Turing (1950)It was designed to provide a satisfactory operational definition of intelligence. Judge communicates with a human and machine over text through a channel. Both human and a machine try to act like a human.Judge tries to tell which is which.

A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or from a computer.

The computer would need to possess the following capabilities:

- Natural Language Processing - To enable it to communicate successfully in English
- Knowledge Representation - to store what it knows or hears
- Automated Reasoning - to use the stored information to answer questions and to draw new conclusions
- Machine Learning - to adapt to new circumstances and to detect and extrapolate patterns.

Turing's test deliberately avoided direct physical interaction between the interrogator and the computer.

The total Turing Test includes a video signal so that the interrogator can test the subject's perceptual abilities, as well as the opportunity for the interrogator to pass physical objects "through the hatch."

To pass the total Turing Test, the computer will need

- computer vision to perceive objects, and
- robotics to manipulate objects and move about.

These six disciplines compose most of AI.

**Thinking humanly: The cognitive modeling approach**

If we are going to say that a given program thinks like a human, we must have some way of determining how humans think.

We need to get inside the actual workings of human minds.

There are three ways to do this:

- Introspection—trying to catch our own thoughts as they go by.
- Psychological experiments—observing a person in action; and
- Brain imaging—observing the brain in action

Once we have a sufficiently precise theory of the mind, it becomes possible to express the theory as a computer program.

If the program's input–output behavior matches corresponding human behavior, that is evidence that some of the program's mechanisms could also be operating in humans.

The interdisciplinary field of cognitive science brings together computer models from AI and experimental techniques from psychology to construct precise and testable theories of the human mind.

**Thinking rationally: The "laws of thought" approach**

This can be done in two ways

**SYLLOGISM**: an instance of a form of reasoning in which a conclusion is drawn from two given or assumed propositions, "Socrates is a man; all men are mortal; therefore, Socrates is mortal."

**LOGIC**: study of laws of thought to govern the operation of the mind not easy to take informal knowledge and state it in the formal terms required by logical notation.

There are two main obstacles to this approach.

First, it is not easy to take informal knowledge and state it in the formal terms required by logical notation, particularly when the knowledge is less than 100% certain.

Second, there is a big difference between solving a problem "in principle" and solving it in practice.

**Acting rationally: The rational agent approach**

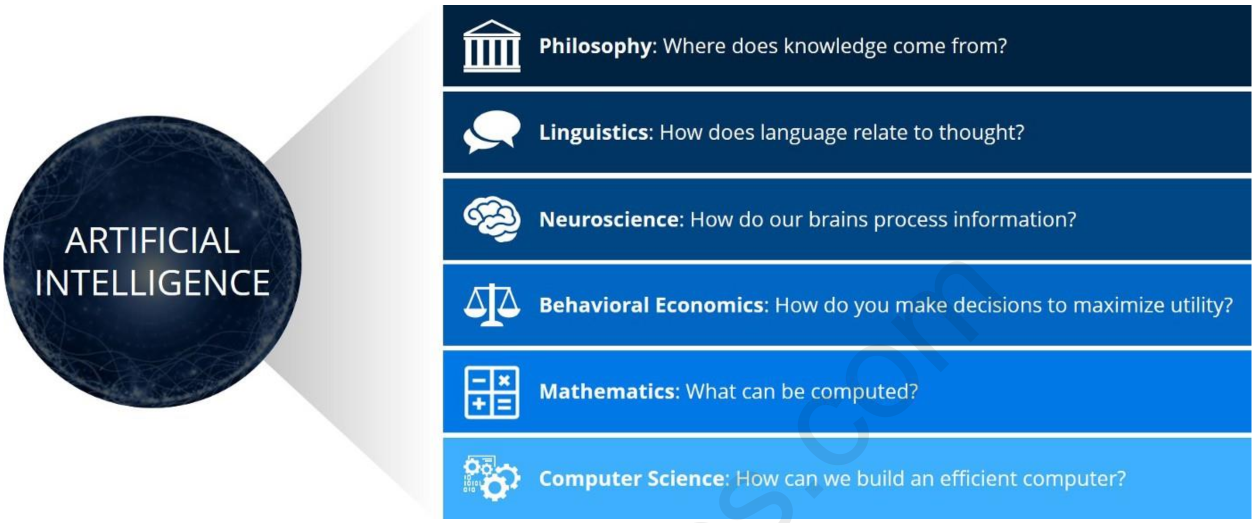
An agent is just something that acts

A computer agent is expected to have other attributes distinguish them from mere programs. They are:

- Operate autonomously
- Perceive their environment
- Persist over a prolonged time period
- Adapt to change, and
- Create and pursue goals

## 1.2 The Foundations of AI

The Foundations of AI gives a brief history of the disciplines that contributed ideas, viewpoints, and techniques to AI. They are




Different people think of AI differently.

