

Unit 13:Artificial Intelligence

- **INTRODUCTION TO AI AND PRODUCTION SYSTEMS:** Introduction to AI-Problem formulation, Problem Definition -Production systems, Control strategies, Search strategies. Problem characteristics, Production system characteristics - Specialized productions system- Problem solving methods – Problem graphs, Matching, Indexing and Heuristic functions -Hill Climbing-Depth first and Breath first, Constraints satisfaction – Related algorithms, Measure of performance and analysis of search algorithms.
- **REPRESENTATION OF KNOWLEDGE:** Game playing – Knowledge representation, Knowledge representation using Predicate logic, Introduction to predicate calculus, Resolution, Use of predicate calculus, Knowledge representation using other logic-Structured representation of knowledge.
- **KNOWLEDGE INFERENCE:** Knowledge representation -Production based system, Frame based system. Inference – Backward chaining, Forward chaining, Rule value approach, Fuzzy reasoning – Certainty factors, Bayesian Theory-Bayesian Network-Dempster – Shafer theory.
- **PLANNING AND MACHINE LEARNING:** Basic plan generation systems – Strips -Advanced plan generation systems – K strips - Strategic explanations - Why, Why not and how explanations. Learning- Machine learning, adaptive Learning.
- **EXPERT SYSTEMS:** Expert systems – Architecture of expert systems, Roles of expert systems – Knowledge Acquisition – Meta knowledge, Heuristics. Typical expert systems – MYCIN, DART, XOON, Expert systems shells.



MULTIPLE INTELLIGENCES

SPACIAL INTELLIGENCE

Strengths

Visual and spacial judgment

Characteristics

- Draws for fun
- Good at puzzles
- Recognizes patterns
- Interprets visuals well

BODILY-KINESTHETIC INTELLIGENCE

Strengths

Physical movement, motor control

Characteristics

- Skilled at sports
- Excellent physical coordination
- Remembers by doing, instead of hearing or seeing

MUSICAL INTELLIGENCE

Strengths

Rhythm and music

Characteristics

- Appreciation for music
- Thinks in sounds and patterns
- Rich understanding of musical structure, notes

LINGUISTIC INTELLIGENCE

Strengths

Words, language, writing

Characteristics

- Enjoys writing, reading
- Good at public speaking
- Very persuasive
- Can explain things well

LOGICAL-MATHEMATICAL SKILLS

Strengths

Analyzing problems, mathematical operations

Characteristics

- Fast problem-solver
- Understands complex computations
- Likes thinking about abstract ideas

INTERPERSONAL INTELLIGENCE

Strengths

Understanding and relating to others

Characteristics

- Strong emotional intelligence skills
- Creates healthy relationships
- Good at solving conflicts

INTRAPERSONAL INTELLIGENCE

Strengths

Introspection and self-reflection

Characteristics

- Understands one's own strengths, weaknesses
- Highly self-aware
- Sensitive to one's own feelings

NATURALISTIC INTELLIGENCE

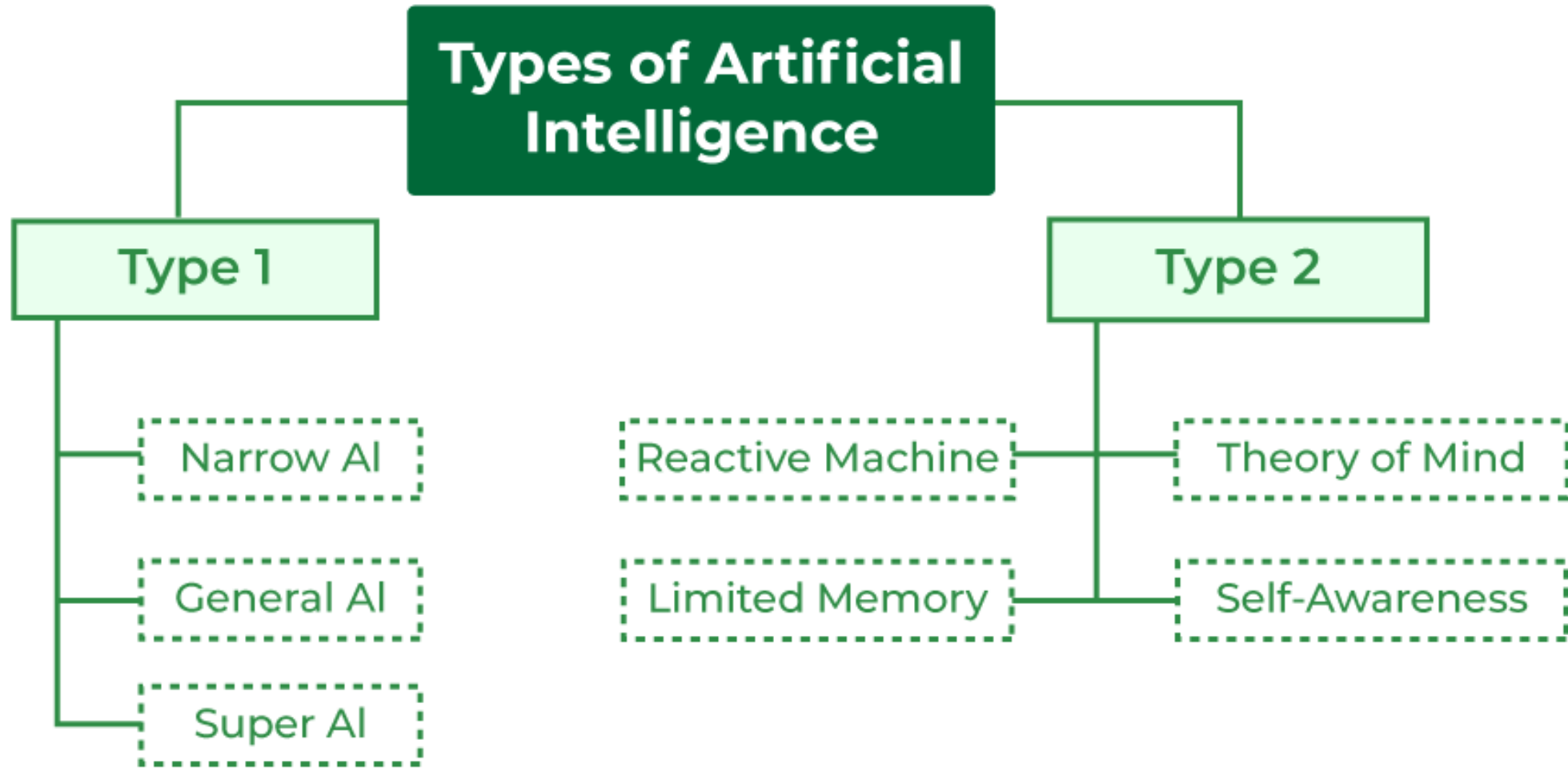
Strengths

Seeing patterns and relationships to nature

Characteristics

- Interested in areas like botany, biology, zoology
- Appreciation for nature
- Enjoys activities like camping, gardening, hiking

