What is Artificial Intelligence

Artificial Intelligence (AI) is the study of how to make computers do things which, at the moment, people do better.

Artificial Intelligence is the science and engineering of making intelligent machines" … "[where] intelligence is the computational part of the ability to achieve goals in the world" (original definition by John McCarthy who coined the term 'Artificial Intelligence' in 1955)

Artificial Intelligence is making a machine behave in ways that would be called intelligent if a human were so behaving" (alternative definition by John McCarthy who coined the term 'Artificial Intelligence' in 1955)

Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions. The term may also be applied to any machine that exhibits traits associated with a human mind such as learning and problem-solving.

Artificial Intelligence (AI) is a branch of Science which deals with helping machines finding solutions to complex problems in a more human-like fashion.

This generally involves borrowing characteristics from human intelligence, and applying them as algorithms in a computer friendly way.

A more or less flexible or efficient approach can be taken depending on the requirements established, which influences how artificial the intelligent behaviour appears.

AI is generally associated with Computer Science, but it has many important links with other fields such as Maths, Psychology, Cognition, Biology and Philosophy, among many others.

Our ability to combine knowledge from all these fields will ultimately benefit our progress in the quest of creating an intelligent artificial being.

AI currently encompasses a huge variety of subfields, from general-purpose areas such as perception and logical reasoning, to specific tasks such as playing chess, proving mathematical theorems, writing poetry, and diagnosing diseases.

Often, scientists in other fields move gradually into artificial intelligence, where they find the tools and vocabulary to systematize and automate the intellectual tasks on which they have been working all their lives.

Similarly, workers in AI can choose to apply their methods to any area of human intellectual Endeavour. In this sense, it is truly a universal field.

AI Techniques – Introduction

Intelligence requires knowledge but knowledge possesses less desirable properties such as

- It is voluminous.
- It is difficult to characterize accurately.
- It is constantly changing.
- It differs from data by being organized in a way that corresponds to its application

An AI technique is a method that exploits knowledge that is represented so that

- The knowledge captures generalizations; situations that share properties, are grouped together, rather than being allowed separate representation.
- It can be understood by people who must provide it; although for many programs the bulk of the data may come automatically, such as from readings. In many AI domains people must supply the knowledge to programs in a form the people understand and in a form that is acceptable to the program.

  An AI technique is a method that exploits knowledge that is represented so that

- It can be easily modified to correct errors and reflect changes in real conditions.
- It can be widely used even if it is incomplete or inaccurate.
- It can be used to help overcome its own sheer bulk by helping to narrow the range of possibilities that must be usually considered.

Problem Spaces and Search

Building a system to solve a problem requires the following steps.

- Define the problem precisely including detailed specifications and what constitutes an acceptable solution.
- Analyze the problem thoroughly for some features may have a dominant affect on the chosen method of solution.
- Isolate and represent the background knowledge needed in the solution of the problem.

- Choose the best problem solving techniques in the solution.

Defining the Problem as state Search

- Problems dealt with in artificial intelligence generally use a common term called 'state'.
- A state represents a status of the solution at a given step of the problem solving procedure. The solution of a problem, thus, is a collection of the problem states.
- The problem solving procedure applies an operator to a state to get the next state.
- Then it applies another operator to the resulting state to derive a new state. The process of applying an operator to a state and its subsequent transition to the next state, thus, is continued until the goal (desired) state is derived.
- Such a method of solving a problem is generally referred to as state space approach For example, in order to solve the problem play a game, which is restricted to two person table or board games, we require the rules of the game and the targets for winning as well as a means of representing positions in the game.
- The opening position can be defined as the initial state and a winning position as a goal state, there can be more than one. legal moves allow for transfer from initial state to other states leading to the goal state.
- The storage also presents another problem but searching can be achieved by hashing. The number of rules that are used must be minimized and the set can be produced by expressing each rule in as general a form as possible.
- The representation of games in this way leads to a state space representation and it is natural for well organised games with some structure.
- This representation allows for the formal definition of a problem which necessitates the movement from a set of initial positions to one of a set of target positions.
- It means that the solution involves using known techniques and a systematic search.
- This is quite a common method in AI.