Two important questions to ask are: Are you concerned with thinking or behavior?

Do you want to model humans or work from an ideal standard?

Intelligence is concerned mainly with rational action. Ideally, an intelligent agent takes the best possible action in a situation.

We will study the problem of building agents that are intelligent in this sense.

**Philosophers** (going back to 400 B.c.) made AI conceivable by considering the ideas that the mind is in some ways like a machine, that it operates on knowledge encoded in some internal language, and that thought can be used to choose what actions to take.

**Mathematicians** provided the tools to manipulate statements of logical certainty as well as uncertain, probabilistic statements. They also set the groundwork for understanding computation and reasoning about algorithms.

**Economists** formalized the problem of making decisions that maximize the expected outcome to the decision-maker.

**Psychologists** adopted the idea that humans and animals can be considered information processing machines.

**Linguists** showed that language use fits into this model.

**Computer engineers** provided the artifacts that make AI applications possible.

AI programs tend to be large, and they could not work without the great advances in speed and memory that the computer industry has provided.

**Control theory** deals with designing devices that act optimally on the basis of feedback from the environment. Initially, the mathematical tools of control theory were quite different from AI, but the fields are coming closer together.

**Philosophy**

• Can formal rules be used to draw valid conclusions?

• How does the mind arise from a physical brain?

• Where does knowledge come from?

• How does knowledge lead to action?

Aristotle (384–322 B.C.), was the first to formulate a precise set of laws governing the rational part of the mind.

He developed an informal system of syllogisms for proper reasoning, which in principle allowed one to generate conclusions mechanically, given initial premises.

In his 1651 book Leviathan, Thomas Hobbes suggested the idea of an "artificial animal," arguing "For what is the heart but a spring; and the nerves, but so many strings; and the joints, but so many wheels."

It's one thing to say that the mind operates, at least in part, according to logical rules, and to build physical systems that emulate some of those rules; it's another to say that the mind itself is such a physical system.

The terms in philosophy which is important in terms of AI are

Rationalism: power of reasoning in understanding the world

Dualism: there is a part of the human mind (or soul or spirit) that is outside of nature, exempt from physical laws

Materialism: brain's operation according to the laws of physics constitutes the mind

Induction: general rules are acquired by exposure to repeated associations between their elements

Logical positivism: doctrine holds that all knowledge can be characterized by logical theories connected, ultimately, to observation sentences that correspond to sensory inputs; thus logical positivism combines rationalism and empiricism

confirmation theory: attempted to analyze the acquisition of knowledge from experience

**Mathematics**

What are the formal rules to draw valid conclusions?

What can be computed?

How do we reason with uncertain information?

Philosophers staked out some of the fundamental ideas of AI, but the leap to a formal science required a level of mathematical formalization in three fundamental areas: logic, computation, and probability.

The idea of formal logic can be traced back to the philosophers of ancient Greece, but its mathematical development really began with the work of George Boole (1815–1864), whoworked out the details of propositional, or Boolean, logic (Boole, 1847).

In 1879, GottlobFrege (1848–1925) extended Boole's logic to include objects and relations, creating the firstorder logic that is used today.

Alfred Tarski (1902–1983) introduced a theory of reference that shows how to relate the objects in a logic to objects in the real world.

Besides logic and computation, the third great contribution of mathematics to AI is the theory of probability.

Thomas Bayes (1702–1761), proposed a rule for updating probabilities in the light of new evidence.

Bayes' rule underlies most modern approaches to uncertain reasoning in AI systems.

**Economics**

How should we make decisions so as to maximize payoff?

How should we do this when others may not go along?

How should we do this when the payoff may be far in the future?

The science of economics got its start in 1776, when Scottish philosopher Adam Smith (1723–1790) published An Inquiry into the Nature and Causes of the Wealth of Nations.

While the ancient Greeks and others had made contributions to economic thought, Smith was the first to treat it as a science, using the idea that economies can be thought of as consisting of individual agents maximizing their own economic well-being.

Most people think of economics as being about money, but economists will say that they are really studying how people make choices that lead to preferred outcomes.

Decision theory, which combines probability theory with utility theory, provides a formal and complete framework for decisions (economic or otherwise) made under uncertainty

**Neuroscience**

How do brains process information?

Neuroscience is the study of the nervous system, particularly the brain.

Although the exact way in which the brain enables thought is one of the great mysteries of science, the fact that it does enable thought has been appreciated for thousands of years because of the evidence that strong blows to the head can lead to mental incapacitation.

**Psychology**

• How do humans and animals think and act?

The origins of scientific psychology are usually traced to the work of the German physicist Hermann von Helmholtz (1821–1894) and his student Wilhelm Wundt (1832–1920).

Helmholtz applied the scientific method to the study of human vision, and his Handbook of Physiological Optics is even now described as “the single most important treatise on the physics and physiology of human vision”

**Computer Engineering**

How can we build an efficient computer?