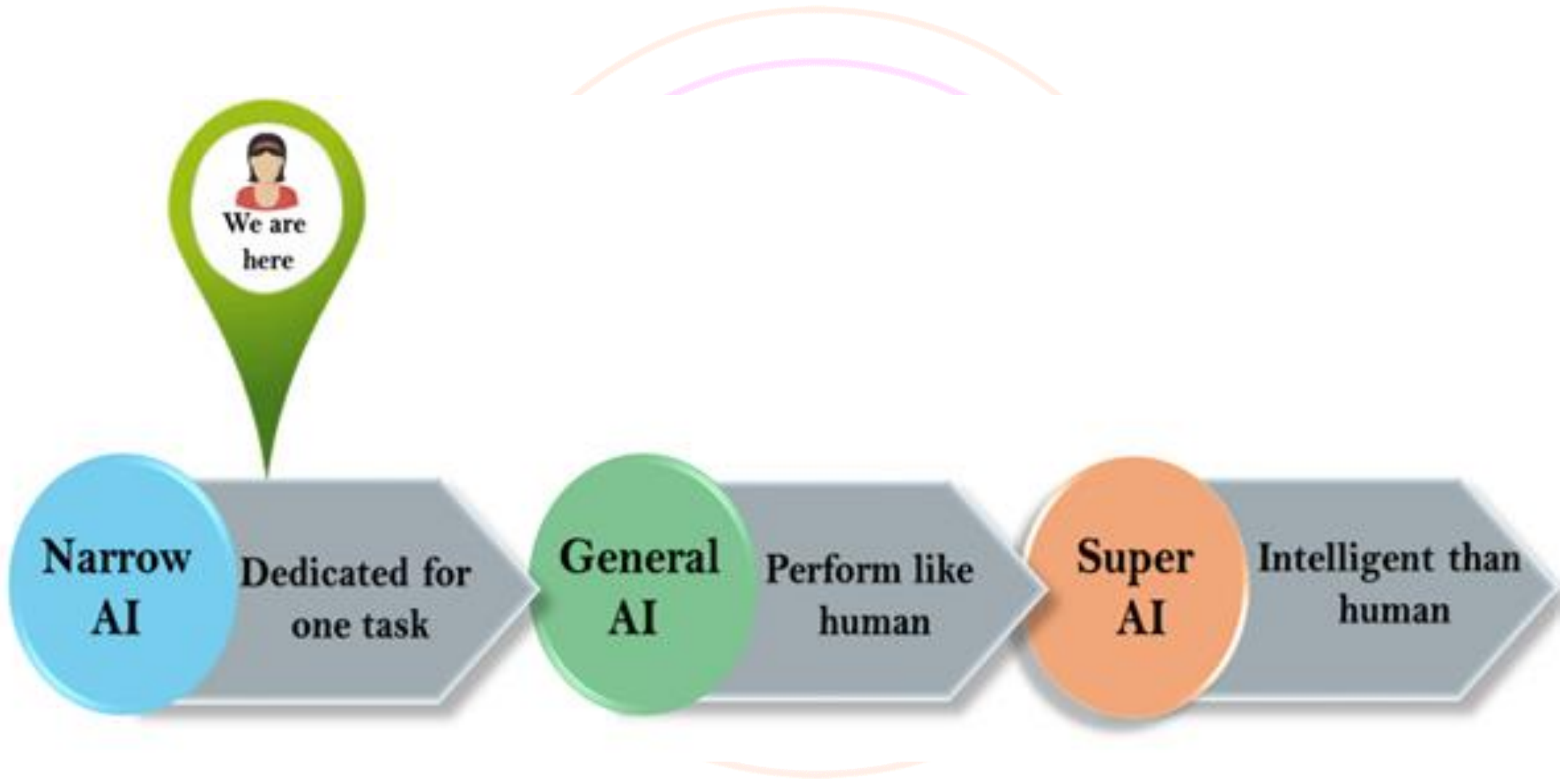




- ✓ Based on Capabilities of AI -Type 1
- ✓ Narrow AI: Narrow AI also known as Weak AI . Narrow AI is designed and trained on a specific task or a narrow range tasks. These Narrow AI systems are designed and trained for a purpose. These Narrow systems perform their designated tasks but mainly lack in the ability to generalize tasks. Personal Virtual assistance like Alexa or Siri, recommendation systems, image recognition software and other language translation tools.



- ✓ General AI: It is known as Strong AI. It refers to AI systems that have human intelligence and abilities to perform various tasks. Systems have capability to understand, learn and apply across a wide range of tasks that are similar to how a human can adapt to various tasks. In general AI remains a theoretical concept, and now no AI can achieve this level of intelligence.
- ✓ Super AI: It is known as Superintelligent AI that surpasses intelligence of human in solving-problem, creativity, and overall abilities. Super AI develops emotions, desires, need and beliefs of their own. They are able to make decisions of their own and solve problem of its own.



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.



Artificial Intelligence type-2: Based on functionality

2. Limited Memory

1. Limited memory machines can store past experiences or some data for a short period of time.
2. These machines can use stored data for a limited time period only.
3. 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, the speed limit, and other information to navigate the road.



3. Theory of Mind

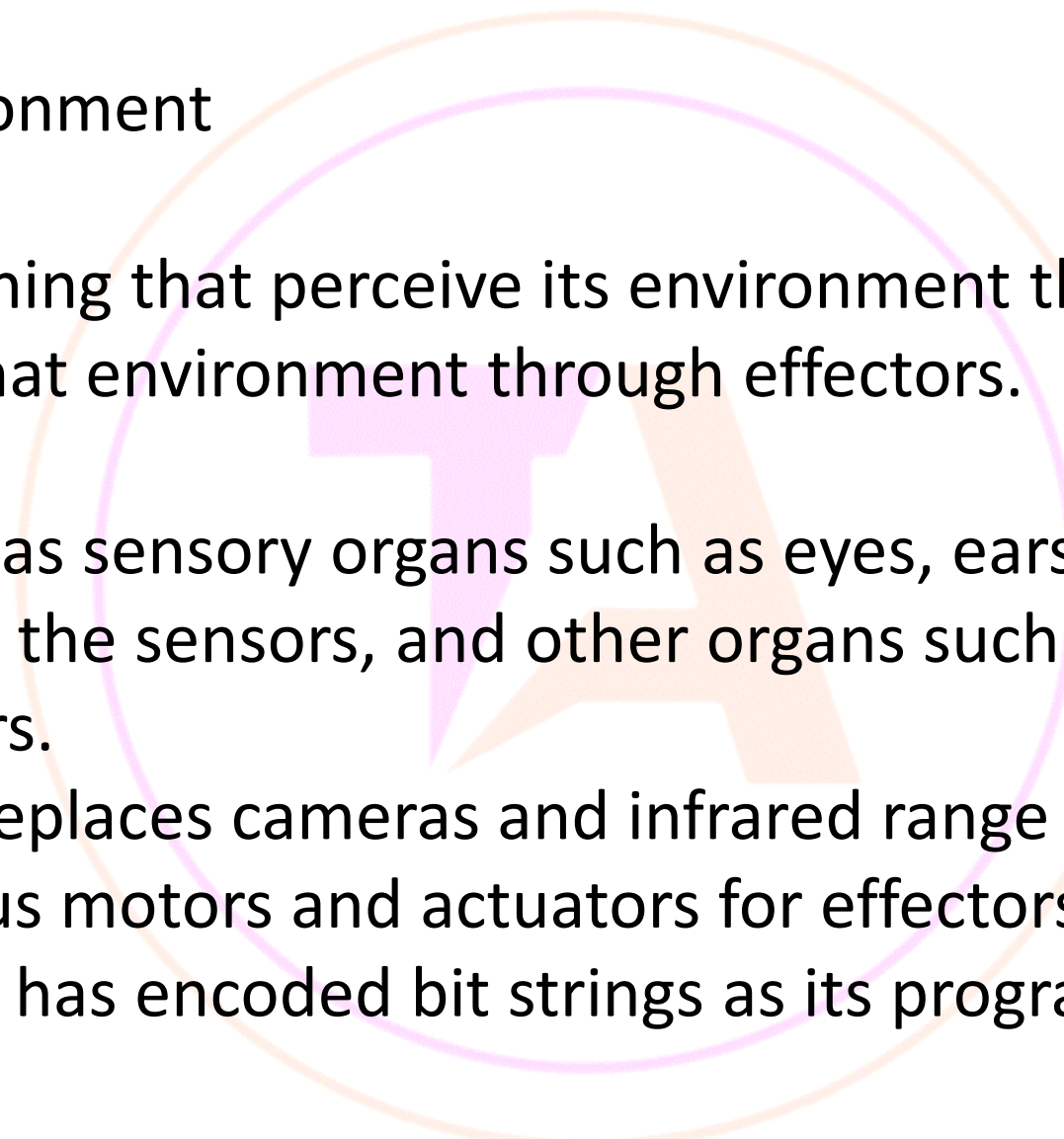
- Theory of Mind AI should understand the human emotions, people, beliefs, and be able to interact socially like humans.
- This type of AI machine are still not developed, but researchers are making lots of efforts and improvement for developing such AI machines.