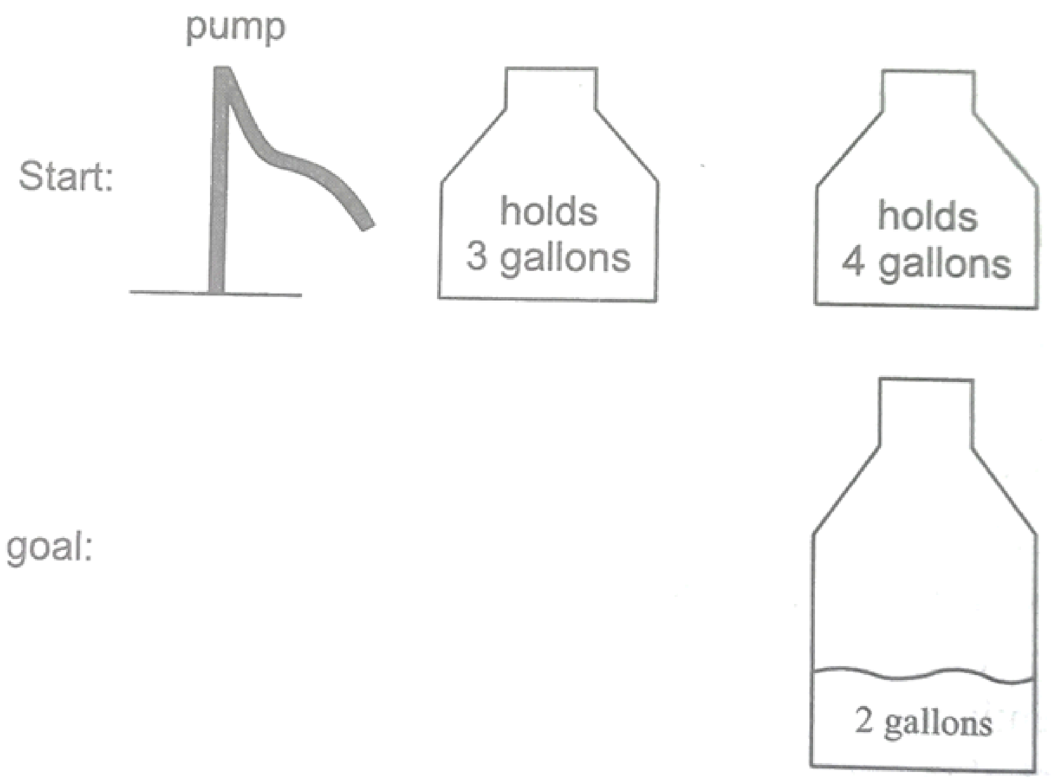
Formal description of a problem

- Define a state space that contains all possible configurations of the relevant objects, without enumerating all the states in it. A state space represents a problem in terms of states and operators that change state.
- Define some of these states as possible initial states;
- Specify one or more as acceptable solutions, these are goal states.
- Specify a set of rules as the possible actions allowed. This involves thinking about the generality of the rules, the assumptions made in the informal presentation and how much work can be anticipated by inclusion in the rules.

Example Problem

The water jug problem :

We are given two jugs, a four-gallon one and a three-gallon one. Neither has any measuring markers on it. There is a pump which can be used to fill the jugs with water. How can we get exactly two gallons of water into the four-gallon jug?



Start State : (0,0)

Goal State : (2,n) for any value of n.