For artificial intelligence to succeed, we need two things: intelligence and an artifact. The computer has been the artifact of choice.

Control theory and Cybernetics

How can artifacts operate under their own control?

Ktesibios of Alexandria (c. 250 B.C.) built the first self-controlling machine: a water clock with a regulator that maintained a constant flow rate. This invention changed the definition of what an artifact could do.

**Linguistics**

How does language relate to thought?

In 1957, B. F. Skinner published Verbal Behavior. This was a comprehensive, detailed account of the behaviorist approach to language learning, written by the foremost expert inthe field.

Modern linguistics and AI, then, were "born" at about the same time, and grew up together, intersecting in a hybrid field called computational linguistics or natural language processing.

## 1.2 History of AI

Artificial Intelligence is not a new word and not a new technology for researchers. This technology is much older than you would imagine. Even there are the myths of Mechanical men in Ancient Greek and Egyptian Myths. Following are some milestones in the history of AI which defines the journey from the AI generation to till date development.

**Maturation of Artificial Intelligence (1943-1952)**

Year 1943: The first work which is now recognized as AI was done by Warren McCulloch and Walter pits in 1943. They proposed a model of artificial neurons.

Year 1949: Donald Hebb demonstrated an updating rule for modifying the connection strength between neurons. His rule is now called Hebbian learning.

Year 1950: The Alan Turing who was an English mathematician and pioneered Machine learning in 1950. Alan Turing publishes "Computing Machinery and Intelligence" in which he proposed a test. The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence, called a Turing test.

**The birth of Artificial Intelligence (1952-1956)**

Year 1955: An Allen Newell and Herbert A. Simon created the "first artificial intelligence program"Which was named as "Logic Theorist". This program had proved 38 of 52 Mathematics theorems, and find new and more elegant proofs for some theorems.

Year 1956: The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI coined as an academic field. At that time high-level computer languages such as FORTRAN, LISP, or COBOL were invented. And the enthusiasm for AI was very high at that time.

**The golden years-Early enthusiasm (1956-1974)**

Year 1966: The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first chatbot in 1966, which was named as ELIZA.

Year 1972: The first intelligent humanoid robot was built in Japan which was named as WABOT-1.

**The first AI winter (1974-1980)**

The duration between years 1974 to 1980 was the first AI winter duration. AI winter refers to the time period where computer scientist dealt with a severe shortage of funding from government for AI researches. o During AI winters, an interest of publicity on artificial intelligence was decreased.

**A boom of AI (1980-1987)**

Year 1980: After AI winter duration, AI came back with "Expert System". Expert systems were programmed that emulate the decision-making ability of a human expert.

In the Year 1980, the first national conference of the American Association of Artificial Intelligence was held at Stanford University.

**The second AI winter (1987-1993)**

The duration between the years 1987 to 1993 was the second AI Winter duration.

Again Investors and government stopped in funding for AI research as due to high cost but not efficient result. The expert system such as XCON was very cost effective.

**The emergence of intelligent agents (1993-2011)**

Year 1997: In the year 1997, IBM Deep Blue beats world chess champion, Gary Kasparov, and became the first computer to beat a world chess champion.

Year 2002: for the first time, AI entered the home in the form of Roomba, a vacuum cleaner.

Year 2006: AI came in the Business world till the year 2006. Companies like Facebook, Twitter, and Netflix also started using AI.

**Deep learning, big data and artificial general intelligence (2011-present)**

Year 2011: In the year 2011, IBM's Watson won jeopardy, a quiz show, where it had to solve the complex questions as well as riddles. Watson had proved that it could understand natural language and can solve tricky questions quickly.

Year 2012: Google has launched an Android app feature "Google now", which was able to provide information to the user as a prediction.

Year 2014: In the year 2014, Chatbot "Eugene Goostman" won a competition in the infamous "Turing test."

Year 2018: The "Project Debater" from IBM debated on complex topics with two master debaters and also performed extremely well.

Google has demonstrated an AI program "Duplex" which was a virtual assistant and P;89+which had taken hairdresser appointment on call, and lady on other side didn't notice that she was talking with the machine.

Now AI has developed to a remarkable level.

The concept of Deep learning, big data, and data science are now trending like a boom. Nowadays companies like Google, Facebook, IBM, and Amazon are working with AI and creating amazing devices. The future of Artificial Intelligence is inspiring and will come with high intelligence.

