ARTIFICIAL INTELLIGENCE

According to the father of Artificial Intelligence, John McCarthy, it is *“The science and engineering of making intelligent machines, especially intelligent computer programs”.*

- Artificial Intelligence is a way of **making a computer, a computer-controlled robot, or a software think intelligently**, in the similar manner the intelligent humans think.
- "It is a branch of computer science by which we can create intelligent machines which can behave like a human, think like humans, and able to make decisions."

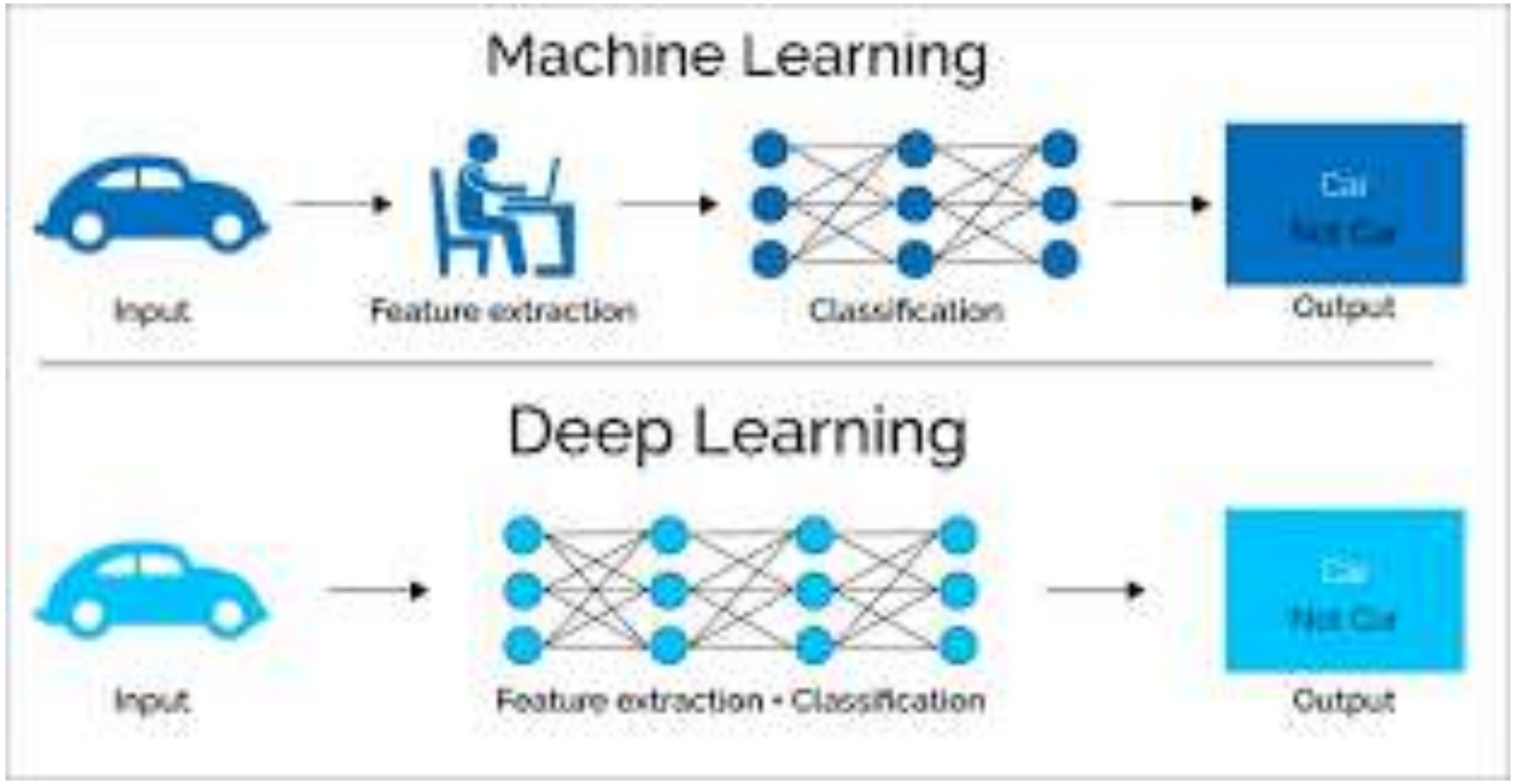
Philosophy of AI

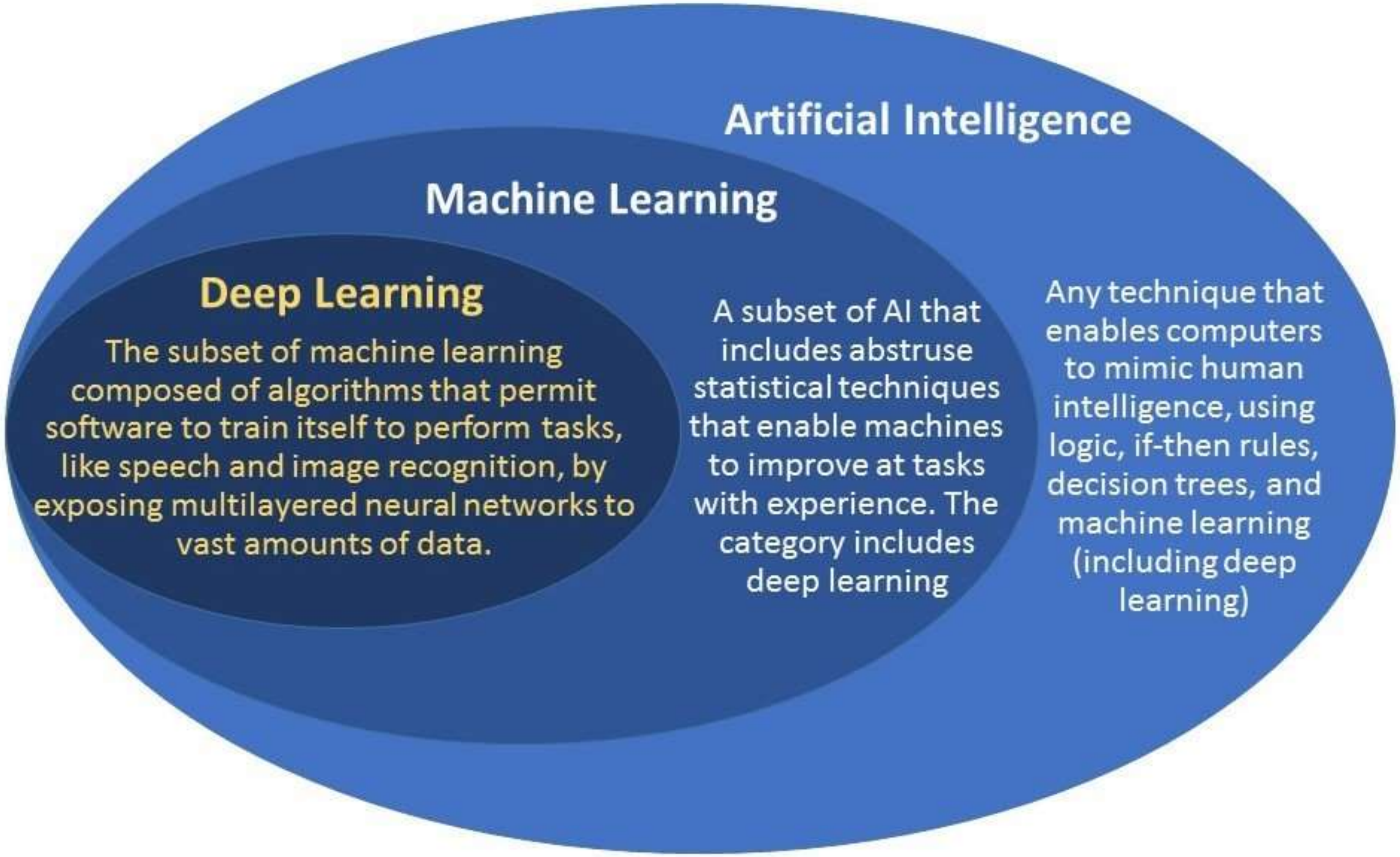
- While exploiting the power of the computer systems, the curiosity of human, lead him to wonder, *“Can a machine think and behave like humans do?”*

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

Goals of AI

- **To Create Expert Systems** − The systems which exhibit intelligent behavior, learn, demonstrate, explain, and advice its users.
- **To Implement Human Intelligence in Machines** − Creating systems that understand, think, learn, and behave like humans.

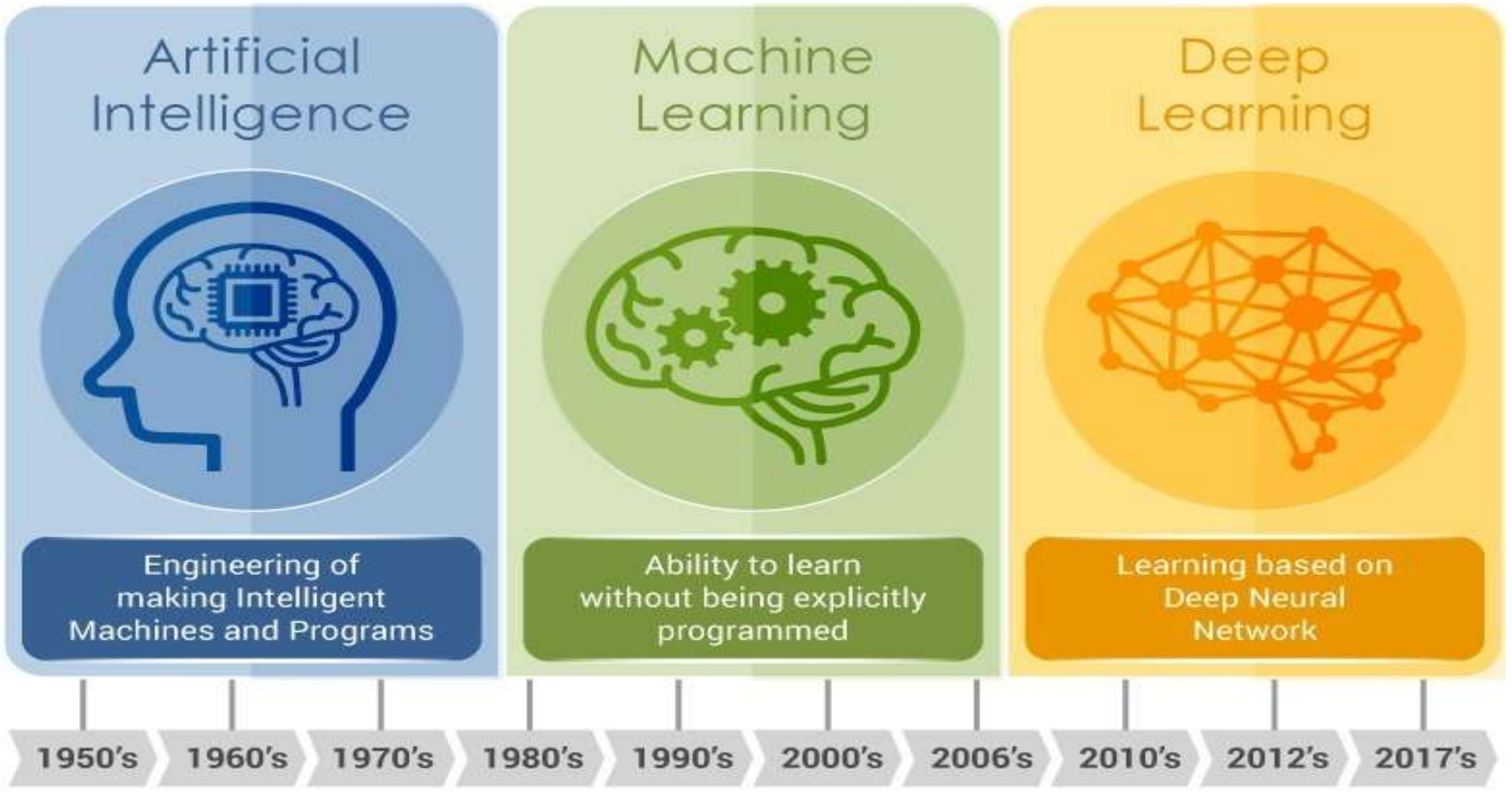

Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction + Classification
Car
Not Car
Output



The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.

A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning

Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)


Artificial Intelligence
Engineering of making Intelligent Machines and Programs
Machine Learning
Ability to learn without being explicitly programmed
Deep Learning
Learning based on Deep Neural Network
1950's
1960's
1970's
1980's
1990's
2000's
2006's
2010's
2012's
2017's

What is Intelligence?

- The ability of a system to calculate, reason, perceive relationships and analogies, learn from experience, store and retrieve information from memory, solve problems, comprehend complex ideas, use natural language fluently, classify, generalize, and adapt new situations.