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

- ✓ From Siri and Alexa, to self-driving cars, artificial intelligence (AI) is progressing rapidly.
- ✓ While science fiction often portrays AI as robots with human-like characteristics, AI can encompass anything from Google's search algorithms, to IBM's Watson, to autonomous weapons.
- ✓ Artificial intelligence today is properly known as narrow AI (or weak AI), in that it is designed to perform a narrow task such as only facial recognition, or only internet searches, or only driving a car).
- ✓ However, the long-term goal of many researchers is to create general AI (AGI or strong AI).
- ✓ While narrow AI may outperform humans at whatever its specific task is, like playing chess or solving equations, AI would outperform humans at nearly every thinking task.



- 
- ✓ Agent and Environment
 - ✓ An agent is anything that perceive its environment through sensors and acts upon that environment through effectors.
 - A human agent has sensory organs such as eyes, ears, nose, tongue and skin parallel to the sensors, and other organs such as hands, legs, mouth, for effectors.
 - A robotic agent replaces cameras and infrared range finders for the sensors, and various motors and actuators for effectors.
 - A software agent has encoded bit strings as its programs and actions.

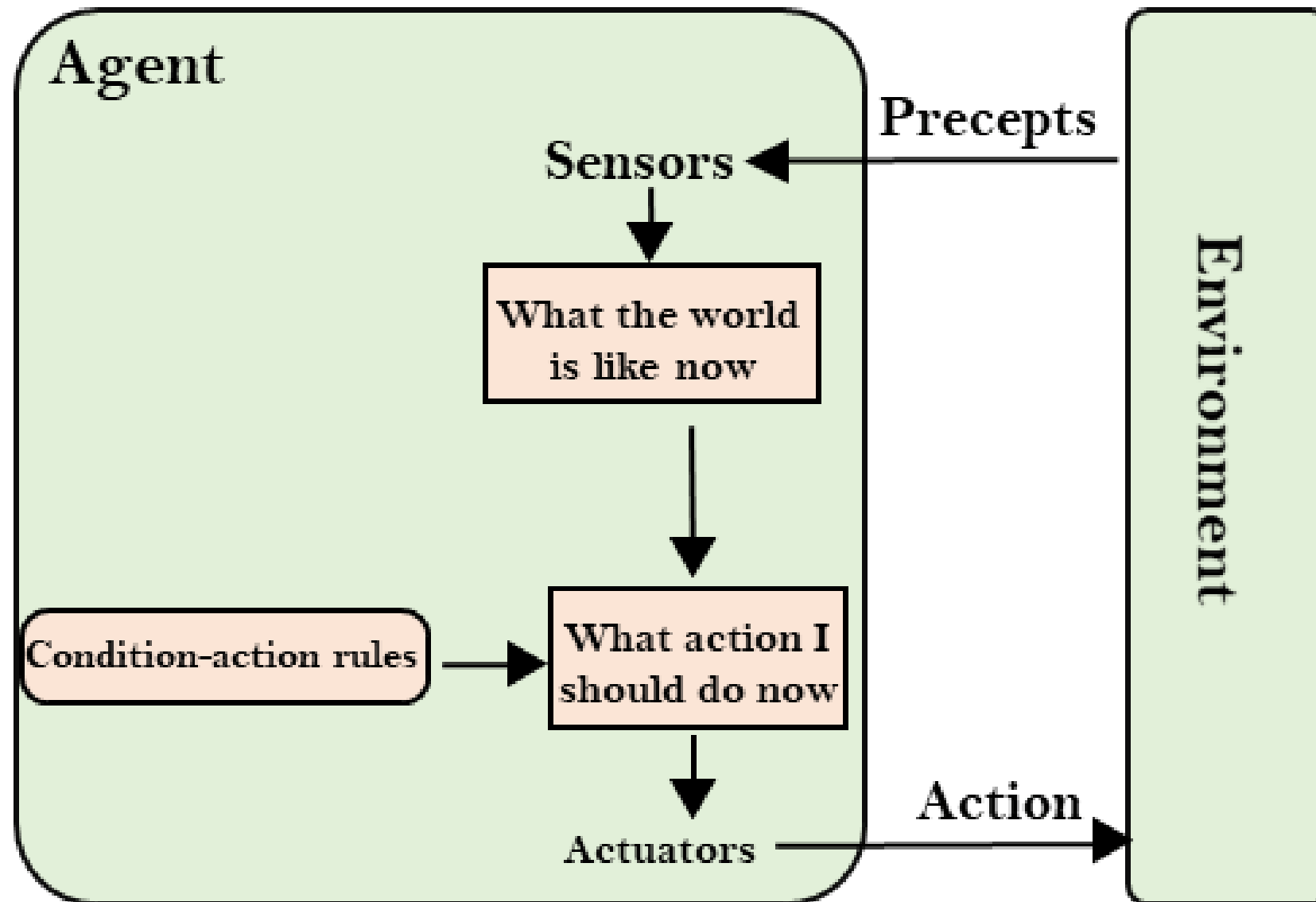
✓ Agent Terminology

- Performance Measure of Agent – It is the criteria, which determines how successful an agent is.
- Behavior of Agent – It is the action that agent performs after any given sequence of percepts.
- Percept – It is agent's perceptual inputs at a given instance.
- Percept Sequence – It is the history of all that an agent has perceived till date.
- Agent Function – It is a map from the precept sequence to an action.

(a) Simple Reflex Agents

- They choose actions only based on the current percept.
- They are rational only if a correct decision is made only based on the current precept.
- Their environment is completely observable.

Condition-Action Rule – It is a rule that maps a state (condition) to an action.



(b) Model Based Reflex Agents

They use a model of the world to choose their actions. They maintain an internal state.