## 1.4 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.

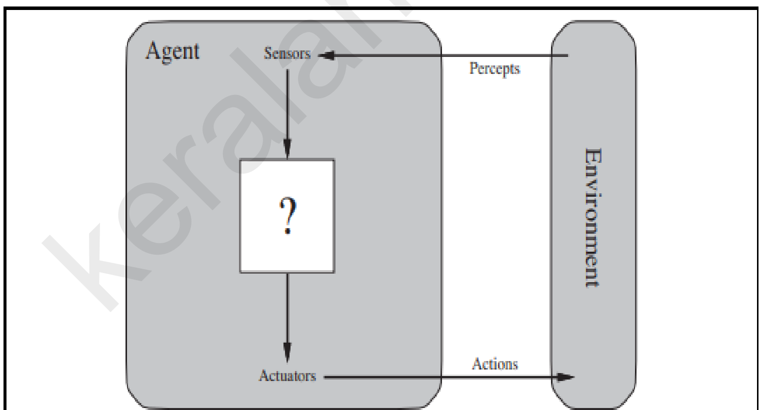



**Figure 2.1** Agents interact with environments through sensors and actuators.

An agent can be anything that perceive its environment through sensors and act upon that environment through actuators.

An Agent runs in the cycle of perceiving, thinking, and acting. An agent can be:

- Human-Agent: A human agent has eyes, ears, and other organs which work for sensors and hand, legs, vocal tract work for actuators.

- Robotic Agent: A robotic agent can have cameras, infrared range finder, NLP for sensors and various motors for actuators.
- Software Agent: Software agent can have keystrokes, file contents as sensory input and act on those inputs and display output on the screen.

The term **percept** 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.

In general, an agent's choice of action at any given instant can depend on the entire percept sequence observed to date, but not on anything it hasn't perceived.

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

Internally, the agent function for an artificial agent will be implemented by an agent program.

**Intelligent Agents**

An intelligent agent is an autonomous entity which act upon an environment using sensors and actuators for achieving goals.

An intelligent agent may learn from the environment to achieve their goals. A thermostat is an example of an intelligent agent.

Following are the main four rules for an AI agent:

- Rule 1: An AI agent must have the ability to perceive the environment.
- Rule 2: The observation must be used to make decisions.
- Rule 3: Decision should result in an action.
- Rule 4: The action taken by an AI agent must be a rational action.

## The Vacuum Cleaner World

This particular world has just two locations: squares A and B.

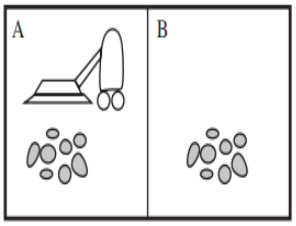



**Figure 2.2** A vacuum-cleaner world with just two locations.

The vacuum agent perceives which square it is in and whether there is dirt in the square. It can choose to move left, move right, suck up the dirt, or do nothing.

One very simple agent function is the following: if the current square is dirty, then suck; otherwise, move to the other square.

A partial tabulation of this agent function is shown in Figure.

| 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* |
| ⋮ | ⋮ |

**Figure 2.3** Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

And an agent program that implements it appears in Figure

```
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
```

**Figure 2.8** The agent program for a simple reflex agent in the two-state vacuum environment. This program implements the agent function tabulated in Figure 2.3.

Percepts: location and contents, e.g., [A,Dirty]

Actions: Left, Right, Suck, NoOp

Agent's function → look-up table

## 1.5 Good behavior: The concept of rationality

A rational agent is said to perform the right things. AI is about creating rational agents to use for game theory and decision theory for various real-world scenarios.

A rational agent is an agent which has clear preference, models uncertainty, and acts in a way to maximize its performance measure with all possible actions.

For an AI agent, the rational action is most important because in AI reinforcement learning algorithm, for each best possible action, agent gets the positive reward and for each wrong action, an agent gets a negative reward.

This notion of desirability is captured by a performance measure that evaluates any given sequence of environment states.

**Vacuum Cleaner Revisited**

We might propose to measure performance by the amount of dirt cleaned up in a single eight hour shift. With a rational agent, of course, what you ask for is what you get.

A rational agent can maximize this performance measure by cleaning up the dirt, then dumping it all on the floor, then cleaning it up again, and so on.

A more suitable performance measure would reward the agent for having a clean floor. For example, one point could be awarded for each clean square at each time step (perhaps with a penalty for electricity consumed and noise generated).

**Rationality**

The rationality of an agent is measured by its performance measure.

Rationality can be judged on the basis of following points:

- **Performance measure** which defines the success criterion.
- Agent prior knowledge of its **environment**.
- Best possible **actions** that an agent can perform.
- The sequence of **percepts**.

## 1.6 The nature of Environments

Task environments, are essentially the "problems" to which rational agents are the "solutions."

Task environment used to specify the performance measure, the environment, and the agent's actuators and sensors.

We can call it as PEAS which stands for a Performance measure, Environment, Actuator, Sensor.

PEAS System is used to categorize similar agents together. The PEAS system delivers the performance measure with respect to the environment, actuators, and sensors of the respective agent. Most of the highest performing agents are Rational Agents.