Types of Intelligence

Intelligence	Description	Example
Linguistic intelligence	The ability to speak, recognize, and use mechanisms of phonology (speech sounds), syntax (grammar), and semantics (meaning).	Narrators, Orators

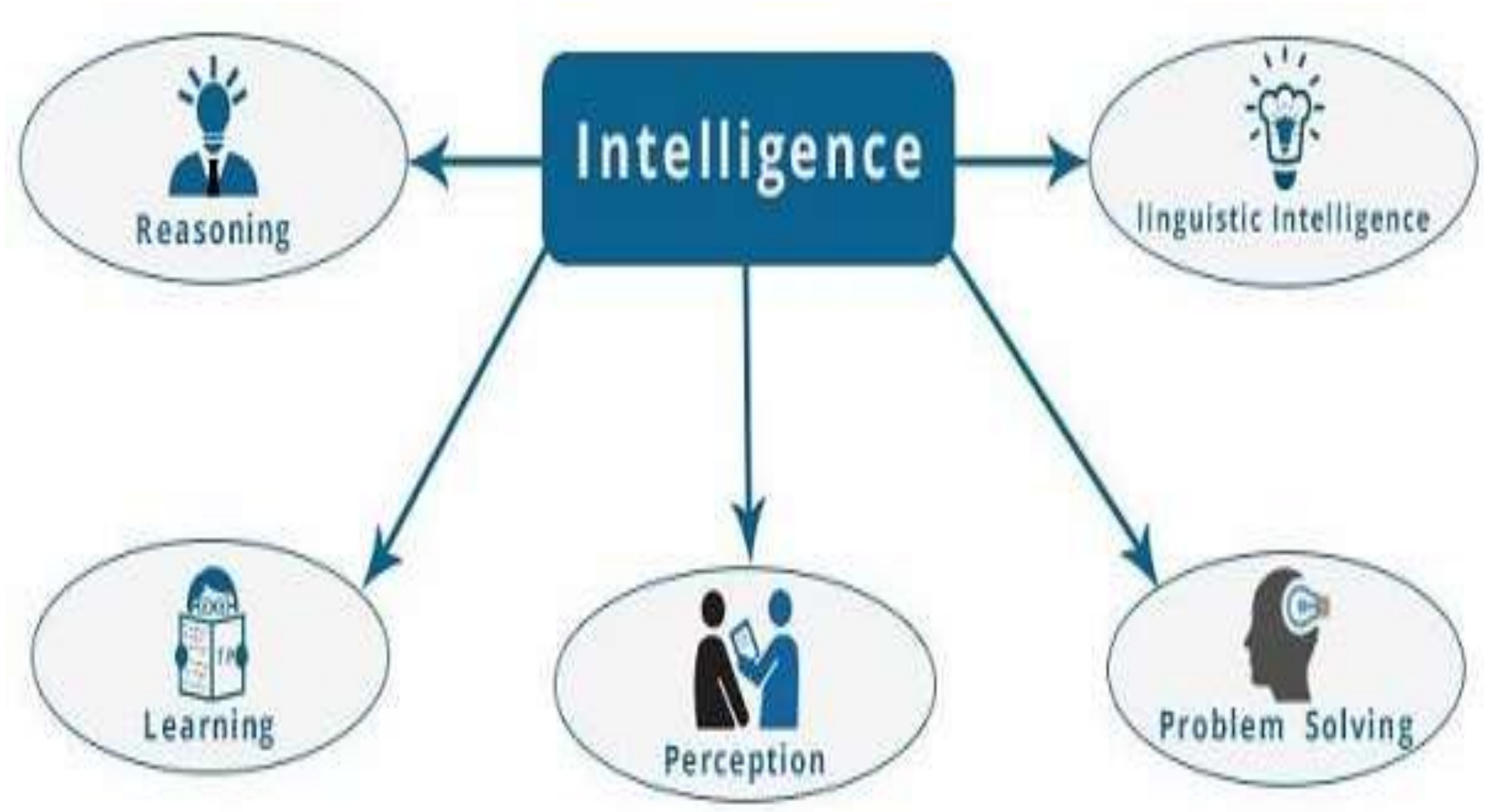
Musical intelligence	The ability to create, communicate with, and understand meanings made of sound, understanding of pitch, rhythm.	Musicians, Singers, Composers
Logical-mathematical intelligence	The ability of use and understand relationships in the absence of action or objects. Understanding complex and abstract ideas.	Mathematicians, Scientists
Spatial intelligence	The ability to perceive visual or spatial information, change it, and re-create visual images without reference to the objects, construct 3D images, and to move and rotate them.	Map readers, Astronauts, Physicists

Intra-personal intelligence	The ability to distinguish among one’s own feelings, intentions, and motivations.	Gautam Buddhha
Interpersonal intelligence	The ability to recognize and make distinctions among other people’s feelings, beliefs, and intentions.	Mass Communicators, Interviewers

What is Intelligence Composed of?

The intelligence is intangible. It is composed of −

- Reasoning
- Learning
- Problem Solving
- Perception
- Linguistic Intelligence



Reasoning – It is the set of processes that enables us to provide basis for judgement, making decisions, and prediction. There are broadly two types

Inductive Reasoning	Deductive Reasoning
It conducts specific observations to makes broad general statements.	It starts with a general statement and examines the possibilities to reach a specific, logical conclusion.
Even if all of the premises are true in a statement, inductive reasoning allows for the conclusion to be false.	If something is true of a class of things in general, it is also true for all members of that class.
Example – "Nita is a teacher. Nita is studious. Therefore, All teachers are studious."	Example – "All women of age above 60 years are grandmothers. Shalini is 65 years. Therefore, Shalini is a grandmother."

Learning − It is the activity of gaining knowledge or skill by studying, practising, being taught, or experiencing something. Learning enhances the awareness of the subjects of the study.

The ability of learning is possessed by humans, some animals, and AI-enabled systems. Learning is categorized as −

- **Auditory Learning** − It is learning by listening and hearing. For example, students listening to recorded audio lectures.
- **Episodic Learning** − To learn by remembering sequences of events that one has witnessed or experienced. This is linear and orderly.
- **Motor Learning** − It is learning by precise movement of muscles. For example, picking objects, Writing, etc.
- **Observational Learning** − To learn by watching and imitating others. For example, child tries to learn by mimicking her parent.

- **Perceptual Learning** − It is learning to recognize stimuli that one has seen before. For example, identifying and classifying objects and situations.
- **Relational Learning** − It involves learning to differentiate among various stimuli on the basis of relational properties, rather than absolute properties. For Example, Adding ‘little less’ salt at the time of cooking potatoes that came up salty last time, when cooked with adding say a tablespoon of salt.
- **Spatial Learning** − It is learning through visual stimuli such as images, colors, maps, etc. For Example, A person can create roadmap in mind before actually following the road.
- **Stimulus-Response Learning** − It is learning to perform a particular behavior when a certain stimulus is present. For example, a dog raises its ear on hearing doorbell.

- **Problem Solving** − It is the process in which one perceives and tries to arrive at a desired solution from a present situation by taking some path, which is blocked by known or unknown hurdles.
- Problem solving also includes **decision making**, which is the process of selecting the best suitable alternative out of multiple alternatives to reach the desired goal are available.
- **Perception** − It is the process of acquiring, interpreting, selecting, and organizing sensory information.
- Perception presumes **sensing**. In humans, perception is aided by sensory organs. In the domain of AI, perception mechanism puts the data acquired by the sensors together in a meaningful manner.
- **Linguistic Intelligence** − It is one’s ability to use, comprehend, speak, and write the verbal and written language. It is important in interpersonal communication.

Difference between Human and Machine Intelligence

- Humans perceive by patterns whereas the machines perceive by set of rules and data.
- Humans store and recall information by patterns, machines do it by searching algorithms. For example, the number 40404040 is easy to remember, store, and recall as its pattern is simple.
- Humans can figure out the complete object even if some part of it is missing or distorted; whereas the machines cannot do it correctly.

Programming Without and With AI

The programming without and with AI is different in following ways −

Programming Without AI	Programming With AI
A computer program without AI can answer the **specific** questions it is meant to solve.	A computer program with AI can answer the **generic** questions it is meant to solve.
Modification in the program leads to change in its structure.	AI programs can absorb new modifications by putting highly independent pieces of information together. Hence you can modify even a minute piece of information of program without affecting its structure.
Modification is not quick and easy. It may lead to affecting the program adversely.	Quick and Easy program modification.

What is AI Technique?

- In the real world, the knowledge has some unwelcomed properties –
- Its volume is huge, next to unimaginable.
- It is not well-organized or well-formatted.
- It keeps changing constantly.

AI Technique is a manner to organize and use the knowledge efficiently in such a way that –

- It should be perceivable by the people who provide it.
- It should be easily modifiable to correct errors.
- It should be useful in many situations though it is incomplete or inaccurate.
- AI techniques elevate the speed of execution of the complex program it is equipped with.

Applications of AI

- AI has been dominant in various fields such as −
- **Gaming** − AI plays crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machine can think of large number of possible positions based on heuristic knowledge.
- **Natural Language Processing** − It is possible to interact with the computer that understands natural language spoken by humans.
- **Expert Systems** − There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanation and advice to the users.
- **Intelligent Robots** − Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment.

- **Vision Systems** − These systems understand, interpret, and comprehend visual input on the computer. For example,
  - A spying aeroplane takes photographs, which are used to figure out spatial information or map of the areas.
  - Doctors use clinical expert system to diagnose the patient.
  - Police use computer software that can recognize the face of criminal with the stored portrait made by forensic artist.
- **Speech Recognition** − Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it. It can handle different accents, slang words, noise in the background, change in human's noise due to cold, etc.
- **Handwriting Recognition** − The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus. It can recognize the shapes of the letters and convert it into editable text.

EXAMPLES OF ARTIFICIAL INTELLIGENCE

- Siri, Alexa and other smart assistants
- Self-driving cars
- Robo-advisors
- Conversational bots
- Email spam filters
- Netflix's recommendations

Why Artificial Intelligence?

- With the help of AI, you can create such software or devices which can solve real-world problems very easily and with accuracy such as health issues, marketing, traffic issues, etc.
- With the help of AI, you can create your personal virtual Assistant, such as Cortana, Google Assistant, Siri, etc.
- With the help of AI, you can build such Robots which can work in an environment where survival of humans can be at risk.
- AI opens a path for other new technologies, new devices, and new Opportunities.

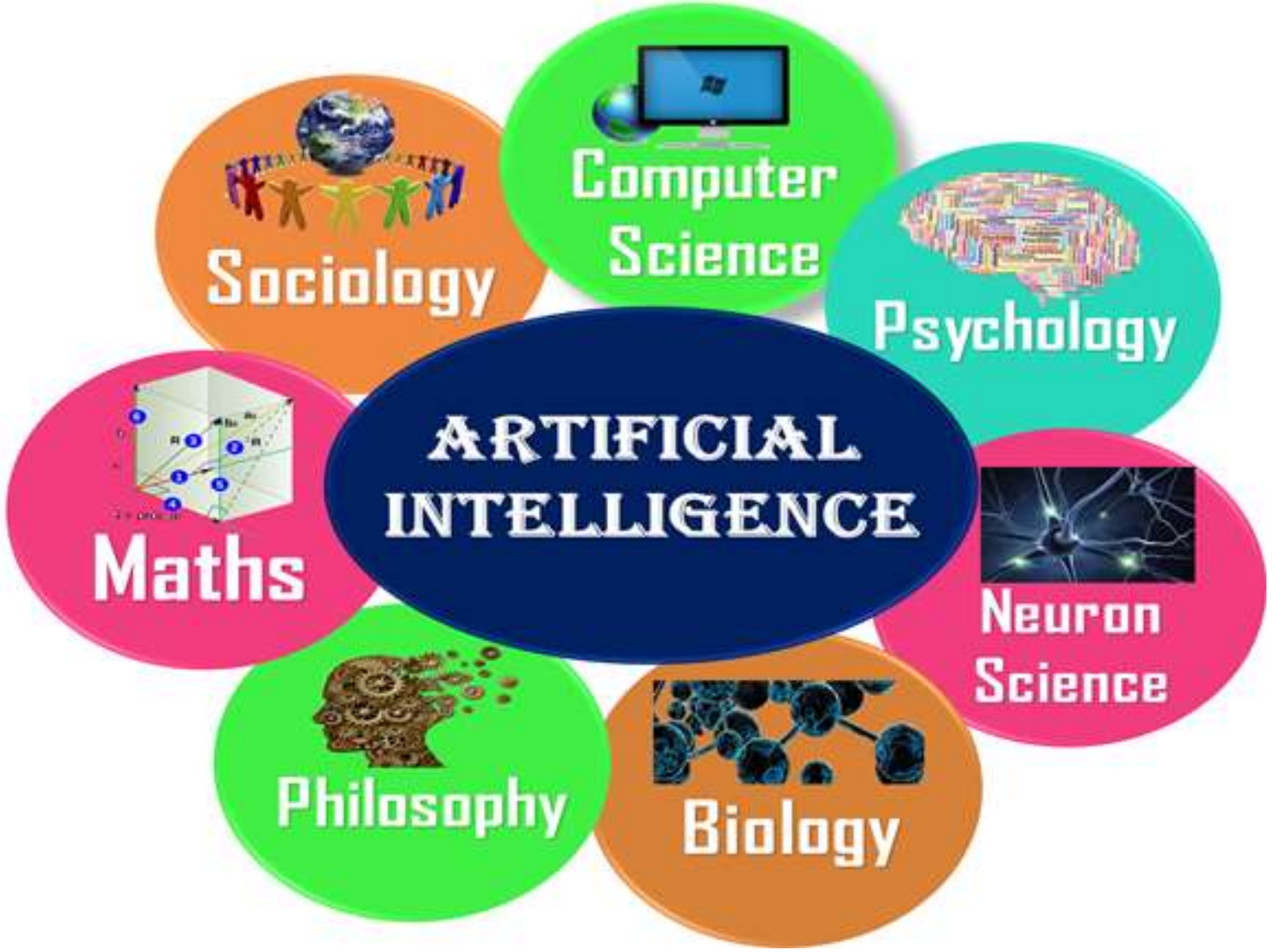
Goals of Artificial Intelligence

Following are the main goals of Artificial Intelligence:

- Replicate human intelligence
- Solve Knowledge-intensive tasks
- An intelligent connection of perception and action
- Building a machine which can perform tasks that requires human intelligence such as:
  - Proving a theorem
  - Playing chess
  - Plan some surgical operation
  - Driving a car in traffic
- Creating some system which can exhibit intelligent behavior, learn new things by itself, demonstrate, explain, and can advise to its user.

What Comprises to Artificial Intelligence?

- Artificial Intelligence is not just a part of computer science even it's so vast and requires lots of other factors which can contribute to it.
- To create the AI first we should know that how intelligence is composed, so the Intelligence is an intangible part of our brain which is a combination of **Reasoning, learning, problem-solving perception, language understanding, etc**.
- To achieve the above factors for a machine or software Artificial Intelligence requires the following discipline:
- Mathematics
- Biology
- Psychology
- Sociology
- Computer Science
- Neurons Study
- Statistics
-


ARTIFICIAL INTELLIGENCE
Sociology
Computer Science
Psychology
Maths
Neuron Science
Philosophy
Biology

Advantages of Artificial Intelligence

- **High Accuracy with less errors:** AI machines or systems are prone to less errors and high accuracy as it takes decisions as per pre-experience or information.
- **High-Speed:** AI systems can be of very high-speed and fast-decision making, because of that AI systems can beat a chess champion in the Chess game.
- **High reliability:** AI machines are highly reliable and can perform the same action multiple times with high accuracy.
- **Useful for risky areas:** AI machines can be helpful in situations such as defusing a bomb, exploring the ocean floor, where to employ a human can be risky.
- **Digital Assistant:** AI can be very useful to provide digital assistant to the users such as AI technology is currently used by various E-commerce websites to show the products as per customer requirement.
- **Useful as a public utility:** AI can be very useful for public utilities such as a self-driving car which can make our journey safer and hassle-free, facial recognition for security purpose, Natural language processing to communicate with the human in human-language, etc.

Disadvantages of Artificial Intelligence

- **High Cost:** The hardware and software requirement of AI is very costly as it requires lots of maintenance to meet current world requirements.
- **Can't think out of the box:** Even we are making smarter machines with AI, but still they cannot work out of the box, as the robot will only do that work for which they are trained, or programmed.
- **No feelings and emotions:** AI machines can be an outstanding performer, but still it does not have the feeling so it cannot make any kind of emotional attachment with human, and may sometime be harmful for users if the proper care is not taken.
- **Increase dependency on machines:** With the increment of technology, people are getting more dependent on devices and hence they are losing their mental capabilities.
- **No Original Creativity:** As humans are so creative and can imagine some new ideas but still AI machines cannot beat this power of human intelligence and cannot be creative and imaginative.