1. $(x,y) \rightarrow (4,y)$ // Fill the 4 gallon jug
if $x<4$

2. $(x,y) \rightarrow (x,3)$ // Fill the 3 gallon jug
if $y<3$

3. $(x,y) \rightarrow (x-d,y)$ // Pour some water out of the 4-gallon jug
if $x>0$

4. $(x,y) \rightarrow (x,y-d)$ // Pour some water out of the 3-gallon jug
if $x>0$

5. $(x,y) \rightarrow (0,y)$ // Empty the 4-gallon jug on the ground
if $x>0$

6. $(x,y) \rightarrow (x,0)$ // Empty the 3-gallon jug on the ground
if $y>0$

7. $(x,y) \rightarrow (4,y-(4-x))$ // Pour water from the 3-gallon jug into the 4 gallons until the 4-gallon jug is full
if $x+y \geq 4$ and $y>0$

8. $(x,y) \rightarrow (x-(3-y),3)$ // Pour water from the 4-gallon jug into the 3 gallons until the 3-gallon jug is full
if $x+y \geq 3$ and $x>0$

9. $(x,y) \rightarrow (x+y,0)$ // Pour water all the water from the 3-gallon jug into the 4-gallon jug
if $x+y \leq 4$ and $y>0$

10. $(x,y) \rightarrow (0,x+y)$ // Pour water all the water from the 4-gallon jug into the 3-gallon jug
if $x+y \leq 3$ and $x>0$

The water jug problem – Solution

Gallons in the 4-Gallon jug	Gallons in the 3-Gallon jug	Rule Applied
0	0	2
0	3	9
3	0	2
3	3	7
4	2	5
0	2	9
2	0	-

PRODUCTION SYSTEMS

- The production system is a model of computation that can be applied to implement search algorithms and model human problem solving.
- Such problem solving knowledge can be packed up in the form of little quanta called productions.
- A production is a rule consisting of a situation recognition part and an action part. A production is a situation-action pair in which the left side is a list of things to watch for and the right side is a list of things to do so.
- When productions are used in deductive systems, the situation that trigger productions are specified combination of facts.
- The actions are restricted to being assertion of new facts deduced directly from the triggering combination.
- Production systems may be called premise conclusion pairs rather than situation action pair

A production system consists of following components.

- A set of production rules, which are of the form A→B. Each rule consists of left hand side constituent that represent the current problem state and a right hand side that represent an output state. A rule is applicable if its left hand side matches with the current problem state.

- A database, which contains all the appropriate information for the particular task. Some part of the database may be permanent while some part of this may pertain only to the solution of the current problem.
- A control strategy that specifies order in which the rules will be compared to the database of rules and a way of resolving the conflicts that arise when several rules match simultaneously.
- A rule applier, which checks the capability of rule by matching the content state with the left hand side of the rule and finds the appropriate rule from database of rules.
- The important roles played by production systems include a powerful knowledge representation scheme. A production system not only represents knowledge but also action. It acts as a bridge between AI and expert systems. Production system provides a language in which the representation of expert knowledge is very natural.

Production Systems – Components

- We can represent knowledge in a production system as a set of rules of the form

If (condition) THEN (condition)

along with a control system and a database. The control system serves as a rule interpreter and sequencer. The database acts as a context buffer, which records the conditions evaluated by the rules and information on which the rules act.

- The production rules are also known as condition – action, antecedent – consequent, pattern – action, situation – response, feedback – result pairs.

If (you have an exam tomorrow) THEN (study the whole night)

Features of Production System

Some of the main features of production system are:

- **Expressiveness and intuitiveness**: In real world, many times situation comes like "if this happen-you will do that", "if this is so-then this should happen" and many more. The production rules essentially tell us what to do in a given situation.
- **Simplicity**: The structure of each sentence in a production system is unique and uniform as they use "IF-THEN" structure. This structure provides simplicity in knowledge representation. This feature of production system improves the readability of production rules.
- **Modularity**: This means production rule code the knowledge available in discrete pieces. Information can be treated as a collection of independent facts which may be added or deleted from the system with essentially no deletetious side effects.
- **Modifiability**: This means the facility of modifying rules. It allows the development of production rules in a skeletal form first and then it is accurate to suit a specific application.
- **Knowledge intensive**: The knowledge base of production system stores pure knowledge. This part does not contain any type of control or programming information. Each production rule is normally written as an English sentence; the problem of semantics is solved by the very structure of the representation

Disadvantages of production system

- **Opacity**: This problem is generated by the combination of production rules. The opacity is generated because of less prioritization of rules. More priority to a rule has the less opacity.
- **Inefficiency**: During execution of a program several rules may active. A well devised control strategy reduces this problem. As the rules of the production system are large in number and they are hardly written in hierarchical manner, it requires some forms of complex search through all the production rules for each cycle of control program.
- **Absence of learning**: Rule based production systems do not store the result of the problem for future use. Hence, it does not exhibit any type of learning capabilities. So for each time for a particular problem, some new solutions may come.