It is made up of four words:

- P: Performance measure
- E: Environment
- A: Actuators
- S: Sensors

Here performance measure is the objective for the success of an agent's behavior

**PEAS for self-driving cars**:

Let's suppose a self-driving car then PEAS representation will be:

- Performance: Safety, time, legal drive, comfort
- Environment: Roads, other vehicles, road signs, pedestrian
- Actuators: Steering, accelerator, brake, signal, horn
- Sensors: Camera, GPS, speedometer, odometer, accelerometer, sonar.

**Performance Measure**

Desirable qualities to measure performance include getting to the correct destination; minimizing fuel consumption and wear and tear; minimizing the trip time or cost; minimizing violations of traffic laws and disturbances to other drivers; maximizing safety and passenger comfort; maximizing profits. Obviously, some of these goals conflict, so tradeoffs will be required.

**Environment**

Any taxi driver must deal with a variety of roads, ranging from rural lanes and urban alleys to 12-lane freeways. The roads contain other traffic, pedestrians, stray animals, road works, police cars, puddles, and potholes. The taxi must also interact with potential and actual passengers. There are also some optional choices. The taxi might need to operate in Southern California, where snow is seldom a problem, or in Alaska, where it seldom is not. It could always be driving on the right, or we might want it to be flexible enough to drive on the left when in Britain or Japan. Obviously, the more restricted the environment, the easier the design problem.

**Actuators**

The actuators for an automated taxi include those available to a human driver: control over the engine through the accelerator and control over steering and braking. In addition, it will need output to a display screen or voice synthesizer to talk back to the passengers, and perhaps some way to communicate with other vehicles, politely or otherwise.

**Sensors**

The basic sensors for the taxi will include one or more controllable video cameras so that it can see the road; it might augment these with infrared or sonar sensors to detect distances to other cars and obstacles. To avoid speeding tickets, the taxi should have a speedometer, and to control the vehicle properly, especially on curves, it should have an accelerometer. To determine the mechanical state of the vehicle, it will need the usual array of engine, fuel, and electrical system sensors. Like many human drivers, it might want a global positioning system (GPS) so that it doesn't get lost. Finally, it will need a keyboard or microphone for the passenger to request a destination

**Example of Agents with their PEAS representation**

| 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 |

**Figure 2.5** Examples of agent types and their PEAS descriptions.

**Properties of task environments**

An environment in artificial intelligence is the surrounding of the agent. The agent takes input from the environment through sensors and delivers the output to the environment through actuators. There are several types of environments:

- Fully Observable vs Partially Observable
- Deterministic vs Stochastic
- Single-agent vs Multi-agent
- Static vs Dynamic
- Discrete vs Continuous
- Episodic vs Sequential
- Known vs Unknown

**Fully Observable vs Partially Observable**

- When an agent sensor is capable to sense or access the complete state of an agent at each point in time, it is said to be a fully observable environment else it is partially observable.
- Maintaining a fully observable environment is easy as there is no need to keep track of the history of the surrounding.
- An environment is called unobservable when the agent has no sensors in all environments.
- Examples:
  - Chess – the board is fully observable, and so are the opponent's moves.
  - Driving – the environment is partially observable because what's around the corner is not known.

**Deterministic vs Stochastic**

- When a uniqueness in the agent's current state completely determines the next state of the agent, the environment is said to be deterministic.
- The stochastic environment is random in nature which is not unique and cannot be completely determined by the agent.
- Examples:
  - Chess – there would be only a few possible moves for a coin at the current state and these moves can be determined.
  - Self-Driving Cars- the actions of a self-driving car are not unique, it varies time to time.

**Single-agent vs Multi-agent**

- An environment consisting of only one agent is said to be a single-agent environment.
- A person left alone in a maze is an example of the single-agent system.

- An environment involving more than one agent is a multi-agent environment.
- The game of football is multi-agent as it involves 11 players in each team.

**Dynamic vs Static**

- An environment that keeps constantly changing itself when the agent is up with some action is said to be dynamic.
- A roller coaster ride is dynamic as it is set in motion and the environment keeps changing every instant.
- An idle environment with no change in its state is called a static environment.
- An empty house is static as there's no change in the surroundings when an agent enters.

**Discrete vs Continuous**

- If an environment consists of a finite number of actions that can be deliberated in the environment to obtain the output, it is said to be a discrete environment.
- The game of chess is discrete as it has only a finite number of moves. The number of moves might vary with every game, but still, it's finite.
- The environment in which the actions are performed cannot be numbered i.e. is not discrete, is said to be continuous.
- Self-driving cars are an example of continuous environments as their actions are driving, parking, etc. which cannot be numbered.

**Episodic vs Sequential**

- In an Episodic task environment, each of the agent's actions is divided into atomic incidents or episodes. There is no dependency between current and previous incidents. In each incident, an agent receives input from the environment and then performs the corresponding action.
- Example: Consider an example of Pick and Place robot, which is used to detect defective parts from the conveyor belts. Here, every time robot (agent) will make the decision on the current part i.e. there is no dependency between current and previous decisions.
- In a Sequential environment, the previous decisions can affect all future decisions. The next action of the agent depends on what action he has taken previously and what action he is supposed to take in the future.
- Example:
    - Checkers- Where the previous move can affect all the following moves.