AI Approaches and Concepts

- Less than a decade after breaking the Nazi encryption machine Enigma and helping the Allied Forces win World War II, mathematician Alan Turing changed history a second time with a simple question: "Can machines think?"
- Turing's paper "Computing Machinery and Intelligence" (1950), and its subsequent Turing Test, established the fundamental goal and vision of artificial intelligence.
- At its core, AI is the branch of computer science that aims to answer Turing's question in the affirmative. It is the endeavor to replicate or simulate human intelligence in machines.
- The expansive goal of artificial intelligence has given rise to many questions and debates. So much so, that no singular definition of the field is universally accepted.

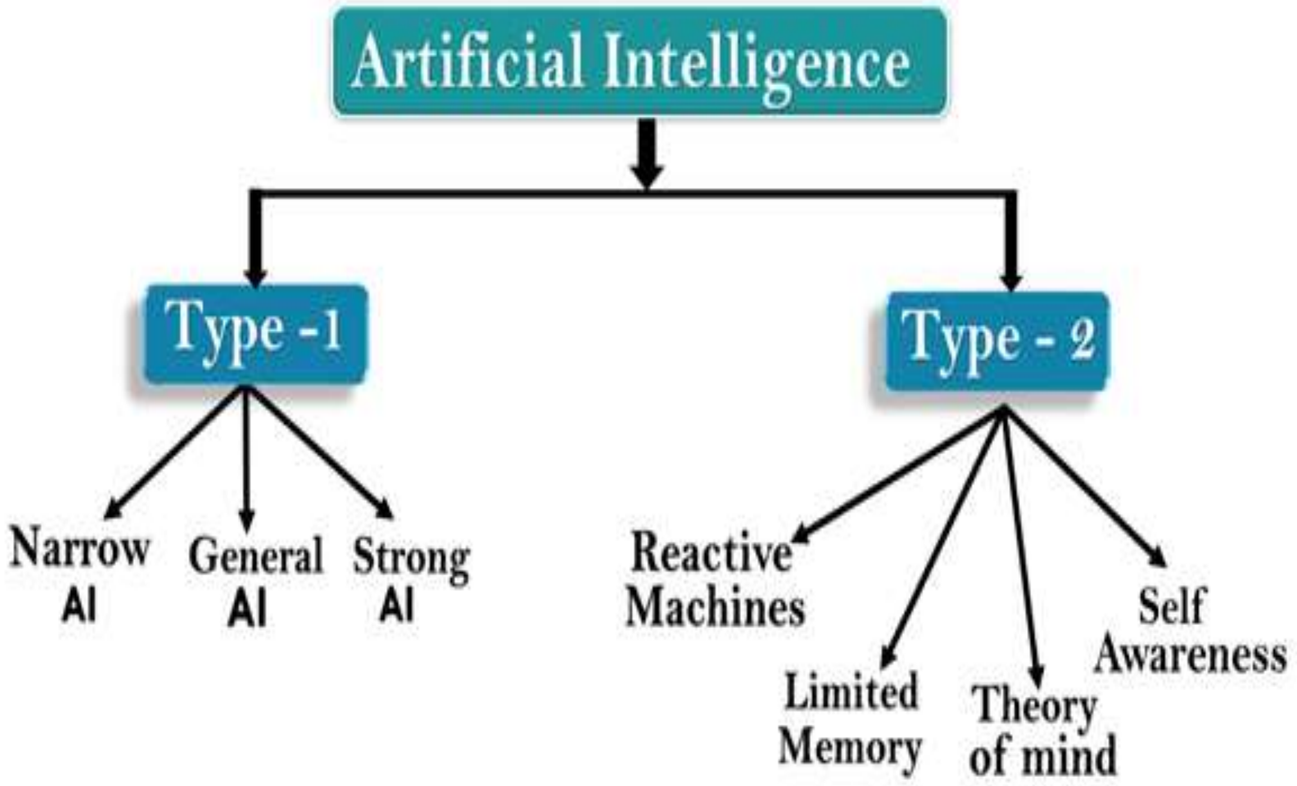
- The major limitation in defining AI as simply "building machines that are intelligent" is that it doesn't actually explain *what artificial intelligence is? What makes a machine intelligent?* AI is an interdisciplinary science with multiple approaches, but advancements in machine learning and deep learning are creating a paradigm shift in virtually every sector of the tech industry.
- In their groundbreaking textbook *Artificial Intelligence: A Modern Approach*, authors Stuart Russell and Peter Norvig approach the question by unifying their work around the theme of intelligent agents in machines. With this in mind, AI is "the study of agents that receive percepts from the environment and perform actions."

- Norvig and Russell go on to explore four different approaches that have historically defined the field of AI:
- **Thinking humanly**
- **Thinking rationally**
- **Acting humanly**
- **Acting rationally**

- The first two ideas concern thought processes and reasoning, while the others deal with behavior. Norvig and Russell focus particularly on rational agents that act to achieve the best outcome, noting "all the skills needed for the Turing Test also allow an agent to act rationally.“
- Patrick Winston, the Ford professor of artificial intelligence and computer science at MIT, defines AI as "algorithms enabled by constraints, exposed by representations that support models targeted at loops that tie thinking, perception and action together."

- While these definitions may seem abstract to the average person, they help focus the field as an area of computer science and provide a blueprint for infusing machines and programs with machine learning and other subsets of artificial intelligence.

Types of Artificial Intelligence:



AI type-1: Based on Capabilities

1. Weak AI or Narrow AI:

- Narrow AI is a type of AI which is able to perform a dedicated task with intelligence.The most common and currently available AI is Narrow AI in the world of Artificial Intelligence.
- Narrow AI cannot perform beyond its field or limitations, as it is only trained for one specific task. Hence it is also termed as weak AI. Narrow AI can fail in unpredictable ways if it goes beyond its limits.
- Apple Siriis a good example of Narrow AI, but it operates with a limited pre-defined range of functions.
- IBM's Watson supercomputer also comes under Narrow AI, as it uses an Expert system approach combined with Machine learning and natural language processing.
- Some Examples of Narrow AI are playing chess, purchasing suggestions on e-commerce site, self-driving cars, speech recognition, and image recognition.

EXAMPLES OF NARROW AI

- Google search
- Image recognition software
- Siri, Alexa and other personal assistants
- Self-driving cars
- IBM's Watson

General AI:

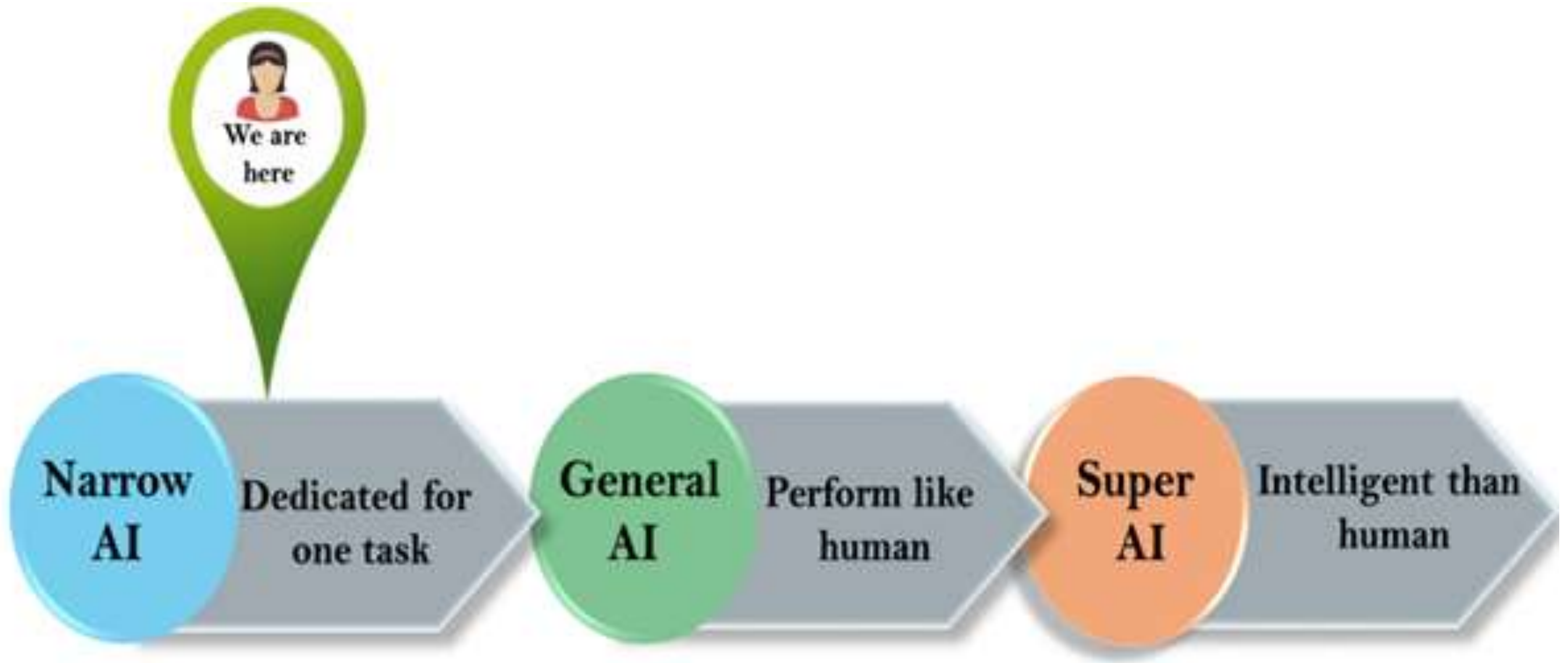
- General AI is a type of intelligence which could perform any intellectual task with efficiency like a human.
- The idea behind the general AI to make such a system which could be smarter and think like a human by its own.
- Currently, there is no such system exist which could come under general AI and can perform any task as perfect as a human.
- The worldwide researchers are now focused on developing machines with General AI.
- As systems with general AI are still under research, and it will take lots of efforts and time to develop such systems.

Super AI:

- Super AI is a level of Intelligence of Systems at which machines could surpass human intelligence, and can perform any task better than human with cognitive properties. It is an outcome of general AI.

- Some key characteristics of strong AI include capability include the ability to think, to reason,solve the puzzle, make judgments, plan, learn, and communicate by its own.

- Super AI is still a hypothetical concept of Artificial Intelligence. Development of such systems in real is still world changing task.


We are here
Narrow AI
Dedicated for one task
General AI
Perform like human
Super AI
Intelligent than human

Artificial Intelligence type-2: Based on functionality

1. Reactive Machines

- Purely reactive machines are the most basic types of Artificial Intelligence.
- Such AI systems do not store memories or past experiences for future actions.
- These machines only focus on current scenarios and react on it as per possible best action.
- IBM's Deep Blue system is an example of reactive machines.
- Google's AlphaGo is also an example of reactive machines.
- A famous example of a reactive machine is Deep Blue, which was designed by IBM in the 1990’s as a chess-playing supercomputer and defeated international grandmaster Gary Kasparov in a game.
- Deep Blue was only capable of identifying the pieces on a chess board and knowing how each moves based on the rules of chess, acknowledging each piece’s present position, and determining what the most logical move would be at that moment

2. Limited Memory

- Limited memory machines can store past experiences or some data for a short period of time.
- These machines can use stored data for a limited time period only.
- Self-driving cars are one of the best examples of Limited Memory systems. These cars can store recent speed of nearby cars, the distance of other cars, speed limit, and other information to navigate the road.
- Limited memory AI is created when a team continuously trains a model in how to analyze and utilize new data or an AI environment is built so models can be automatically trained and renewed.
- When utilizing limited memory AI in machine learning, six steps must be followed: Training data must be created, the machine learning model must be created, the model must be able to make predictions, the model must be able to receive human or environmental feedback, that feedback must be stored as data, and these these steps must be reiterated as a cycle.

- There are three major machine learning models that utilize limited memory artificial intelligence:
- **Reinforcement learning**, which learns to make better predictions through repeated trial-and-error.
- **Long Short Term Memory (LSTM)**, which utilizes past data to help predict the next item in a sequence. LTSMs view more recent information as most important when making predictions and discounts data from further in the past, though still utilizing it to form conclusions
- **Evolutionary Generative Adversarial Networks (E-GAN)**, which evolves over time, growing to explore slightly modified paths based off of previous experiences with every new decision. This model is constantly in pursuit of a better path and utilizes simulations and statistics, or chance, to predict outcomes throughout its evolutionary mutation cycle.
-

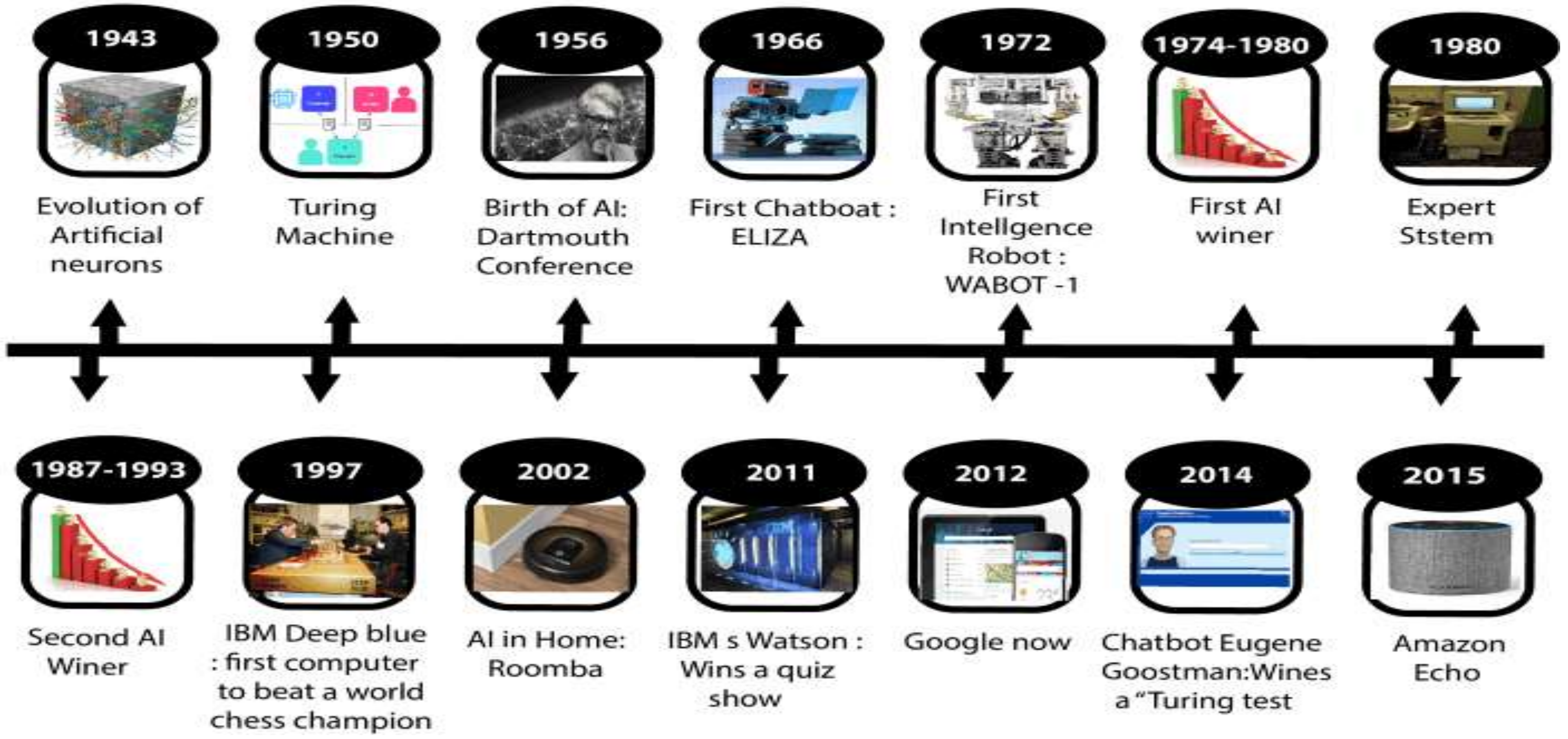
3. Theory of Mind

- Theory of Mind AI should understand the human emotions, people, beliefs, and be able to interact socially like humans.
- This type of AI machines are still not developed, but researchers are making lots of efforts and improvement for developing such AI machines.
- Theory of Mind is just that — theoretical. We have not yet achieved the technological and scientific capabilities necessary to reach this next level of artificial intelligence.