- **Conflict resolution:** The rules in a production system should not have any type of conflict operations. When a new rule is added to a database, it should ensure that it does not have any conflicts with the existing rule

Heuristic Search Technique – Introduction

- Heuristic search is defined as a procedure of search that endeavors to upgrade an issue by iteratively improving the arrangement dependent on a given heuristic capacity or a cost measure.
- This technique doesn't generally ensure to locate an ideal or the best arrangement, however, it may rather locate a decent or worthy arrangement inside a sensible measure of time and memory space. This is a sort of an alternate route as we regularly exchange one of optimality, culmination, exactness, or accuracy for speed.
- A Heuristic (or a heuristic capacity) investigates search calculations. At each stretching step, it assesses the accessible data and settles on a choice on which branch to follow

Hill Climbing

Hill Climbing is a kind of heuristic quest for logical progression issues in the field of Artificial Intelligence. Given a set of data sources and a better than average heuristic limit, it endeavors to find an adequate enough response for the issue. This course of action may not be the overall perfect most noteworthy.

Features of Hill Climbing

- Produce and Test variation: Hill Climbing is the variation of the Generate and Test strategy. The Generate and Test technique produce input which assists with choosing which bearing to move in the inquiry space.
- Use of Greedy Approach: Hill-climbing calculation search moves toward the path which improves the expense.
- No backtracking: It doesn't backtrack the pursuit space, as it doesn't recall the past states.

Types of Hill Climbing

1. **Simple Hill Climbing**

- Simple Hill climbing is the least difficult approach to execute a slope climbing calculation. It just assesses the neighbor hub state at once and chooses the first which enhances current expense and sets it as a present state.
- It just checks it's one replacement state, and on the off chance that it discovers superior to the present state, at that point move else be in a similar state.

Its features include:

- Less tedious
- Less ideal arrangement that isn't ensured

2. Steepest Ascent Hill Climbing

- The steepest-Ascent calculation is a variety of basic slope climbing calculations. It first examines all the neighboring nodes then selects the node closest to the answer state as of next node.
- This calculation looks at all the neighboring hubs of the present state and chooses one neighbor hub which is nearest to the objective state.
- This calculation expends additional time as it looks for different neighbors

3. **Stochastic Hill Climbing**

- Stochastic slope climbing doesn't analyze for all its neighbors before moving. It makes use of randomness as a part of the search process. It is also an area search algorithm, meaning that it modifies one solution and searches the relatively local area of the search space until the local optima is found .

- This suggests that it's appropriate on unimodal optimization problems or to be used after the appliance of a worldwide optimization algorithm.
- This calculation chooses one neighbor hub aimlessly and concludes whether to pick it as a present state or analyze another state.

4. **Simulated Annealing**

- Simulated Annealing is an algorithm that never makes a move towards lower esteem destined to be incomplete that it can stall out on a nearby extreme.
- Also, on the off chance that calculation applies an irregular stroll, by moving a replacement, at that point, it might finish yet not proficient.
- In terms of metallurgy, Annealing is a procedure of solidifying a metal or glass to a high temperature at that point cooling bit by bit, so this permits the metal to arrive at a low-vitality crystalline state.
- A similar procedure is utilized in reenacted toughening in which the calculation picks an arbitrary move, rather than picking the best move.

Problems associated with Hill Climbing

- **Local Maximum:** All the surrounding states have values lower than the current. With the implementation of the Greedy Approach, implies we won't be moving to a lower state. This ends the procedure despite the fact that there may have been a superior arrangement. As a workaround, we use backtracking.
- **Plateau:** All neighbors to it have a similar worth. This makes it difficult to pick a course. To dodge this, we haphazardly make a major jump.
- **Ridge:** At an edge, development in every conceivable course is descending. This makes it resemble a pinnacle and ends the procedure. To stay away from this, we may utilize at least two guidelines before testing.