**Known vs Unknown**

In a known environment, the output for all probable actions is given. Obviously, in case of unknown environment, for an agent to make a decision, it has to gain knowledge about how the environment works.

| 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.

## 1.7 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.

This program will run on some sort of computing device with physical sensors and actuators called the architecture.

**agent = architecture + program**

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 program: use current percept as input from the sensors and return an action to the actuators.

Agent function: takes the entire percept history

Table Driven Approach

To build a rational agent in this way, we as designers must construct a table that contains the appropriate action for every possible percept sequence.

**function** TABLE-DRIVEN-AGENT(*percept*) **returns** an action
  **persistent**: *percepts*, a sequence, initially empty
    *table*, a table of actions, indexed by percept sequences, initially fully specified

  append *percept* to the end of *percepts*
  *action* ← LOOKUP(*percepts*, *table*)
  **return** *action*

**Figure 2.7** The TABLE-DRIVEN-AGENT program is invoked for each new percept and returns an action each time. It retains the complete percept sequence in memory.

Let P be the set of possible percepts and let T be the lifetime of the agent (the total number of percepts it will receive)

The lookup table will contain $\sum_{t=1}^{T} |\mathcal{P}|^t$ entries.

Consider the automated taxi: the visual input from a single camera comes in at the rate of roughly 27 megabytes per second (30 frames per second, 640 × 480 pixels with 24 bits of color information). This gives a lookup table with over 10250,000,000,000 entries for an hour's driving.

Even the lookup table for chess a tiny, well-behaved fragment of the real world would have at least 10150 entries.

The daunting size of these tables (the number of atoms in the observable universe is less than 1080) means that

- no physical agent in this universe will have the space to store the table,
- the designer would not have time to create the table,
- no agent could ever learn all the right table entries from its experience, and
- even if the environment is simple enough to yield a feasible table size, the designer still has no guidance about how to fill in the table entries.

**Types of Agent Programs**

Four basic kinds of agent programs that embody the principles underlying almost all intelligent systems:

1. Simple reflex agents;
2. Model-based reflex agents;
3. Goal-based agents; and
4. Utility-based agents

**Simple reflex agents**

- Select actions on the basis of the current percept, ignoring the rest of the percept history
- Agents do not have memory of past world states or percepts.
- So, actions depend solely on current percept. Action becomes a "reflex."

Agents select actions on the basis of the current percept, ignoring the rest of the percept history. For example, the vacuum agent is a simple reflex agent, because its decision is based only on the current location and on whether that location contains dirt. An agent program for this agent is shown in Figure.

```
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
```

**Figure 2.8** The agent program for a simple reflex agent in the two-state vacuum environment. This program implements the agent function tabulated in Figure 2.3.

Simple reflex behaviors occur even in more complex environments. Imagine yourself as the driver of the automated taxi. If the car in front brakes and its brake lights come on, then you should notice this and initiate braking. In other words, some processing is done on the visual input to establish the condition we call "The car in front is braking." Then, this triggers some established connection in the agent program to the action "initiate braking." We call such a connection a condition–action rule, written as

**if car-in-front-is-braking then initiate-braking**

Figure below gives the structure of the general program in schematic form, showing how the condition–action rules allow the agent to make the connection from percept to action.

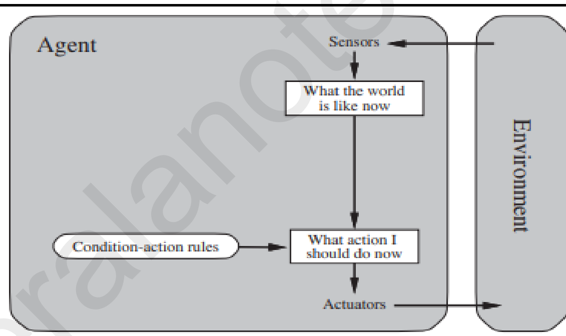



**Figure 2.9** Schematic diagram of a simple reflex agent.

The agent program, which is also very simple, is shown in Figure below.

```
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
```

**Figure 2.10** A simple reflex agent. It acts according to a rule whose condition matches the current state, as defined by the percept.

The INTERPRET-INPUT function generates an abstracted description of the current state from the percept, and.

The RULE-MATCH function returns the first rule in the set of rules that matches the given state description.

This will work only if the correct decision can be made on the basis of only the current percept—that is, only if the environment is fully observable.

Even a little bit of unobservability can cause serious trouble. For example, the braking rule given earlier assumes that the condition car-in-front-is-braking can be determined from the current percept—a single frame of video. This works if the car in front has a centrally mounted brake light.