4. Self-Awareness

- Self-awareness AI is the future of Artificial Intelligence. These machines will be super intelligent, and will have their own consciousness, sentiments, and self-awareness.
- These machines will be smarter than human mind.
- Self-Awareness AI does not exist in reality still and it is a hypothetical concept.

History of AI


1943
Evolution of Artificial neurons
1950
Turing Machine
1956
Birth of AI: Dartmouth Conference
1966
First Chatboat : ELIZA
1972
First Intellgence Robot : WABOT -1
1974-1980
First AI winer
1980
Expert Ststem
1987-1993
Second AI Winer
1997
IBM Deep blue : first computer to beat a world chess champion
2002
AI in Home: Roomba
2011
IBM s Watson : Wins a quiz show
2012
Google now
2014
Chatbot Eugene Goostman:Wines a "Turing test
2015
Amazon Echo

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 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.
- 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 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.

Search Algorithm Terminologies:

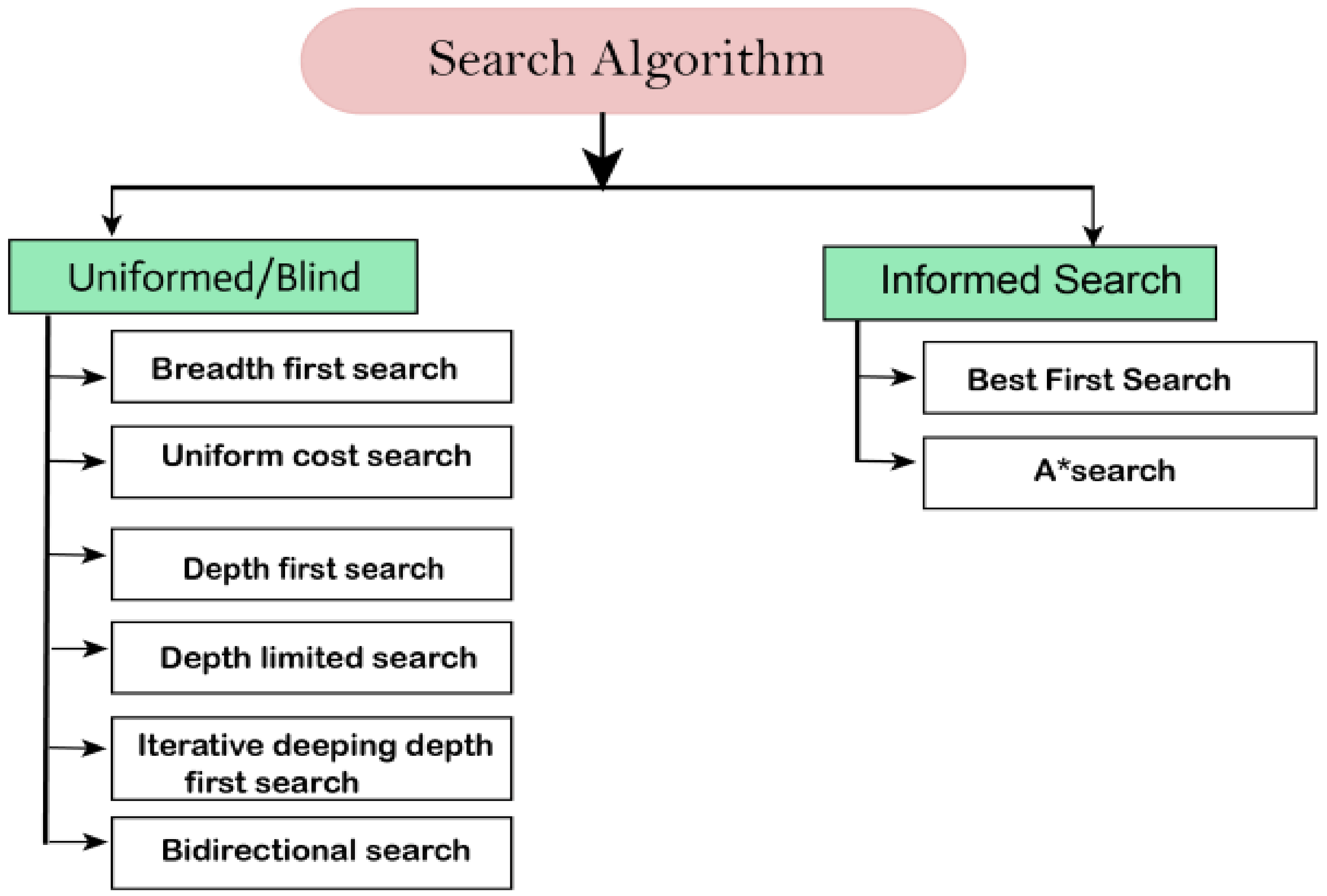
- **Search:** Searching is a step by step procedure to solve a search-problem in a given search space.
- A search problem can have three main factors:
    - **Search Space:** Search space represents a set of possible solutions, which a system may have.
    - **Start State:** It is a state from where agent begins **the search**.
    - **Goal test:** It is a function which observe the current state and returns whether the goal state is achieved or not.

- **Search tree:** A tree representation of search problem is called Search tree. The root of the search tree is the root node which is corresponding to the initial state.
- **Actions:** It gives the description of all the available actions to the agent.
- **Transition model:** A description of what each action do, can be represented as a transition model.
- **Path Cost:** It is a function which assigns a numeric cost to each path.
- **Solution:** It is an action sequence which leads from the start node to the goal node.
- **Optimal Solution:** If a solution has the lowest cost among all solutions

Properties of Search Algorithms:

- Following are the four essential properties of search algorithms to compare the efficiency of these algorithms:
- **Completeness:** A search algorithm is said to be complete if it guarantees to return a solution if at least any solution exists for any random input.
- **Optimality:** If a solution found for an algorithm is guaranteed to be the best solution (lowest path cost) among all other solutions, then such a solution for is said to be an optimal solution.
- **Time Complexity:** Time complexity is a measure of time for an algorithm to complete its task.
- **Space Complexity:** It is the maximum storage space required at any point during the search, as the complexity of the problem.

Types of search algorithms



Uninformed/Blind Search:

- The uninformed search does not contain any domain knowledge such as closeness, the location of the goal. It operates in a brute-force way as it only includes information about how to traverse the tree and how to identify leaf and goal nodes.
- Uninformed search applies a way in which search tree is searched without any information about the search space like initial state operators and test for the goal, so it is also called blind search.It examines each node of the tree until it achieves the goal node.

- **It can be divided into five main types:**
- Breadth-first search
- Uninform cost search
- Depth-first search
- Iterative deepening depth-first search
- Bidirectional Search

Informed Search

- Informed search algorithms use domain knowledge. In an informed search, problem information is available which can guide the search. Informed search strategies can find a solution more efficiently than an uninformed search strategy. Informed search is also called a Heuristic search.
- A heuristic is a way which might not always be guaranteed for best solutions but guaranteed to find a good solution in reasonable time.
- Informed search can solve much complex problem which could not be solved in another way.
- An example of informed search algorithms is a traveling salesman problem.
- Greedy Search
- A* Search

Uninformed Search Algorithms

- Uninformed search is a class of general-purpose search algorithms which operates in brute force-way. Uninformed search algorithms do not have additional information about state or search space other than how to traverse the tree, so it is also called blind search.
- Following are the various types of uninformed search algorithms:
- Breadth-first Search
- Depth-first Search
- Depth-limited Search
- Iterative deepening depth-first search
- Uniform cost search
- Bidirectional Search

Breadth-first Search:

- Breadth-first search is the most common search strategy for traversing a tree or graph. This algorithm searches breadthwise in a tree or graph, so it is called breadth-first search.
- BFS algorithm starts searching from the root node of the tree and expands all successor node at the current level before moving to nodes of next level.
- The breadth-first search algorithm is an example of a general-graph search algorithm.
- Breadth-first search implemented using FIFO queue data structure.

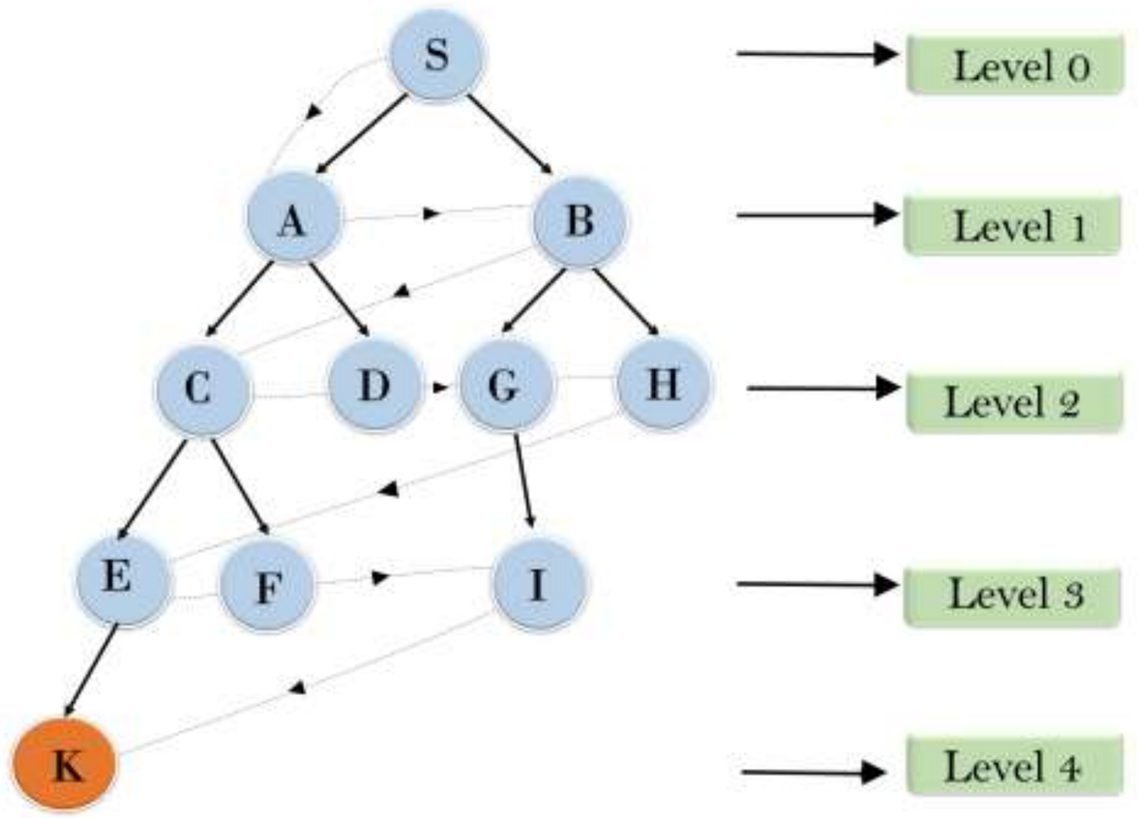
Advantages:

- BFS will provide a solution if any solution exists.
- If there are more than one solutions for a given problem, then BFS will provide the minimal solution which requires the least number of steps.

Disadvantages:

- It requires lots of memory since each level of the tree must be saved into memory to expand the next level.
- BFS needs lots of time if the solution is far away from the root node.
- Example:
- In the below tree structure, we have shown the traversing of the tree using BFS algorithm from the root node S to goal node K. BFS search algorithm traverse in layers, so it will follow the path which is shown by the dotted arrow, and the traversed path will be:
- S---> A--->B---->C--->D---->G--->H--->E---->F---->I---->K

Breadth First Search



- **Time Complexity:** Time Complexity of BFS algorithm can be obtained by the number of nodes traversed in BFS until the shallowest Node. Where the d= depth of shallowest solution and b is a node at every state.
- **T (b) = $1+b^2+b^3+.......+ b^d= O (b^d)$**
- **Space Complexity:** Space complexity of BFS algorithm is given by the Memory size of frontier which is $O(b^d)$.
- **Completeness:** BFS is complete, which means if the shallowest goal node is at some finite depth, then BFS will find a solution.
- **Optimality:** BFS is optimal if path cost is a non-decreasing function of the depth of the node.

2. Depth-first Search

- Depth-first search is a recursive algorithm for traversing a tree or graph data structure.
- It is called the depth-first search because it starts from the root node and follows each path to its greatest depth node before moving to the next path.
- DFS uses a stack data structure for its implementation.
- The process of the DFS algorithm is similar to the BFS algorithm.