Best First Search (BFS)

- Best first search uses the concept of a priority queue and heuristic search. It is a search algorithm that works on a specific rule.

- The aim is to reach the goal from the initial state via the shortest path.
- BFS is a Heuristic method, where search is carried out by using additional information to determine the next step towards finding the solution. Best First Search is an example of such algorithms.
- Informed search methods are more efficient, low in cost and high in performance as compared to the uninformed search methods.
- If we consider searching as a form of traversal in a graph, an uninformed search algorithm would blindly traverse to the next node in a given manner without considering the cost associated with that step.
- An informed search, like Best first search, on the other hand would use an evaluation function to decide which among the various available nodes is the most promising (or 'BEST') before traversing to that node.
- The Best first search uses the concept of a Priority queue and heuristic search. To search the graph space, the BFS method uses two lists for tracking the traversal. An 'Open' list which keeps track of the current 'immediate' nodes available for traversal and 'CLOSED' list that keeps track of the nodes already traversed.

Best First Search Algorithm

- Create 2 empty lists: OPEN and CLOSED
- Start from the initial node (say N) and put it in the 'ordered' OPEN list
- Repeat the next steps until GOAL node is reached
- If OPEN list is empty, then EXIT the loop returning 'False'
- Select the first/top node (say N) in the OPEN list and move it to the CLOSED list. Also capture the information of the parent node
- If N is a GOAL node, then move the node to the Closed list and exit the loop returning 'True'. The solution can be found by backtracking the path
- If N is not the GOAL node, expand node N to generate the 'immediate' next nodes linked to node N and add all those to the OPEN list

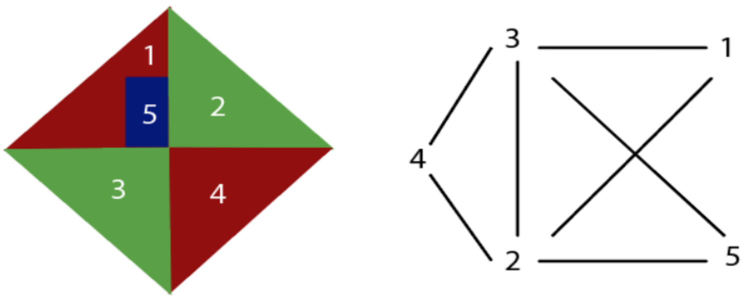
- Reorder the nodes in the OPEN list in ascending order according to an evaluation function f(n)

Constraint Satisfaction Problem (CSP)

- A constraint satisfaction problem (CSP) is a problem that requires its solution within some limitations or conditions also known as constraints. It consists of the following:
- A finite set of variables which stores the solution (V = {V1, V2, V3,....., Vn})
- A set of discrete values known as domain from which the solution is picked (D = {D1, D2, D3,.....,Dn})
- A finite set of constraints (C = {C1, C2, C3,......, Cn})

Constraint Satisfaction Problem (CSP) – Examples

Graph Coloring: The problem where the constraint is that no adjacent sides can have the same color.



Sudoku Playing: The gameplay where the constraint is that no number from 0-9 can be repeated in the same row or column.

SUDOKU

4							5	9
2	6		5				3	
				9	2			
		2		6			1	
		3	8	1	9	7		
	7			3		5		
			3	4				
	3				6		2	7
5	9							6

Puzzle

4	1	7	6	8	3	2	5	9
2	6	9	5	7	1	8	3	4
3	8	5	4	9	2	6	7	1
8	4	2	7	6	5	9	1	3
6	5	3	8	1	9	7	4	2
9	7	1	2	3	4	5	6	8
7	2	6	3	4	8	1	9	5
1	3	8	9	5	6	4	2	7
5	9	4	1	2	7	3	8	6

Solution

Crossword: In crossword problem, the constraint is that there should be the correct formation of the words, and it should be meaningful.