Infinite loops are often unavoidable for simple reflex agents operating in partially observable environments. Escape from infinite loops is possible if the agent can randomize its actions.

**Model-based reflex agents**

It works by finding a rule whose condition matches the current situation

Key difference (with respect to simple reflex agents):

- Agents have internal state, which is used to keep track of past states of the world.
- Agents have the ability to represent change in the World.

The current state is stored inside the agent which maintains some kind of structure describing the part of the world which cannot be seen.

Internal state information as time goes by requires two kinds of knowledge to be encoded in the agent program

- we need some information about how the world evolves independently of the agent
- we need some information about how the agent's own actions affect the world Knowledge about "how the world works is called a model of the world.

An agent that uses such a model is called a model-based agent.

Figure below gives the structure of the model-based reflex agent with internal state, showing how the current percept is combined with the old internal state to generate the updated description of the current state, based on the agent's model of how the world works.

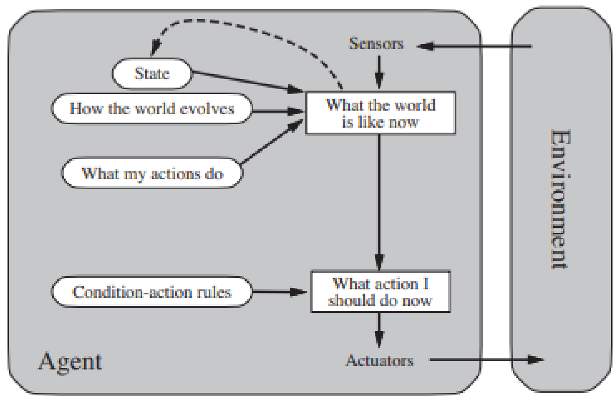



**Figure 2.11** A model-based reflex agent.

The agent program is shown in Figure below. UPDATE-STATE, which is responsible for creating the new internal state description

**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*

**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 agents

Knowing something about the current state of the environment is not always enough to decide what to do. For example, at a road junction, the taxi can turn left, turn right, or go straight on. The correct decision depends on where the taxi is trying to get to. In other words, as well as a current state description, the agent needs some sort of goal information that describes situations that are desirable—for example, being at the passenger's destination. The agent program can combine this with the model (the same information as was used in the model based reflex agent) to choose actions that achieve the goal.

Figure below shows the goal-based agent's structure

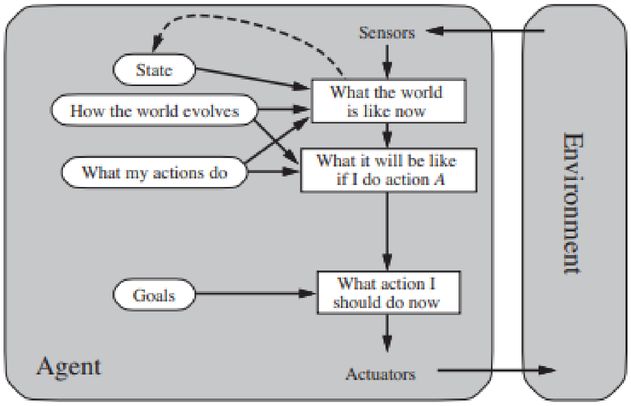



**Figure 2.13** A model-based, goal-based agent. It keeps track of the world state as well as a set of goals it is trying to achieve, and chooses an action that will (eventually) lead to the achievement of its goals.

- Sometimes goal-based action selection is straightforward—for example, when goal satisfaction results immediately from a single action.
- Sometimes it will be trickier—for example, when the agent has to consider long sequences of twists and turns in order to find a way to achieve the goal.
- Search and planning are the subfields of AI devoted to finding action sequences that achieve the agent's goals.

**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. Because "happy" does not sound very scientific, economists and computer scientists use the term utility instead

An agent's utility function is essentially an internalization of the performance measure. 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.

The utility-based agent structure appears in Figure below.

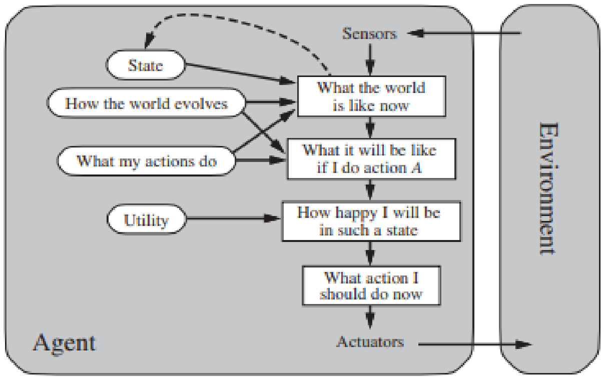



**Figure 2.14** A model-based, utility-based agent. It uses a model of the world, along with a utility function that measures its preferences among states of the world. Then it chooses the action that leads to the best expected utility, where expected utility is computed by averaging over all possible outcome states, weighted by the probability of the outcome.