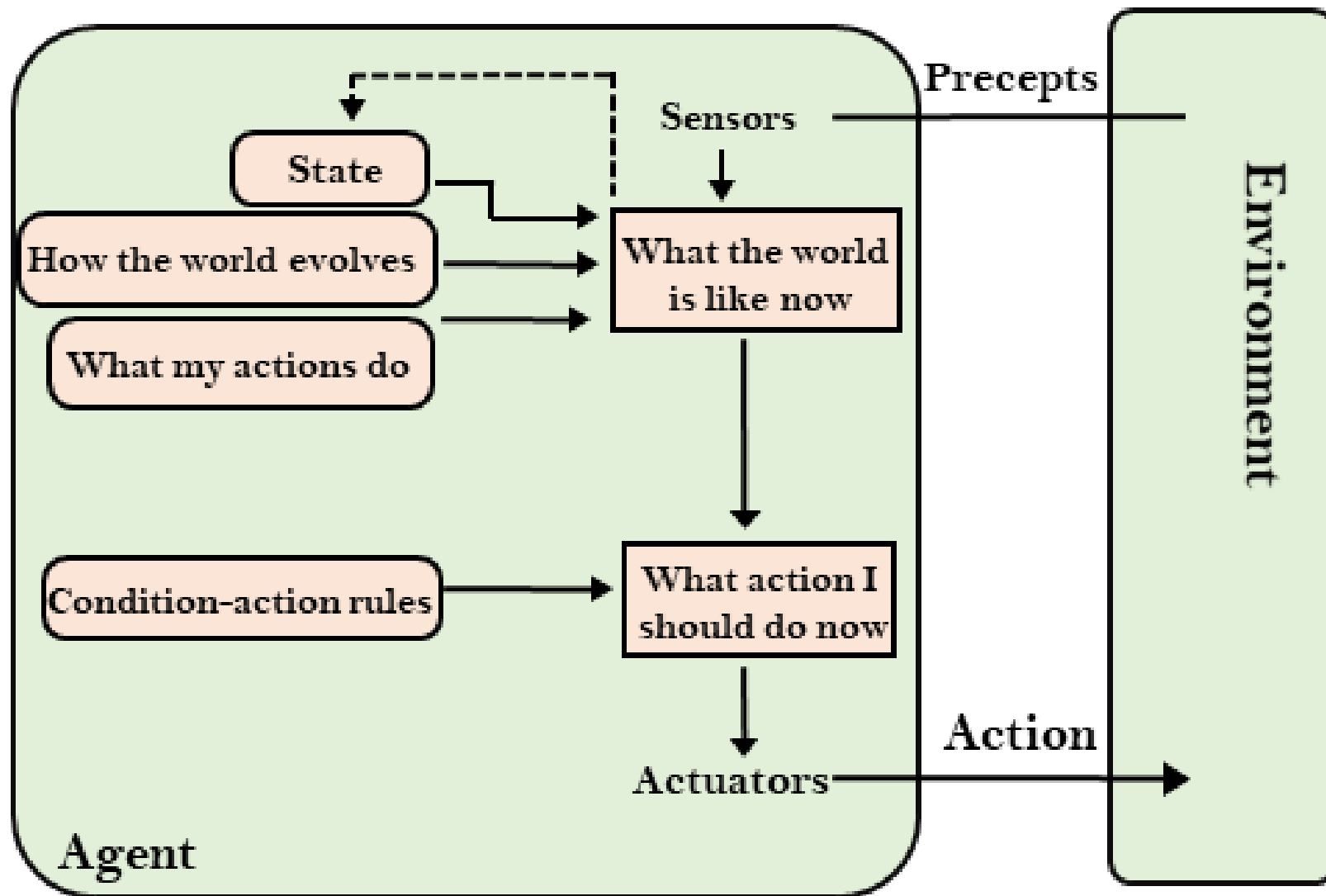
Model – knowledge about “how the things happen in the world”.

Internal State – It is a representation of unobserved aspects of current state depending on percept history.

Updating the state requires the information about –

- How the world evolves.
- How the agent's actions affect the world.



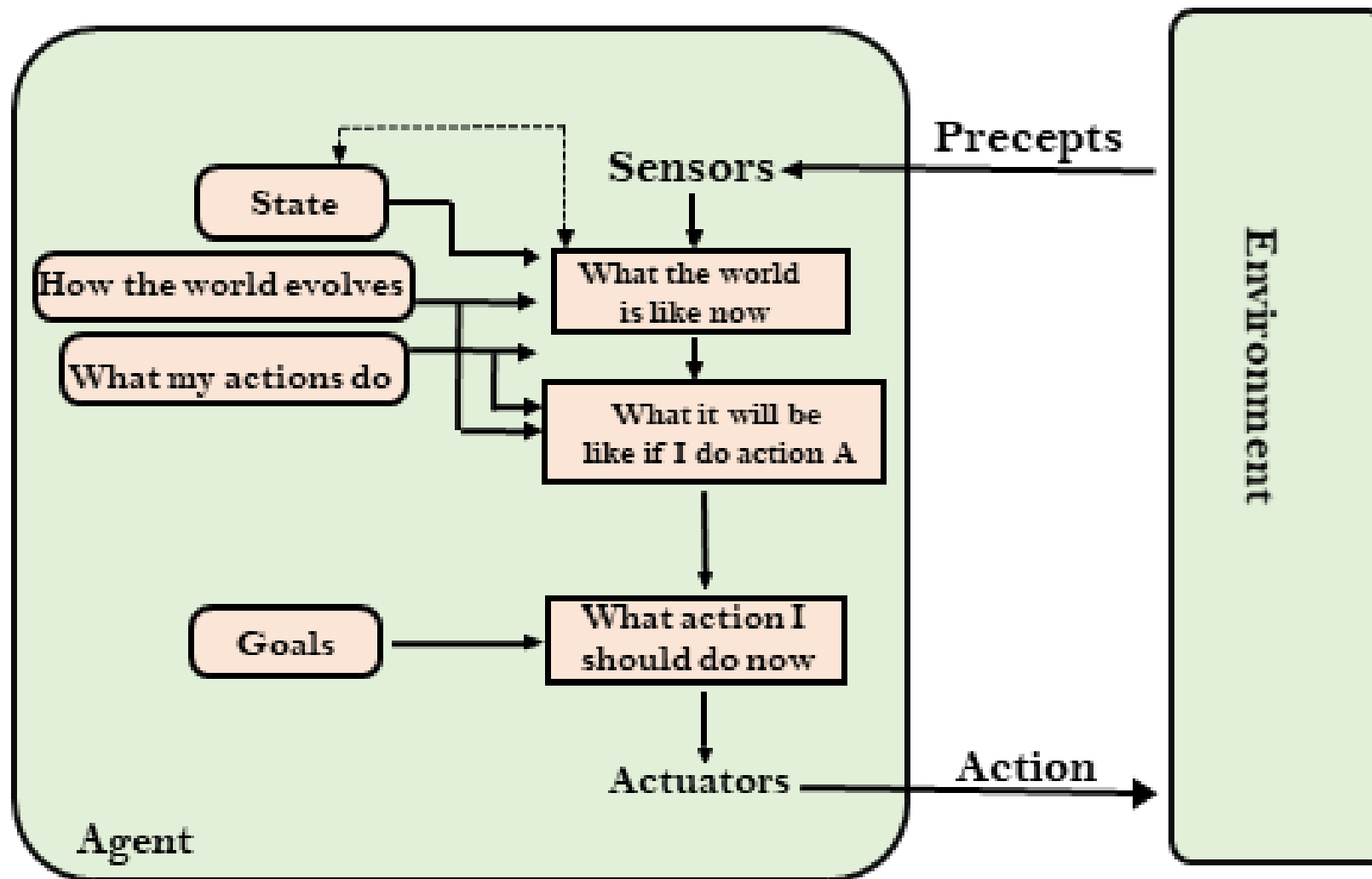


(c) Goal Based Agents

They choose their actions in order to achieve goals. Goal-based approach is more flexible than reflex agent since the knowledge supporting a decision is explicitly modeled, thereby allowing for modifications.

Goal – It is the description of desirable situations.