Intelligent Agents

- An intelligent agent (IA) is an entity that makes a decision, that enables artificial intelligence to be put into action.
- It can also be described as a software entity that conducts operations in the place of users or programs after sensing the environment.
- It uses actuators to initiate action in that environment.
- This agent has some level of autonomy that allows it to perform specific, predictable, and repetitive tasks for users or applications.
- It’s also termed as ‘intelligent’ because of its ability to learn during the process of performing tasks.
- The two main functions of intelligent agents include perception and action.
- Perception is done through sensors while actions are initiated through actuators.
- Intelligent agents consist of sub-agents that form a hierarchical structure. Lower-level tasks are performed by these sub-agents.
- The higher-level agents and lower-level agents form a complete system that can solve difficult problems through intelligent behaviors or responses.

Characteristics of intelligent agents

Intelligent agents have the following distinguishing characteristics:

- They have some level of autonomy that allows them to perform certain tasks on their own.
- They have a learning ability that enables them to learn even as tasks are carried out.
- They can interact with other entities such as agents, humans, and systems.
- New rules can be accommodated by intelligent agents incrementally.
- They exhibit goal-oriented habits.
- They are knowledge-based. They use knowledge regarding communications, processes, and entities.

Structure of Intelligent Agents

The Intelligent Agents structure consists of three main parts:

1. Architecture: This refers to machinery or devices that consists of actuators and sensors. The intelligent agent executes on this machinery. Examples include a personal computer, a car, or a camera.

2. Agent function: This is a function in which actions are mapped from a certain percept sequence. Percept sequence refers to a history of what the intelligent agent has perceived.

3. Agent program: This is an implementation or execution of the agent function. The agent function is produced through the agent program's execution on the physical architecture.

Categories of Intelligent Agents

There are 5 main categories of intelligent agents. The grouping of these agents is based on their capabilities and level of perceived intelligence.

1. Simple reflex agents: These agents perform actions using the current percept, rather than the percept history. The condition-action rule is used as the basis for the agent function. In this category, a fully observable environment is ideal for the success of the agent function.

2. Model-based reflex agents: Unlike simple reflex agents, model-based reflex agents consider the percept history in their actions. The agent function can still work well even in an environment that is not fully observable. These agents use an internal model that determines the percept history and effect of actions.

3. Goal-based agents: These agents have higher capabilities than model-based reflex agents. Goal-based agents use goal information to describe desirable capabilities. This allows them to choose among various possibilities. These agents select the best action that enhances the attainment of the goal.

4. Utility-based agents: These agents make choices based on utility. They are more advanced than goal-based agents because of an extra component of utility measurement. Using a utility function, a state is mapped against a certain measure of utility. A rational agent selects the action that optimizes the expected utility of the outcome.

5. Learning agents: These are agents that have the capability of learning from their previous experience.

Learning agents have the following elements.

- **The learning element**: This element enables learning agents to learn from previous experiences.
- **The critic**: It provides feedback on how the agent is doing.
- **The performance element**: This element decides on the external action that needs to be taken.
- **The problem generator**: This acts as a feedback agent that performs certain tasks such as making suggestions (new) and keeping history.

How intelligent agents work

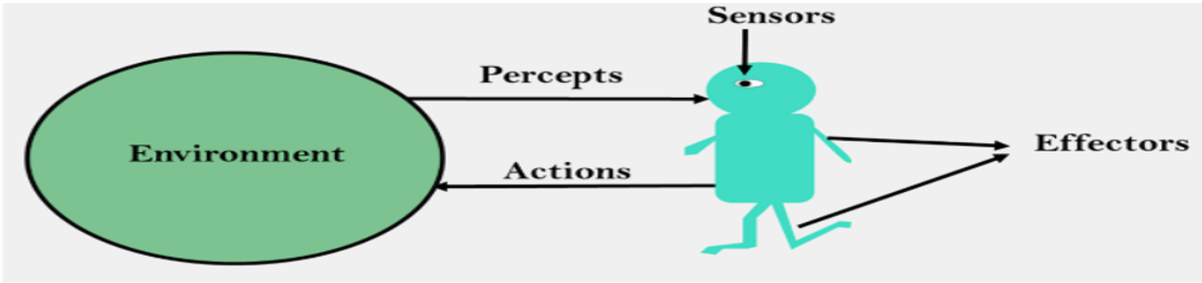
Intelligent agents work through three main components: sensors, actuators, and effectors. Getting an overview of these components can improve our understanding of how intelligent agents work.