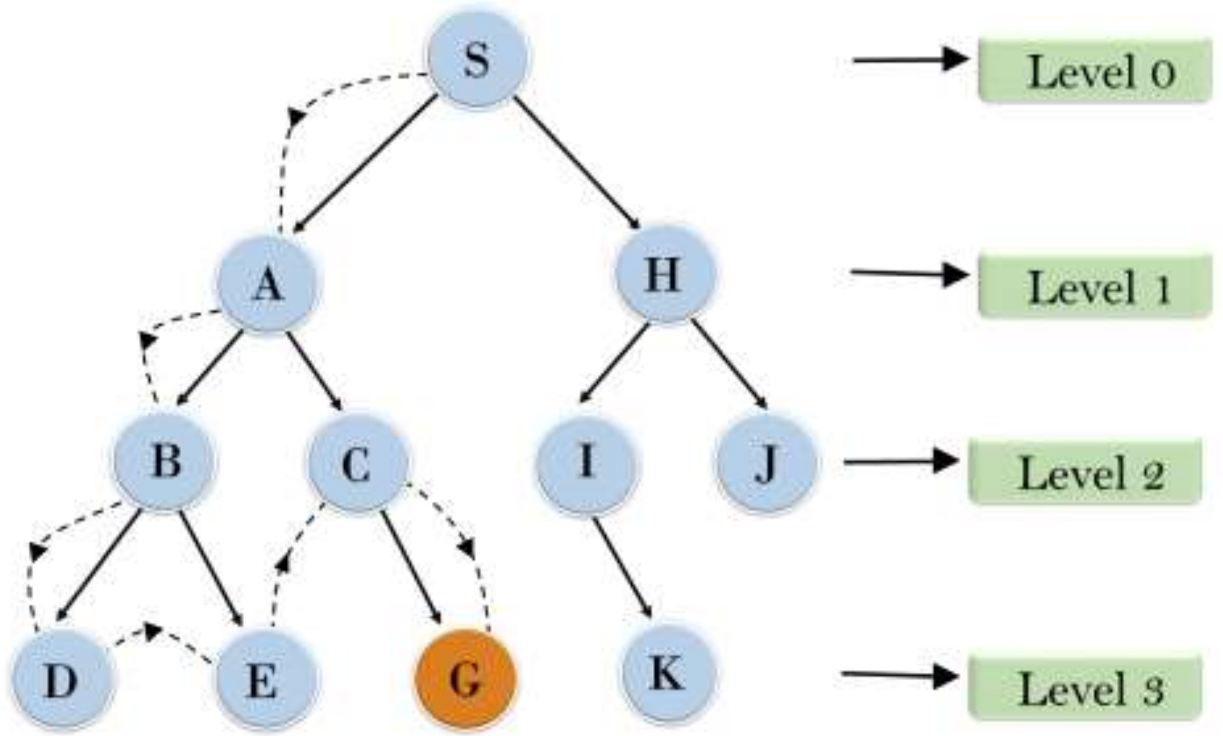
Advantage:

- DFS requires very less memory as it only needs to store a stack of the nodes on the path from root node to the current node.
- It takes less time to reach to the goal node than BFS algorithm (if it traverses in the right path).

Disadvantage:

- There is the possibility that many states keep re-occurring, and there is no guarantee of finding the solution.
- DFS algorithm goes for deep down searching and sometime it may go to the infinite loop.
- Example:
- In the below search tree, we have shown the flow of depth-first search, and it will follow the order as:
- Root node--->Left node ----> right node.
- It will start searching from root node S, and traverse A, then B, then D and E, after traversing E, it will backtrack the tree as E has no other successor and still goal node is not found. After backtracking it will traverse node C and then G, and here it will terminate as it found goal node.

Depth First Search



- **Completeness:** DFS search algorithm is complete within finite state space as it will expand every node within a limited search tree.
- **Time Complexity:** Time complexity of DFS will be equivalent to the node traversed by the algorithm. It is given by:
- **$T(n)= 1+ n^2+ n^3 +.........+ n^m=O(n^m)$**
- **Where, m= maximum depth of any node and this can be much larger than d (Shallowest solution depth)**
- **Space Complexity:** DFS algorithm needs to store only single path from the root node, hence space complexity of DFS is equivalent to the size of the fringe set, which is **O(bm)**.

Production System in AI

- A production system (popularly known as a production rule system) is a kind of cognitive architecture that is used to implement search algorithms and replicate human problem-solving skills. This problem-solving knowledge is encoded in the system in the form of little quanta popularly known as productions. It consists of two components: rule and action.
- Rules recognize the condition, and the actions part has the knowledge of how to deal with the condition. In simpler words, the production system in AI contains a set of rules which are defined by the left side and right side of the system. The left side contains a set of things to watch for (condition), and the right side contains the things to do (action).
- **What are the Elements of a Production System?**
- An AI production system has three main elements which are as follows:

- **Global Database:** The primary database which contains all the information necessary to successfully complete a task.
- It is further broken down into two parts:

  1. temporary

  2.permanent.
- The temporary part contains information relevant to the current situation only whereas the permanent part contains information about the fixed actions.
- **A set of Production Rules:** A set of rules that operates on the global database. Each rule consists of a precondition and postcondition that the global database either meets or not.
- For example, if a condition is met by the global database, then the production rule is applied successfully.
- **Control System:** A control system that acts as the decision-maker, decides which production rule should be applied. The Control system stops computation or processing when a termination condition is met on the database.

Features of a Production System

- **A production system has the following features:**
- **Simplicity:** Due to the use of the IF-THEN structure, each sentence is unique in the production system. This uniqueness makes the knowledge representation simple to enhance the readability of the production rules.
- **Modularity:** The knowledge available is coded in discrete pieces by the production system, which makes it easy to add, modify, or delete the information without any side effects.
- **Modifiability:** This feature allows for the modification of the production rules. The rules are first defined in the skeletal form and then modified to suit an application.
- **Knowledge-intensive:** As the name suggests, the system only stores knowledge. All the rules are written in the English language. This type of representation solves the semantics problem.

Classes of a Production System

A production system is classified into four main classes which are:

- **Monotonic Production System:** In a monotonic production system, the use of one rule never prevents the involvement of another rule when both the rules are selected at the same time. Hence, it enables the system to apply rules simultaneously.
- **Partially Commutative Production System:** In this production system if a set of rules is used to change state A to state B then any allowable combination of these rules will also produce the same results (convert state A to state B).
- **Non-Monotonic Production System:** This production system increases the problem-solving efficiency of the machine by not keeping a record of the changes made in the previous search process. These types of production systems are useful from an implementation point of view as they do not backtrack to the previous state when it is found that an incorrect path was followed.
- **Commutative Production System:** These type of production systems is used when the order of operation is not important, and the changes are reversible.

Advantages of using a Production System

- Offers modularity as all the rules can be added, deleted, or modified individually.
- Separate control system and knowledge base.
- An excellent and feasible model that imitates human problem-solving skills.
- Beneficial in real-time applications and environment.
- Offers language independence.

Advantages & Disadvantages

- Provides **excellent tools** for structuring AI programs
- The system is highly **modular** because individual rules can be added, removed or modified independently
- Separation of **knowledge** and **Control-Recognises Act Cycle**
- A natural **mapping** onto state-space research data or goal-driven
- The system uses pattern directed control which is more **flexible** than algorithmic control
- Provides opportunities for **heuristic control** of the search
- A good way to model the **state-driven nature** of intelligent machines
- Quite helpful in **a real-time** environment and applications.

Disadvantages:

- It is very **difficult** to analyze the flow of control within a production system
- It describes the operations that can be performed in a search for a solution to the problem.
- There is an **absence of learning** due to a rule-based production system that does not store the result of the problem for future use.
- The rules in the production system should not have any type of **conflict resolution** as when a new rule is added to the database it should ensure that it does not have any conflict with any existing rule.
-
-

Production System in Artificial Intelligence: Example

We have two jugs of capacity 5l and 3l (liter), and a tap with an endless supply of water. The objective is to obtain 4 liters exactly in the 5-liter jug with the minimum steps possible.

- **Production System:**
- Fill the 5 liter jug from tap
- Empty the 5 liter jug
- Fill the 3 liter jug from tap
- Empty the 3 liter jug
- Then, empty the 3 liter jug to 5 liter
- Empty the 5 liter jug to 3 liter
- Pour water from 3 liters to 5 liter
- Pour water from 5 liters to 3 liters but do not empty
- **Solution:**
- **1,8,4,6,1,8** or **3,5,3,7,2,5,3,5;**
- It is possible to have other solutions as well but these are the shortest and the 1st sequence should be chosen as it has the minimum number of steps.

Heuristic Search Techniques

- A heuristic is a technique that is used to solve a problem faster than the classic methods.
- Problem Solving method that uses shortcuts/calculated guess to provide good enough solutions.
- These techniques are used to find the approximate solution of a problem when classical methods do not.
- Heuristics are said to be the problem-solving techniques that result in practical and quick solutions.
- Heuristics are strategies that are derived from past experience with similar problems. Heuristics use practical methods and shortcuts used to produce the solutions that may or may not be optimal, but those solutions are sufficient in a given limited timeframe.

Why do we need heuristics?

- One reason is to produce, in a reasonable amount of time, a solution that is good enough for the problem in question.
- It doesn’t have to be the best- an approximate solution will do since this is fast enough.
- Most problems are exponential. Heuristic Search let us reduce this to a rather polynomial number. We use this in AI because we can put it to use in situations where we can’t find known algorithms.
- Heuristics are used in situations in which there is the requirement of a short-term solution. On facing complex situations with limited resources and time, Heuristics can help the companies to make quick decisions by shortcuts and approximated calculations. Most of the heuristic methods involve mental shortcuts to make decisions on past experiences.

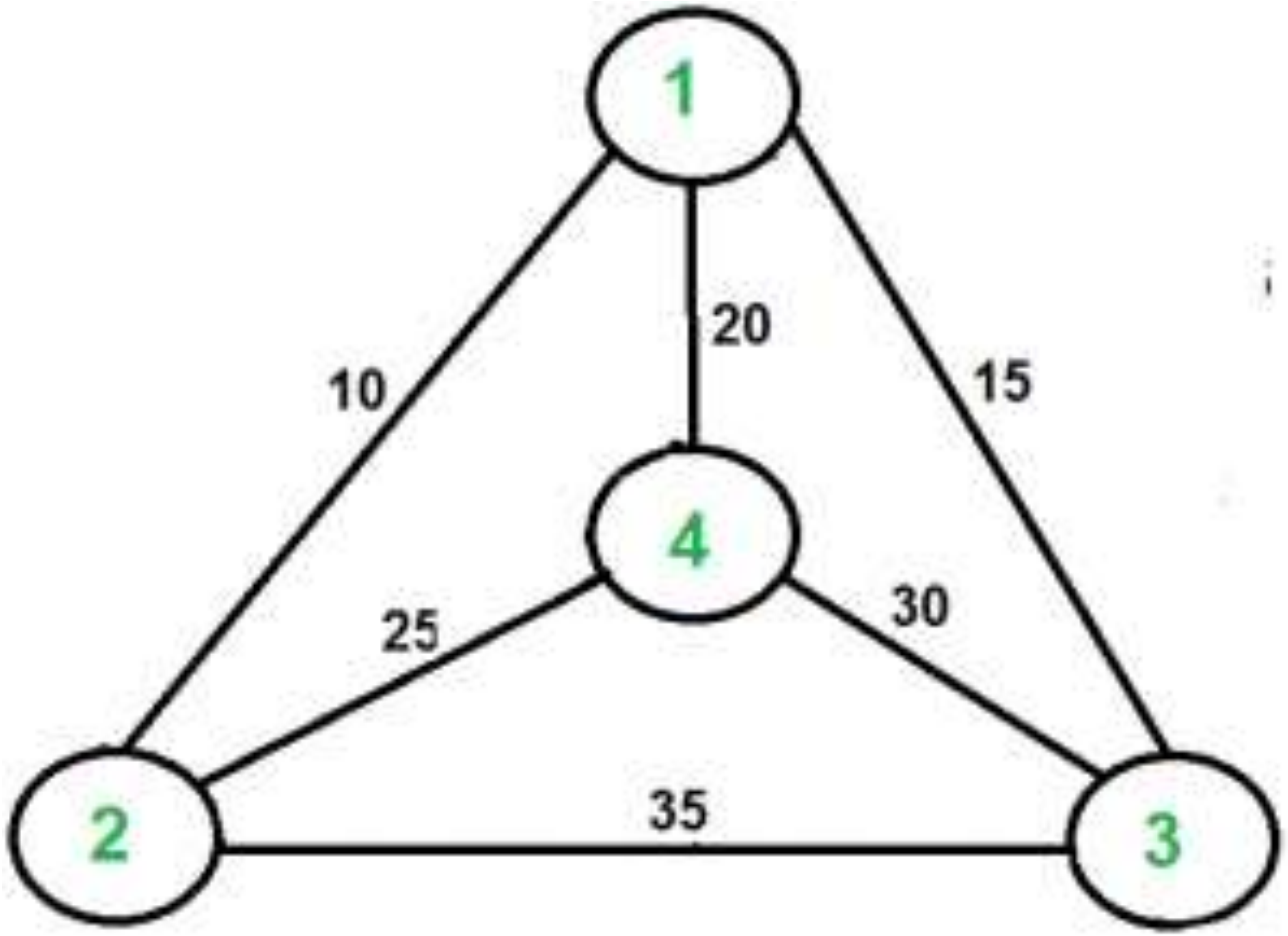
- The heuristic method might not always provide us the finest solution, but it is assured that it helps us find a good solution in a reasonable time.

- Based on context, there can be different heuristic methods that correlate with the problem's scope. The most common heuristic methods are - trial and error, guesswork, the process of elimination, historical data analysis.

- These methods involve simply available information that is not particular to the problem but is most appropriate. They can include representative, affect, and availability heuristics.

Traveling Salesman Problem (TSP)

Given a set of cities and distances between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point.
Note the difference between Hamiltonian Cycle and TSP. The Hamiltonian cycle problem is to find if there exists a tour that visits every city exactly once. Here we know that Hamiltonian Tour exists (because the graph is complete) and in fact, many such tours exist, the problem is to find a minimum weight Hamiltonian Cycle.
For example, A TSP tour in the graph is 1-2-4-3-1. The cost of the tour is 10+25+30+15 which is 80.
The problem is a famous NP-hard problem. There is no polynomial-time known solution for this problem.