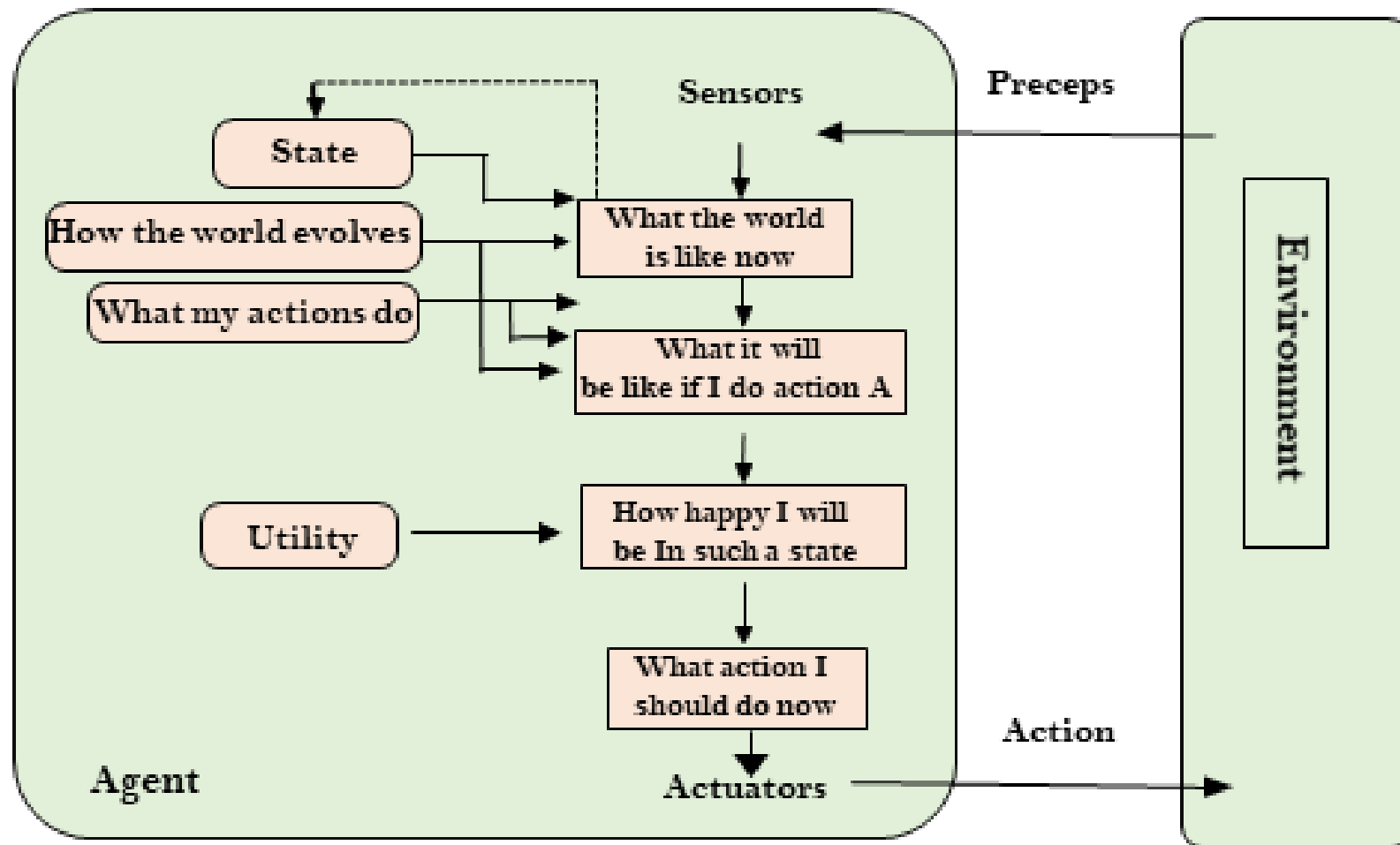
(d) Utility Based Agents

They choose actions based on a preference (utility) for each state.

Goals are inadequate when –

- There are conflicting goals, out of which only few can be achieved.
- Goals have some uncertainty of being achieved and you need to weigh likelihood of success against the importance of a goal.





(e). Learning Agents

A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities.

It starts to act with basic knowledge and then able to act and adapt automatically through learning.

A learning agent has mainly four conceptual components, which are:

1. Learning element: It is responsible for making improvements by learning from environment



(e). Learning Agents

2. Critic: Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
3. Performance element: It is responsible for selecting external action
4. Problem generator: This component is responsible for suggesting actions that will lead to new and informative experiences.



(f) Intelligent Agents:

An intelligent agent is an autonomous entity which act upon an environment using sensors and actuators for achieving goals. An intelligent agent may learn from the environment to achieve their goals. A thermostat is an example of an intelligent agent.

Following are the main four rules for an AI agent:

Rule 1: An AI agent must have the ability to perceive the environment.

Rule 2: The observation must be used to make decisions.

Rule 3: Decision should result in an action.

Rule 4: The action taken by an AI agent must be a rational action.



(g) Rational Agent:

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

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

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



Problem-solving agents:

In Artificial Intelligence, Search techniques are universal problem-solving methods. Rational agents or Problem-solving agents in AI mostly used these search strategies or algorithms to solve a specific problem and provide the best result.

Problem-solving agents are the goal-based agents and use atomic representation.



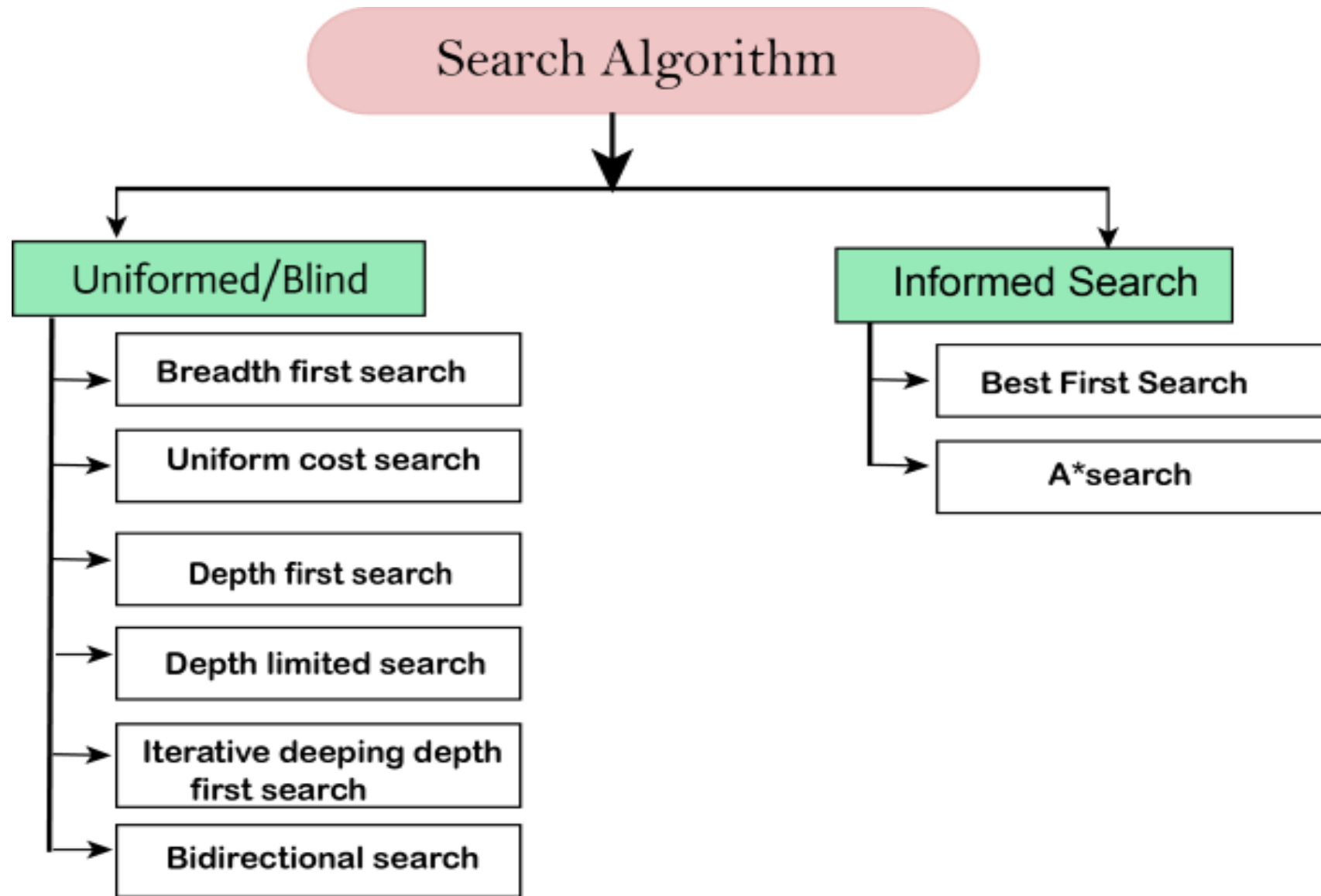
Problem-solving agents:

- ✓ 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.