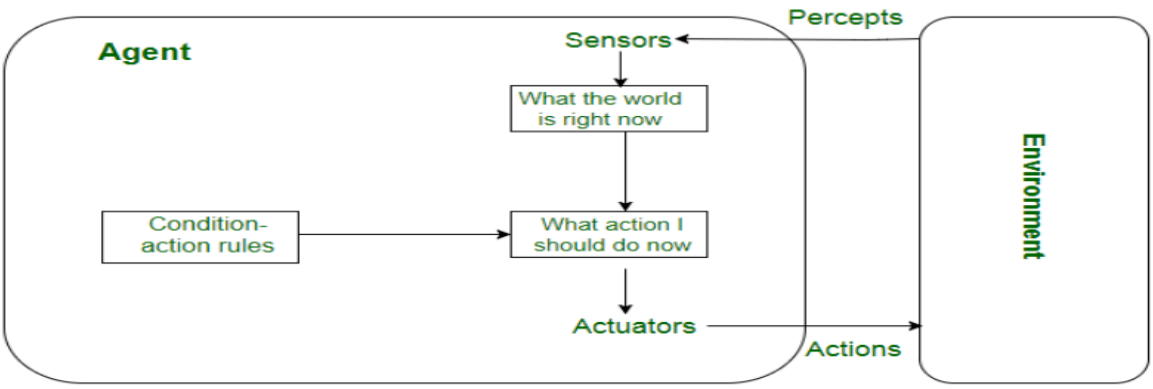
1.Sensors: These are devices that detect any changes in the environment. This information is sent to other devices. In artificial intelligence, the environment of the system is observed by intelligent agents through sensors.

2. Actuators: These are components through which energy is converted into motion. They perform the role of controlling and moving a system. Examples include rails, motors, and gears.

3.Effectors: The environment is affected by effectors. Examples include legs, fingers, wheels, display screen, and arms.

- Inputs (percepts) from the environment are received by the intelligent agent through sensors.
- This agent uses artificial intelligence to make decisions using the acquired information/ observations.
- Actions are then triggered through actuators.
- Future decisions will be influenced by percept history and past actions.





Applications of Intelligent Agents

1. Information search, retrieval, and navigation

- Intelligent agents enhance the access and navigation of information.
- This is achieved through the search of information using search engines.
- The internet consists of many data objects that may take users a lot of time to search for a specific data object.
- Intelligent agents perform this task on behalf of users within a short time.

2. Repetitive office activities

- Some companies have automated certain administrative tasks to reduce operating costs.

- Some of the functional areas that have been automated include customer support and sales.
- Intelligent agents have also been used to enhance office productivity.

3. Medical diagnosis

- Intelligent agents have also been applied in healthcare services to improve the health of patients.
- In this case, the patient is considered as the environment. The computer keyboard is used as the sensor that receives data on the symptoms of the patient.
- The intelligent agent uses this information to decide the best course of action. Medical care is given through actuators such as tests and treatments.

4. Vacuum cleaning

- AI agents are also used to enhance efficiency and cleanness in vacuum cleaning.
- In this case, the environment can be a room, table, or carpet. Some of the sensors employed in vacuum cleaning include cameras, bump sensors, and dirt detection sensors.
- Action is initiated by actuators such as brushes, wheels, and vacuum extractors.

5. Autonomous driving

- Intelligent agents enhance the operation of self-driving cars. In autonomous driving, various sensors are employed to collect information from the environment.
- These include cameras, GPS, and radar. In this application, the environment can be pedestrians, other vehicles, roads, or road signs.
- Various actuators are used to initiate actions. For example, brakes are used to bring the car to a stop.