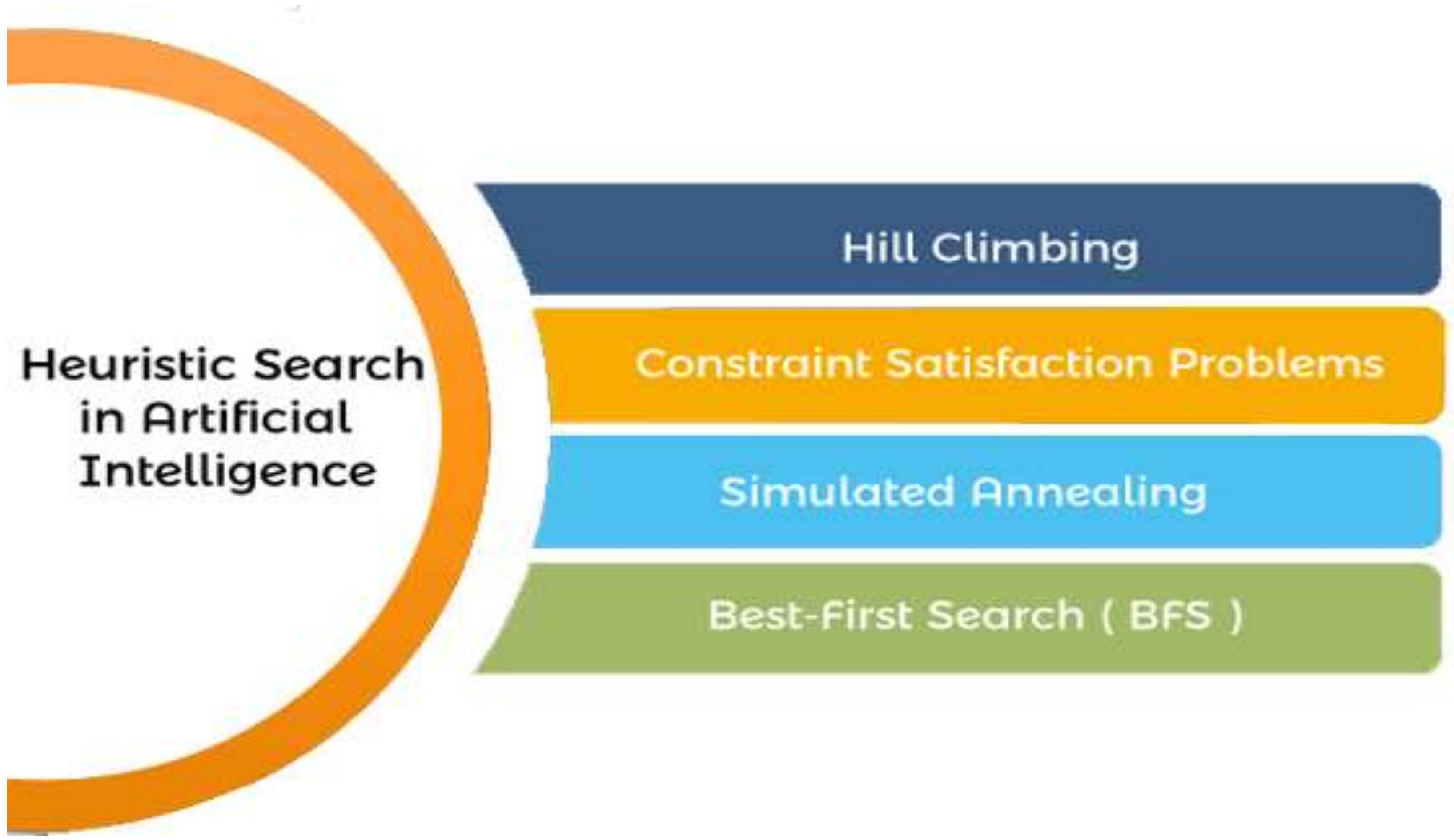

1
20
10
15
4
25
30
35
2
3

- Consider city 1 as the starting and ending point. Since the route is cyclic, we can consider any point as a starting point.
- Generate all (n-1)! permutations of cities.
- Calculate the cost of every permutation and keep track of the minimum cost permutation.
- Return the permutation with minimum cost.

Techniques in Heuristic Search

- Direct Heuristic Search(Informed Search)
- 1. **A*Search**
- 2. **Greedy Best First Search**
- Weak Heuristic Search (Uninformed Search)
- 1. **Breadth-First Search**
- 2. **Uniform Cost Search**
- 3. **Depth First Search**
- 4. **Iterative Deepening Depth First Search**
- 5. **Bidirectional Search**

Heuristic search techniques in AI (Artificial Intelligence)


Heuristic Search in Artificial Intelligence
Hill Climbing
Constraint Satisfaction Problems
Simulated Annealing
Best-First Search (BFS)

- uninformed search algorithms which looked through search space for all possible solutions of the problem without having any additional knowledge about search space. But informed search algorithm contains an array of knowledge such as how far we are from the goal, path cost, how to reach to goal node, etc.
- This knowledge help agents to explore less to the search space and find more efficiently the goal node.
- The informed search algorithm is more useful for large search space. Informed search algorithm uses the idea of heuristic, so it is also called Heuristic search.

Heuristics function

- Heuristic is a function which is used in Informed Search, and it finds the most promising path. It takes the current state of the agent as its input and produces the estimation of how close agent is from the goal.
- The heuristic method, however, might not always give the best solution, but it guaranteed to find a good solution in reasonable time. Heuristic function estimates how close a state is to the goal. It is represented by h(n), and it calculates the cost of an optimal path between the pair of states. The value of the heuristic function is always positive.

$H(n) <= h^*(n)$

Here h(n) is heuristic cost, and h*(n) is the estimated cost. Hence heuristic cost should be less than or equal to the estimated cost.

Pure Heuristic Search:

- Pure heuristic search is the simplest form of heuristic search algorithms. It expands nodes based on their heuristic value h(n). It maintains two lists, OPEN and CLOSED list. In the CLOSED list, it places those nodes which have already expanded and in the OPEN list, it places nodes which have yet not been expanded.
- On each iteration, each node n with the lowest heuristic value is expanded and generates all its successors and n is placed to the closed list. The algorithm continues unit a goal state is found.
- In the informed search we will discuss two main algorithms which are given below:
- Best First Search Algorithm(Greedy search)
- A* Search Algorithm

- Greedy best-first search algorithm always selects the path which appears best at that moment. It is the combination of depth-first search and breadth-first search algorithms. It uses the heuristic function and search. Best-first search allows us to take the advantages of both algorithms.
- With the help of best-first search, at each step, we can choose the most promising node. In the best first search algorithm, we expand the node which is closest to the goal node and the closest cost is estimated by heuristic function, i.e.
- $f(n)= g(n)$.
- Were, $h(n)=$ estimated cost from node n to the goal.
- The greedy best first algorithm is implemented by the priority queue.

Advantages:

- Best first search can switch between BFS and DFS by gaining the advantages of both the algorithms.
- This algorithm is more efficient than BFS and DFS algorithms.

Disadvantages:

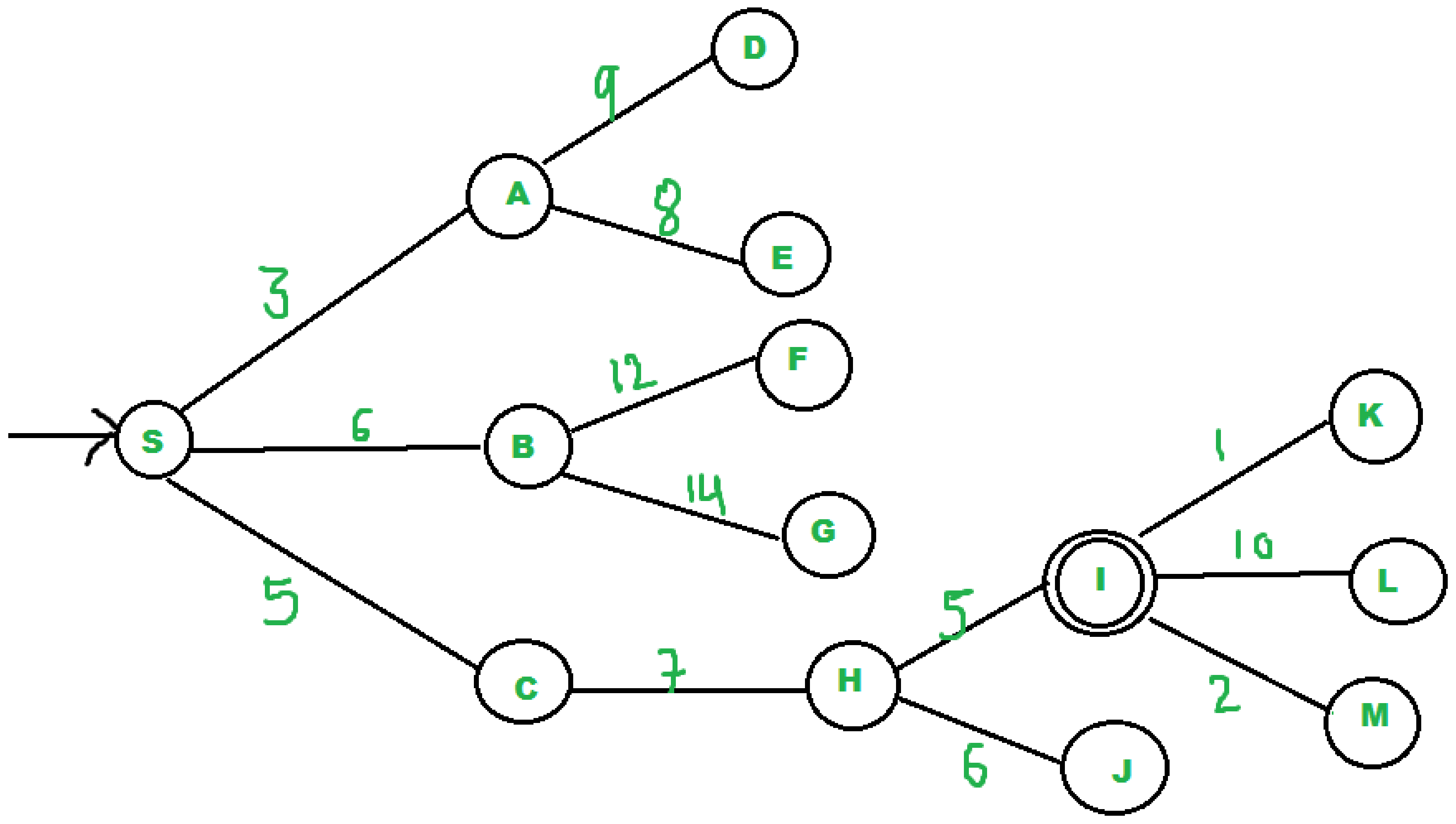
- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS.
- This algorithm is not optimal.

Best First Search (Informed Search)

In BFS and DFS, when we are at a node, we can consider any of the adjacent as the next node. So both BFS and DFS blindly explore paths without considering any cost function.

*The idea of **Best First Search** is to use an evaluation function to decide which adjacent is most promising and then explore.*

We use a priority queue or heap to store the costs of nodes that have the lowest evaluation function value. So the implementation is a variation of BFS, we just need to change Queue to PriorityQueue.


S
A
B
C
D
E
F
G
H
I
J
K
L
M
3
6
5
9
8
12
14
7
5
6
1
10
2

We start from source "S" and search for goal "I" using given costs and Best First search.

STEP-2

- *pq initially contains SWe remove S from pq and process unvisited neighbors of S to pq.*
- *pq now contains {A, C, B} (C is put before B because C has lesser cost)*

STEP-3

- *We remove A from pq and process unvisited neighbors of A to pq.*
    - *pq now contains {C, B, E, D}*

- *We remove C from pq and process unvisited neighbors of C to pq.*
    - *pq now contains {B, H, E, D}*
- *We remove B from pq and process unvisited neighbors of B to pq.*
    - *pq now contains {H, E, D, F, G}*
- *We remove H from pq.*
- *Since our goal "I" is a neighbor of H, we return.*

```
1.Create an Empty Priority Queue
              PriorityQueue pq
2.  Insert "start" in pq.
              pq.insert(start)
  3. Until PriorityQueue is empty
            u = PriorityQueue.DeleteMin
   If u is the goal
      Exit
```

```
• Else
        Foreach neighbor v of u
If v is “Unvisited”
       Mark v “visited”
       Pq.insert(V)
       Mark u Examined
End procedure
```

Direct Heuristic Search techniques in AI

- It includes Blind Search, Uninformed Search, and Blind control strategy. These search techniques are not always possible as they require much memory and time. These techniques search the complete space for a solution and use the arbitrary ordering of operations.
- The examples of Direct Heuristic search techniques include Breadth-First Search (BFS) and Depth First Search (DFS).

Weak Heuristic Search techniques in AI

- It includes Informed Search, Heuristic Search, and Heuristic control strategy. These techniques are helpful when they are applied properly to the right types of tasks. They usually require domain-specific information.
- The examples of Weak Heuristic search techniques include Best First Search (BFS) and A*.

The picture can't be displayed.

Generate and Test Heuristic Search

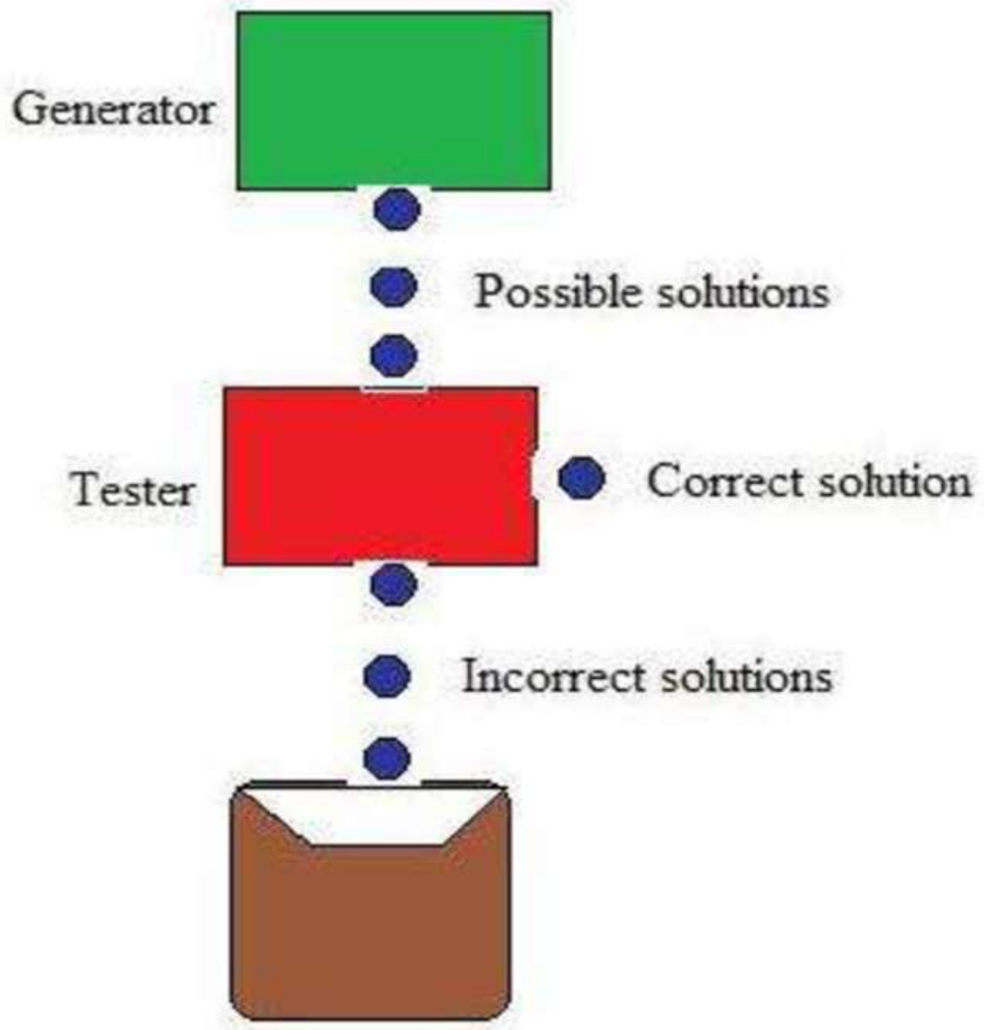
- The generate-and-test strategy is the simplest of all the approaches. It consists of the following steps:

Algorithm: Generate-and-Test

1. Generate a possible solution. For some problems. this means generating a particular point in the problem space. For others, it means generating a path from a start state.