Problem-solving agents:

- ✓ 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.





✓ 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.



✓ 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 a reasonable time..



1. Breadth-first Search:

- ✓ 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

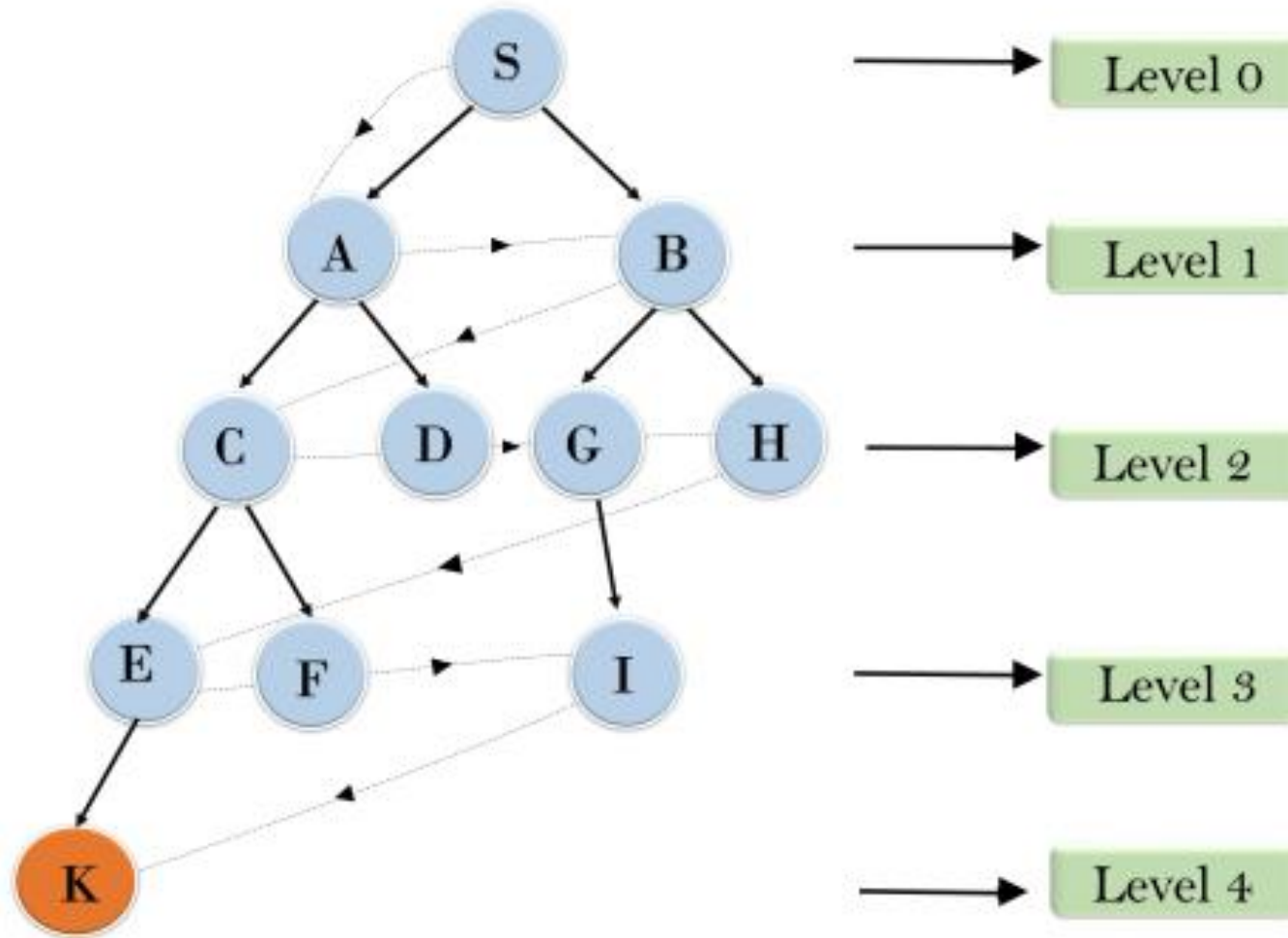


1. Breadth-first Search:

- ✓ 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



Breadth First Search



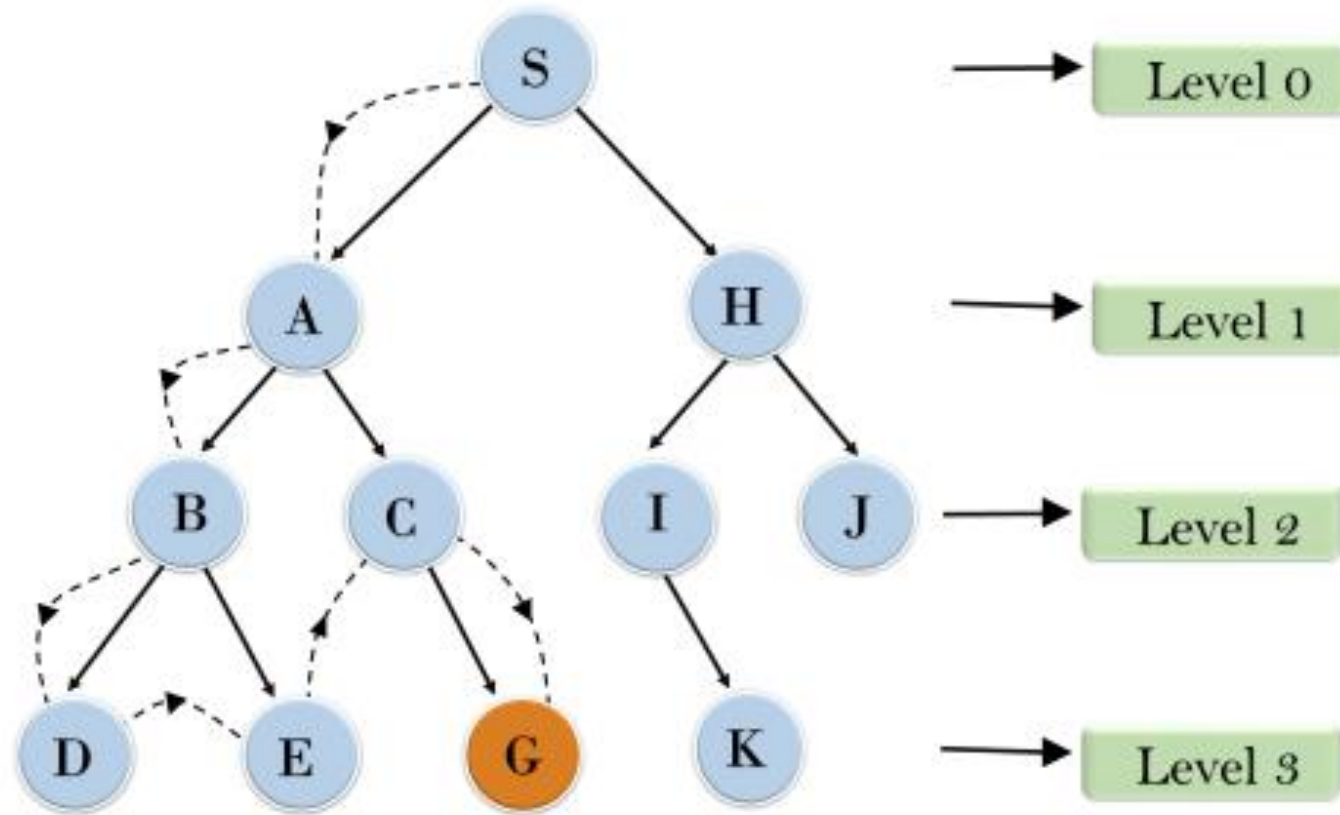
2. Depth-first Search

- ✓ 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.
- ✓ Note: Backtracking is an algorithm technique for finding all possible solutions using recursion.