Learning agents

A learning agent can be divided into four conceptual components, as shown in Figure below.

- Learning Element
- Performance Element
- Critic
- Problem Generator

The most important distinction is between the learning element, which is responsible for making improvements, and the performance element, which is responsible for selecting external actions.

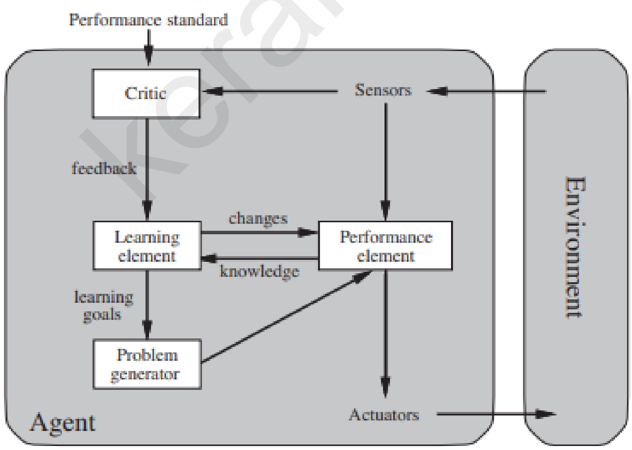



**Figure 2.15** A general learning agent.

The **performance element** is what we have previously considered to be the entire agent: it takes in percepts and decides on actions.

The **learning element** uses CRITIC feedback from the critic on how the agent is doing and determines how the performance element should be modified to do better in the future.

The **critic** tells the learning element how well the agent is doing with respect to a fixed performance standard. The critic is necessary because the percepts themselves provide no indication of the agent's success.

The last component of the learning agent is the **problem generator**. It is responsible for suggesting actions that will lead to new and informative experiences.

**How the components of agent programs work (Agent Transitions)**

We can place the representations along an axis of increasing complexity and expressive power—

- atomic,
- factored, and
- structured.

**Atomic representation**

In an atomic representation each state of the world is indivisible—it has no internal structure. Consider the problem of finding a driving route from one end of a country to the other via some sequence of cities. For the purposes of solving this problem, it may suffice to reduce the state of world to just the name of the city we are in—a single atom of knowledge; a "black box" whose only discernible property is that of being identical to or different from another black box.

The algorithms underlying search and game-playing, Hidden Markov models, and Markov decision processes all work with atomic representations.

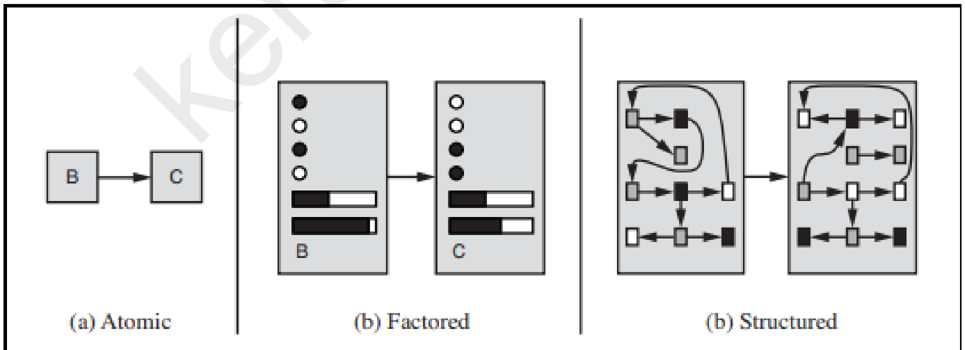



**Figure 2.16** Three ways to represent states and the transitions between them. (a) Atomic representation: a state (such as B or C) is a black box with no internal structure; (b) Factored representation: a state consists of a vector of attribute values; values can be Boolean, real-valued, or one of a fixed set of symbols. (c) Structured representation: a state includes objects, each of which may have attributes of its own as well as relationships to other objects.

**Factored Representation**

A factored representation splits up each state into a fixed set of variables or attributes, each of which can have a value. While two different atomic states ATTRIBUTE VALUE have nothing in common—they are just different black boxes—two different factored states can share some attributes (such as being at some particular GPS location) and not others (such as having lots of gas or having no gas); this makes it much easier to work out how to turn one state into another.

With factored representations, we can also represent uncertainty—for example, ignorance about the amount of gas in the tank can be represented by leaving that attribute blank.

Many important areas of AI are based on factored representations, including constraint satisfaction algorithms, propositional logic, planning, and Bayesian networks.

**Structured Representation**

In a structured representation, objects and their various and varying relationships can be described explicitly. Structured representations underlie relational databases and first-order logic, first-order probability models, knowledge-based learning and much of natural language understanding.