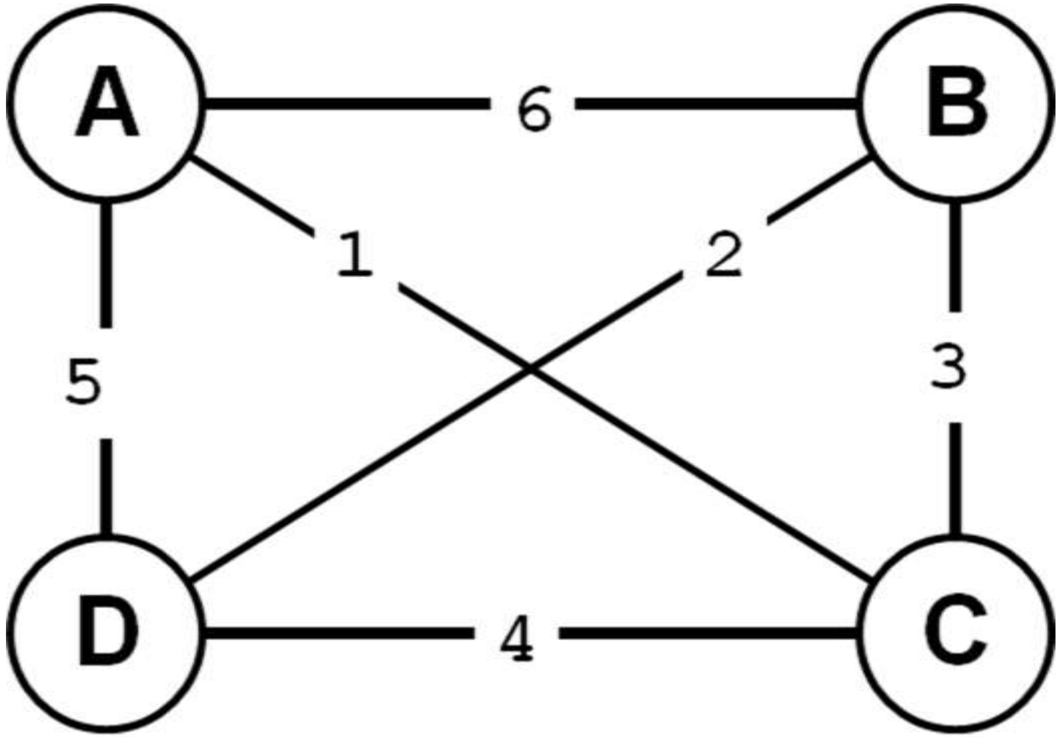
2. Test to see if this is actually a solution by comparing the chosen point or the endpoint of the chosen path to the set of acceptable goal states.

3. If a solution has been found, quit. Otherwise, return to step 1.

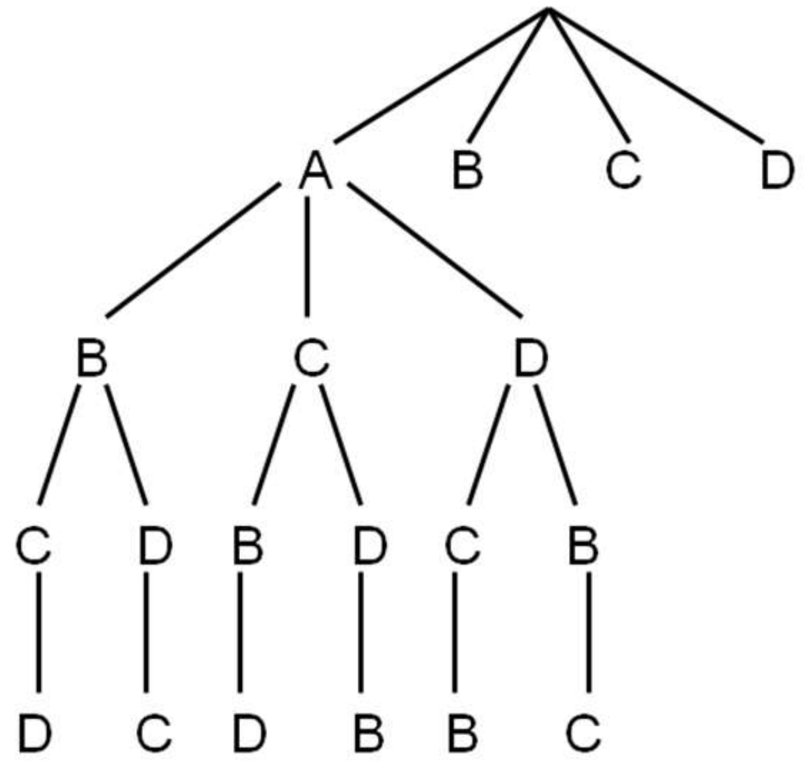

Generator
Possible solutions
Tester
Correct solution
Incorrect solutions

- Generate-and-test, like depth-first search, requires that complete solutions be generated for testing.
- In its most systematic form, it is only an exhaustive search of the problem space.
- Solutions can also be generated randomly but the solution is not guaranteed.
- This approach is what is known as the British Museum algorithm: finding an object in the British Museum by wandering randomly.

- **Example – Traveling Salesman Problem (TSP)**
- A salesman has a list of cities, each of which he must visit exactly once. There are direct roads between each pair of cities on the list. Find the route the salesman should follow for the shortest possible round trip that both starts and finishes at any one of the cities.
- Traveler needs to visit n cities.
- Know the distance between each pair of cities.
- Want to know the shortest route that visits all the cities once.


A
B
6
1
2
5
3
D
C
4

Search flow with Generate and Test



Search for	Path	Length of Path
1	ABCD	19
2	ABDC	18
3	ACBD	12
4	ACDB	13
5	ADBC	16

Requirement of Search Algorithms / Techniques

- The first requirement is that it causes motion, in a game playing program, it moves on the board and in the water jug problem, filling water is used to fill jugs. It means the control strategies without the motion will never lead to the solution.
- The second requirement is that it is systematic, that is, it corresponds to the need for global motion as well as for local motion. This is a clear condition that neither would it be rational to fill a jug and empty it repeatedly, nor it would be worthwhile to move a piece round and round on the board in a cyclic way in a game.

- **Example:** Finding a route from one city to another city is an example of a search problem in which different search orders and the use of heuristic knowledge are easily understood.
- **State:** The current city in which the traveler is located.
- **Operators:** Roads linking the current city to other cities.
- **Cost Metric:** The cost of taking a given road between cities.
- **Heuristic information:** The search could be guided by the direction of the goal city from the current city, or we could use airline distance as an estimate of the distance to the goal.

- For complex problems, the traditional algorithms, presented above, are unable to find the solution within some practical time and space limits. Consequently, many special techniques are developed, using heuristic functions.
- Blind search is not always possible, because it requires too much time or space (memory).
- Heuristics are rules of thumb; they do not guarantee a solution to a problem.
- Heuristic Search is a weak technique but can be effective if applied correctly; it requires domain-specific information.

Heuristic search compared with other search

Brute force / Blind search	**Heuristic search**
Can only search what it has knowledge about already	Estimates ‘distance’ to goal state through explored nodes
No knowledge about how far a node from the goal state	Guides search process toward the goal
	Prefers states (nodes) that lead close to and not away from goal State

Hill Climbing Algorithm in Artificial Intelligence

- A hill-climbing algorithm is a local search algorithm that moves continuously upward (increasing) until the best solution is attained. This algorithm comes to an end when the peak is reached.
- This algorithm has a node that comprises two parts: state and value. It begins with a non-optimal state (the hill's base) and upgrades this state until a certain precondition is met.
- The heuristic function is used as the basis for this precondition. The process of continuous improvement of the current state of iteration can be termed as climbing. This explains why the algorithm is termed as a *hill-climbing algorithm*.

- Hill Climbing is a heuristic search used for mathematical optimization problems in the field of Artificial Intelligence.
  Given a large set of inputs and a good heuristic function, it tries to find a sufficiently good solution to the problem. This solution may not be the global optimal maximum.
- In the above definition, **mathematical optimization problems** imply that hill-climbing solves the problems where we need to maximize or minimize a given real function by choosing values from the given inputs.
- Example-Travelling salesman problem where we need to minimize the distance traveled by the salesman.
- ‘Heuristic search’ means that this search algorithm may not find the optimal solution to the problem. However, it will give a good solution in a **reasonable time.**
- A **heuristic function** is a function that will rank all the possible alternatives at any branching step in the search algorithm based on the available information. It helps the algorithm to select the best route out of possible routes.

- A hill-climbing algorithm’s objective is to attain an optimal state that is an upgrade of the existing state.
- When the current state is improved, the algorithm will perform further incremental changes to the improved state. This process will continue until a peak solution is achieved. The peak state cannot undergo further improvements.

Features of a hill climbing algorithm

- A hill-climbing algorithm has four main features:

1.greedy approach

2.No Backtracking

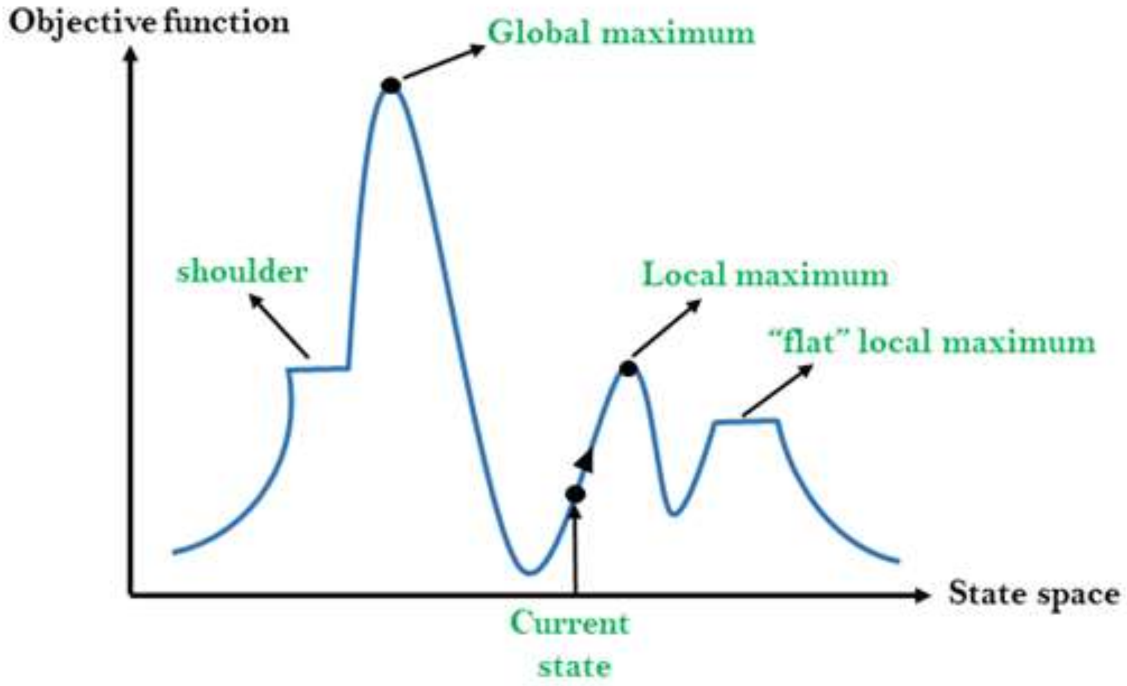
3.Feedback mechanism

4.Incremental change

1. It employs a **greedy approach:** This means that it moves in a direction in which the cost function is optimized. The greedy approach enables the algorithm to establish local maxima or minima.
2. **No Backtracking:** A hill-climbing algorithm only works on the current state and succeeding states (future). It does not look at the previous states.
3. **Feedback mechanism:** The algorithm has a feedback mechanism that helps it decide on the direction of movement (whether up or down the hill). The feedback mechanism is enhanced through the generate-and-test technique.
4. **Incremental change:** The algorithm improves the current solution by incremental changes.

State-space diagram analysis

- A state-space diagram provides a graphical representation of states and the optimization function. If the objective function is the y-axis, we aim to establish the local maximum and global maximum.
- If the cost function represents this axis, we aim to establish the local minimum and global minimum. More information about local minimum, local maximum, global minimum, and global maximum can be found here.
- The following diagram shows a simple state-space diagram. The objective function has been shown on the y-axis, while the state-space represents the x-axis.


Objective function
Global maximum
shoulder
Local maximum
"flat" local maximum
State space
Current state

A state-space diagram consists of various regions that can be explained as follows;

- **Local maximum:** A local maximum is a solution that surpasses other neighboring solutions or states but is not the best possible solution.
- **Global maximum:** This is the best possible solution achieved by the algorithm.
- **Current state:** This is the existing or present state.
- **Flat local maximum:** This is a flat region where the neighboring solutions attain the same value.
- **Shoulder:** This is a plateau whose edge is stretching upwards.