2. Depth-first Search

- ✓ 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

Depth First Search



3. Uniform-cost Search Algorithm:

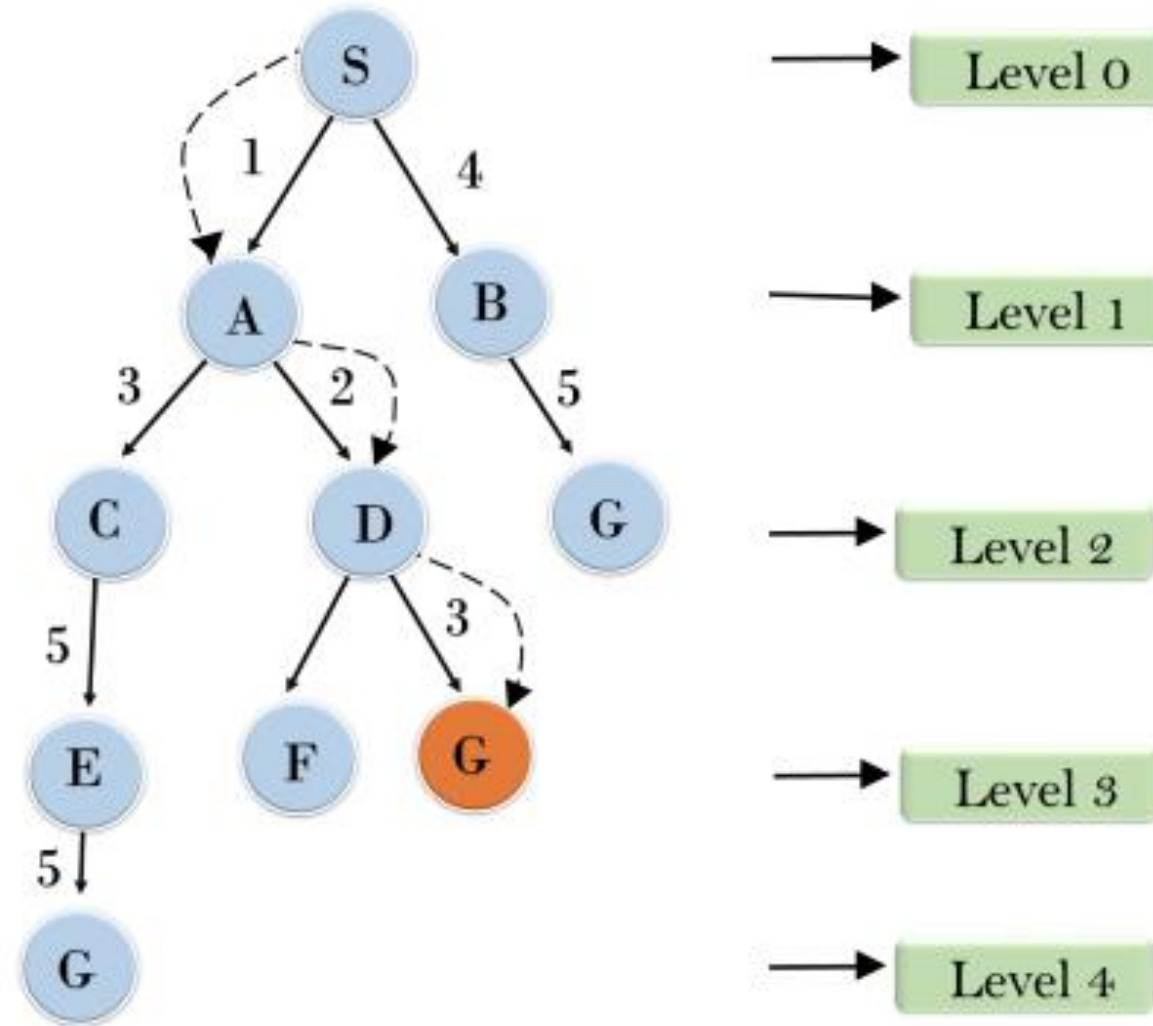
Uniform-cost search is a searching algorithm used for traversing a weighted tree or graph. This algorithm comes into play when a different cost is available for each edge.

The primary goal of the uniform-cost search is to find a path to the goal node which has the lowest cumulative cost. Uniform-cost search expands nodes according to their path costs from the root node. It can be used to solve any graph/tree where the optimal cost is in demand.

A uniform-cost search algorithm is implemented by the priority queue. It gives maximum priority to the lowest cumulative cost. Uniform cost search is equivalent to BFS algorithm if the path cost of all edges is the same.



Uniform Cost Search



Informed Search Algorithms

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.



1.) Best-first Search Algorithm (Greedy Search):

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



1.) Best-first Search Algorithm (Greedy Search):

Step 1: Place the starting node into the OPEN list.

Step 2: If the OPEN list is empty, Stop and return failure.

Step 3: Remove the node n , from the OPEN list which has the lowest value of $h(n)$, and places it in the CLOSED list.

Step 4: Expand the node n , and generate the successors of node n .

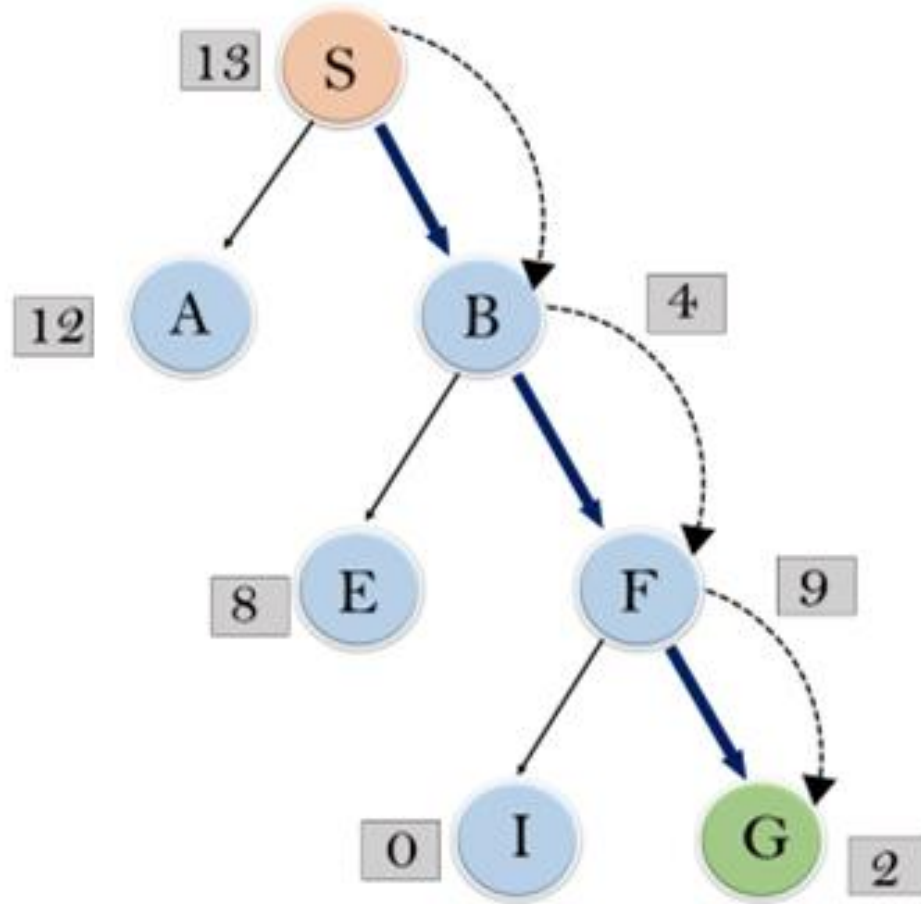
Step 5: Check each successor of node n , and find whether any node is a goal node or not. If any successor node is goal node, then return success and terminate the search, else proceed to Step 6.