Problems with hill climbing

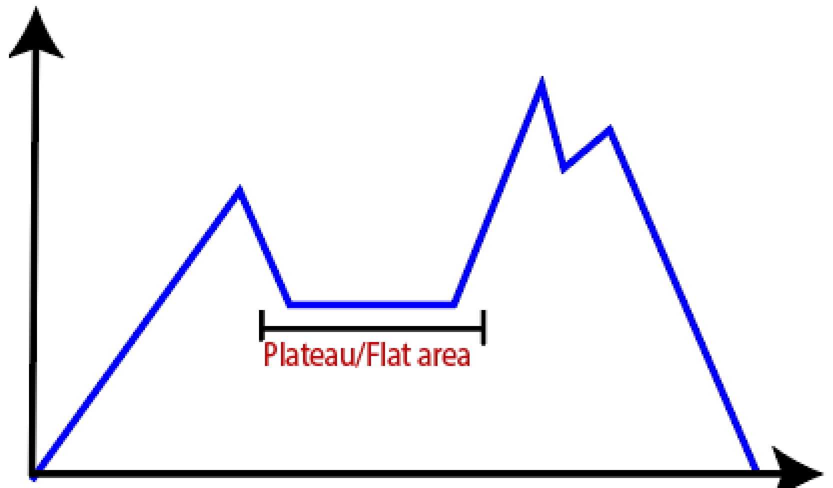
- There are three regions in which a hill-climbing algorithm cannot attain a global maximum or the optimal solution: local maximum, ridge, and plateau.

Local maximum

- At this point, the neighboring states have lower values than the current state. The greedy approach feature will not move the algorithm to a worse off state. This will lead to the hill-climbing process’s termination, even though this is not the best possible solution.
- This problem can be solved using momentum. This technique adds a certain proportion (m) of the initial weight to the current one. m is a value between 0 and 1. Momentum enables the hill-climbing algorithm to take huge steps that will make it move past the local maximum.

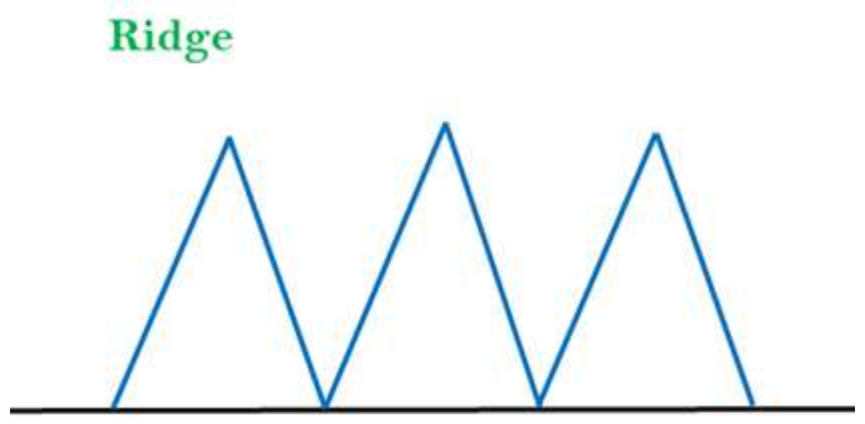
Plateau

- In this region, the values attained by the neighboring states are the same. This makes it difficult for the algorithm to choose the best direction.
- This challenge can be overcome by taking a huge jump that will lead you to a non-plateau space.



Ridge

- The hill-climbing algorithm may terminate itself when it reaches a ridge. This is because the peak of the ridge is followed by downward movement rather than upward movement.
- This impediment can be solved by going in different directions at once.



Types of hill climbing algorithms

- The following are the types of a hill-climbing algorithm:

1. Simple hill climbing
2. Steepest – Ascent hill climbing
3. Stochastic hill climbing

Simple hill climbing

- This is a simple form of hill climbing that evaluates the neighboring solutions. If the next neighbor state has a higher value than the current state, the algorithm will move. The neighboring state will then be set as the current one.
- This algorithm consumes less time and requires little computational power. However, the solutions produced by the algorithm are sub-optimal. In some cases, an optimal solution may not be guaranteed.

Algorithm:

- **Step 1:** It will evaluate the initial state.
- **Condition:**
  a) If it is found to be final state, stop and return success
  b) If it is not found to be the final state, make it a current state.
- **Step 2:** If no state is found giving a solution, perform looping.
- A state which is not applied should be selected as the current state and with the help of this state, produce a new state.
- Evaluate the new state produced.
- **Conditions:**
  1. If it is found to be final state, stop and return success.
  2. If it is found better compared to current state, then declare itself as a current state and proceed.
  3. If it is not better, perform looping until it reaches a solution.
- **Step3:** Exit the process.

Steepest – Ascent hill climbing

- This algorithm is more advanced than the simple hill-climbing algorithm. It chooses the next node by assessing the neighboring nodes. The algorithm moves to the node that is closest to the optimal or goal state.
- This algorithm selects the next node by performing an evaluation of all the neighbor nodes. The node that gives the best solution is selected as the next node.

Algorithm:

- **Step 1:** Perform evaluation on the initial state.
- **Condition:**
  a) If it reaches the goal state, stop the process
  b) If it fails to reach the final state, the current state should be declared as the initial state.
- **Step 2:** Repeat the state if the current state fails to change or a solution is found.
- **Step 3:** Exit

Stochastic hill climbing

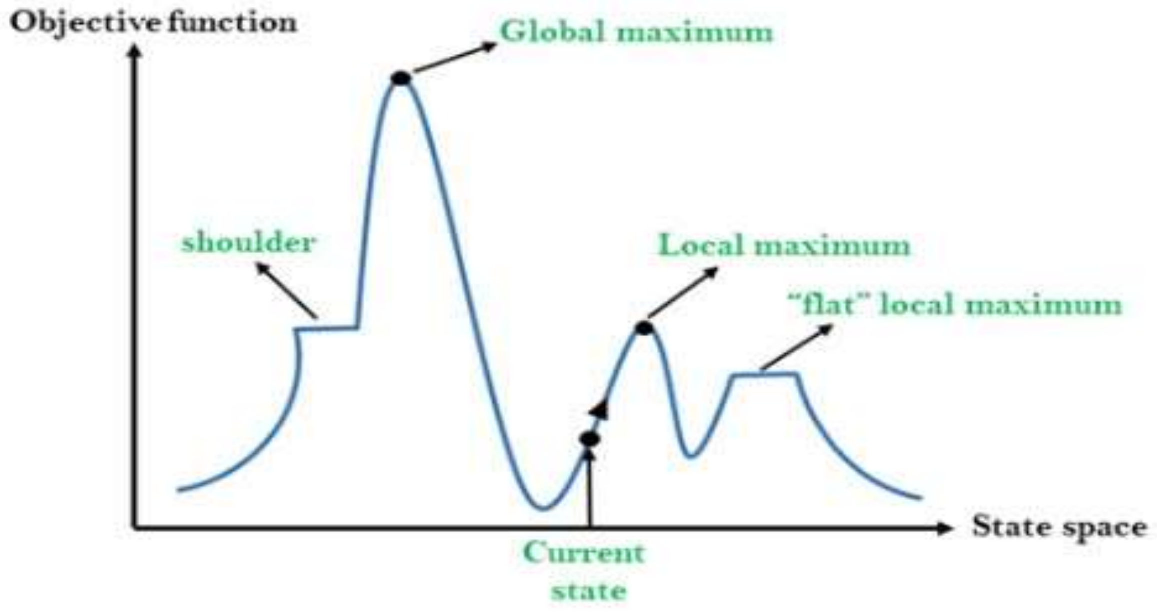
- In this algorithm, the neighboring nodes are selected randomly. The selected node is assessed to establish the level of improvement. The algorithm will move to this neighboring node if it has a higher value than the current state.

Features:

- The features of this algorithm are given below:
- It uses a greedy approach as it goes on finding those states which are capable of reducing the cost function irrespective of any direction.
- It is considered as a variant in generating expected solutions and the test algorithm. It first tries to generate solutions that are optimal and evaluates whether it is expected or not. If it is found the same as expected, it stops; else it again goes to find a solution.
- It does not perform a backtracking approach because it does not contain a memory to remember the previous space.

State Space Concept for Hill Climbing

- A state space is a landscape or a region which describes the relation between cost function and various algorithms. The following diagram gives the description of various regions.



- **Local Maximum:** As visible from the diagram, it is the state which is slightly better than the neighbor states but it is always lower than the highest state.
- **Global maximum:** It is the highest state of the state space and has the highest value of cost function.
- **Current State:** It is the state which contains the presence of an active agent.
- **Flat local maximum:** If the neighbor states all having same value, they can be represented by a flat space
  (as seen from the diagram) which are known as flat local maximums.
- **Shoulder region:** It is a region having an edge upwards and it is also considered as one of the problems in hill climbing algorithms.

Problems faced in Hill Climbing Algorithm

- **Local maximum:** The hill climbing algorithm always finds a state which is the best but it ends in a local maximum because neighboring states have worse values compared to the current state and hill climbing algorithms tend to terminate as it follows a greedy approach.
- To overcome such problems, backtracking technique can be used where the algorithm needs to remember the values of every state it visited.
- **Plateau:** In this region, all neighbors seem to contain the same value which makes it difficult to choose a proper direction.
- To overcome such issues, the algorithm can follow a stochastic process where it chooses a random state far from the current state. That solution can also lead an agent to fall into a non-plateau region.
- **Ridge:** In this type of state, the algorithm tends to terminate itself; it resembles a peak but the movement tends to be possibly downward in all directions.
- To overcome such issues, we can apply several evaluation techniques such as travelling in all possible directions at a time.

Applications of Hill climbing technique

Robotics

- Hill climbing Is mostly used in robotics which helps their system to work as a team and maintain coordination.

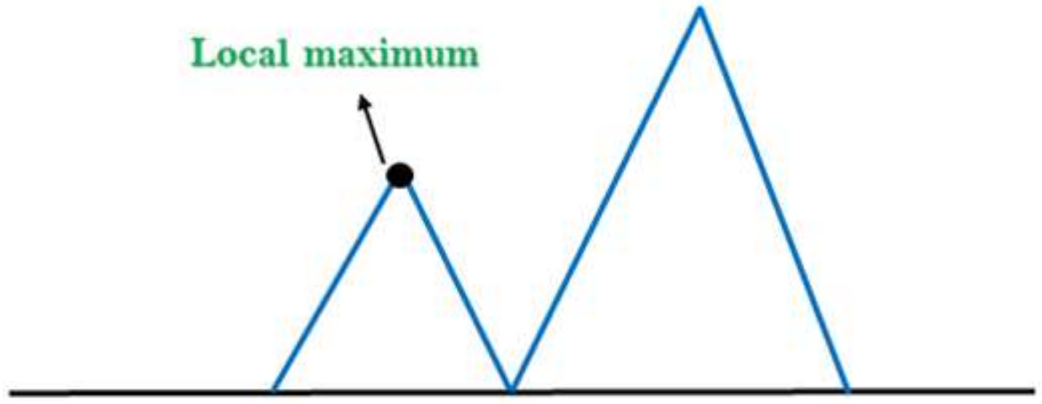
Marketing

- The algorithm can be helpful in team management in various marketing domains where hill climbing can be used to find an optimal solution. The travelling time taken by a sale member or the place he visited per day can be optimized using this algorithm.

- Problems in Hill Climbing Algorithm:

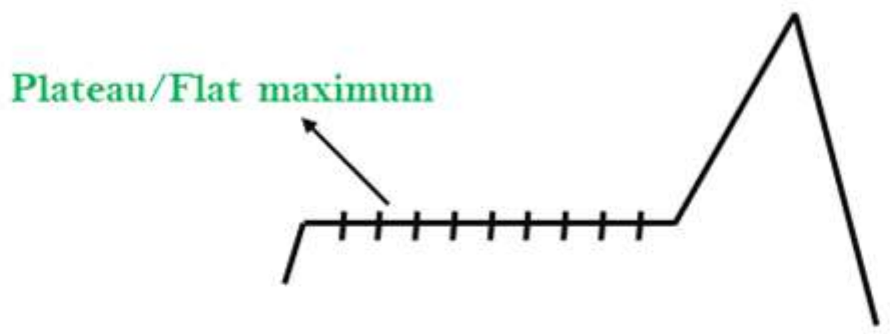
1. **Local Maximum:** A local maximum is a peak state in the landscape which is better than each of its neighboring states, but there is another state also present which is higher than the local maximum.

Solution: Backtracking technique can be a solution of the local maximum in state space landscape. Create a list of the promising path so that the algorithm can backtrack the search space and explore other paths as well.



2. Plateau: A plateau is the flat area of the search space in which all the neighbor states of the current state contains the same value, because of this algorithm does not find any best direction to move. A hill-climbing search might be lost in the plateau area.

- **Solution:** The solution for the plateau is to take big steps or very little steps while searching, to solve the problem. Randomly select a state which is far away from the current state so it is possible that the algorithm could find non-plateau region.



3. Ridges: A ridge is a special form of the local maximum. It has an area which is higher than its surrounding areas, but itself has a slope, and cannot be reached in a single move.

- **Solution:** With the use of bidirectional search, or by moving in different directions, we can improve this problem.

Types of Hill Climbing Algorithm:

1.Simple hill Climbing:

2.Steepest-Ascent hill-climbing:

3.Stochastic hill Climbing:

1. Simple Hill Climbing:

- Simple hill climbing is the simplest way to implement a hill climbing algorithm. **It only evaluates the neighbor node state at a time and selects the first one which optimizes current cost and set it as a current state**. It only checks it's one successor state, and if it finds better than the current state, then move else be in the same state. This algorithm has the following features:
- Less time consuming
- Less optimal solution and the solution is not guaranteed

Algorithm for Simple Hill Climbing:

- **Step 1:** Evaluate the initial state, if it is goal state then return success and Stop.
- **Step 2:** Loop Until a solution is found or there is no new operator left to apply.
- **Step 3:** Select and apply an operator to the current state.
- **Step 4:** Check new state:
    - If it is goal state, then return success and quit.
    - Else if it is better than the current state then assign new state as a current state.
    - Else if not better than the current state, then return to step2.
- **Step 5:** Exit.

2. Steepest-Ascent hill climbing:

- The steepest-Ascent algorithm is a variation of simple hill climbing algorithm. This algorithm examines all the neighboring nodes of the current state and selects one neighbor node which is closest to the goal state. This algorithm consumes more time as it searches for multiple neighbors

Algorithm for steepest-ascent hill climbing:

- **Step 1:** evaluate the initial state, if it is goal state then return success and stop, else make current state as initial state.
- **Step 2:** loop until a solution is found or the current state does not change.
  - Let succ be a state such that any successor of the current state will be better than it.
  - For each operator that applies to the current state:
    - Apply the new operator and generate a new state.
    - Evaluate the new state.
    - If it is goal state, then return it and quit, else compare it to the succ.
    - If it is better than succ, then set new state as succ.
    - If the succ is better than the current state, then set current state to succ.
- **Step 5:** exit.