Step 6: For each successor node, algorithm checks for evaluation function $f(n)$, and then check if the node has been in either OPEN or CLOSED list. If the node has not been in both list, then add it to the OPEN list.

Step 7: Return to Step 2.



1.) Best-first Search Algorithm (Greedy Search):



2.) A* Search Algorithm:

A* search is the most commonly known form of best-first search. It uses heuristic function $h(n)$, and cost to reach the node n from the start state $g(n)$. It has combined features of UCS and greedy best-first search, by which it solve the problem efficiently. A* search algorithm finds the shortest path through the search space using the heuristic function. This search algorithm expands less search tree and provides optimal result faster. A* algorithm is similar to UCS except that it uses $g(n)+h(n)$ instead of $g(n)$.

In A* search algorithm, we use search heuristic as well as the cost to reach the node. Hence we can combine both costs as following, and this sum is called as a fitness number.

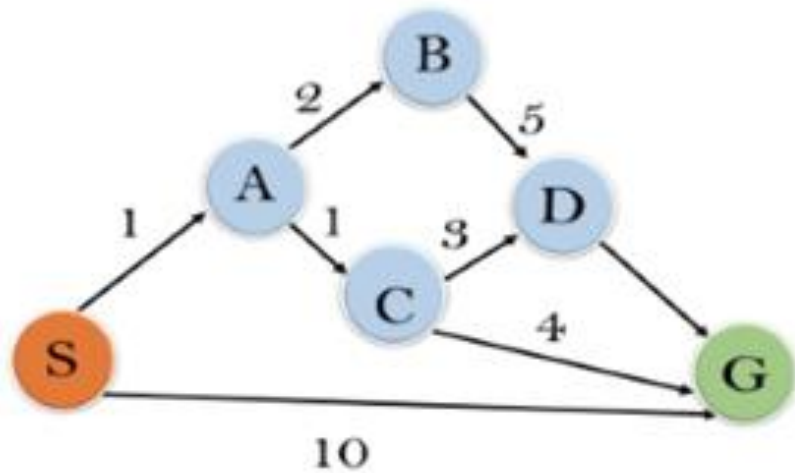


$$f(n) = g(n) + h(n)$$

Estimated cost of the cheapest solution.

Cost to reach node n from start state.

Cost to reach from node n to goal node



State	$h(n)$
S	5
A	3
B	4
C	2
D	6
G	0

Hill Climbing Algorithm in Artificial Intelligence

- ✓ Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem. It terminates when it reaches a peak value where no neighbor has a higher value.
- ✓ Hill climbing algorithm is a technique which is used for optimizing the mathematical problems. One of the widely discussed examples of Hill climbing algorithm is Traveling-salesman Problem in which we need to minimize the distance traveled by the salesman.



Hill Climbing Algorithm in Artificial Intelligence

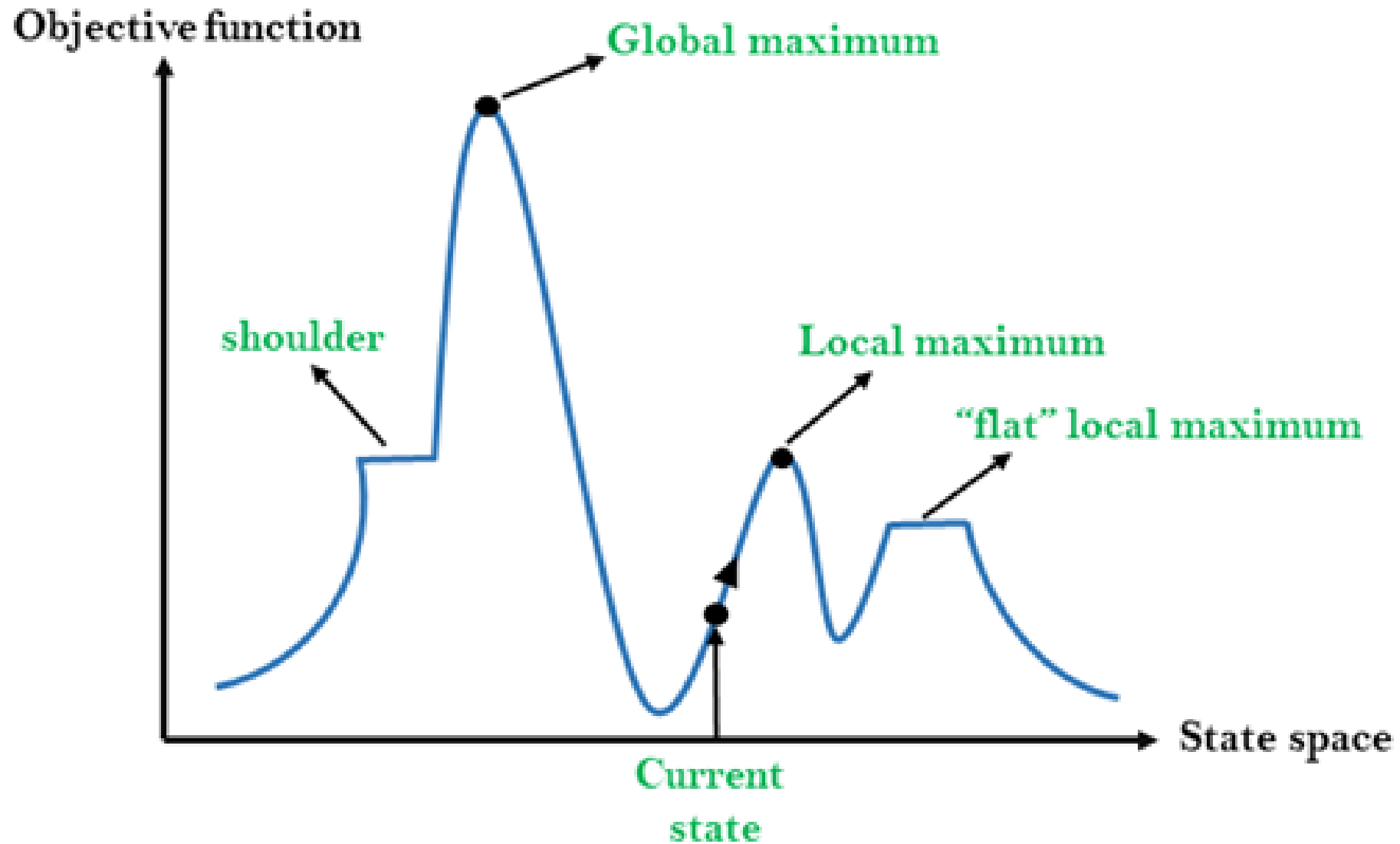
- ✓ It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that.
- ✓ A node of hill climbing algorithm has two components which are state and value.
- ✓ Hill Climbing is mostly used when a good heuristic is available.
- ✓ In this algorithm, we don't need to maintain and handle the search tree or graph as it only keeps a single current state.



Hill Climbing Algorithm in Artificial Intelligence

- ✓ Features of Hill Climbing:
- ✓ Generate and Test variant: Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
- ✓ Greedy approach: Hill-climbing algorithm search moves in the direction which optimizes the cost.
- ✓ No backtracking: It does not backtrack the search space, as it does not remember the previous states.





Hill Climbing Algorithm in Artificial Intelligence

Types of Hill Climbing Algorithm:

- ✓ Simple hill Climbing:
- ✓ Steepest-Ascent hill-climbing:
- ✓ Stochastic hill Climbing:

