**Artificial Intelligence:-**

It is the science & engineering of making intelligent machines and intelligent computer programs.

The tasks of AI is to use computers to understand the human behavior and not restrict to the biologically observable behaviors.

**Intelligence:-**

* It learns from the experience.
* It makes sense from unclear messages.
* It responds quickly & effectively to a new situation.
* It deals with the complex situations.
* It applies knowledge to change the environment.

**Tasks Easy for AI:-**

* It can behave cognitively, like, playing chess, proving theorems and some aspects of medical diagnosis.
* It can recognize faces and voices.
* It can suggest medicines and process huge calculations in just seconds.

**Tasks Difficult for AI:-**

* It cannot hunt.
* It cannot interpretate complex sensory information.
* It cannot work as a team.
* It cannot solve differential equations & database operations.

**AI vs Natural Intelligence:-**

Advantages of AI:-

* AI is permanent as long as computer programs remain unchanged.
* AI offers ease of duplication & distribution which means we can copy knowledge of one computer to another easily.
* AI is cheap.
* AI can be easily documented.

Natural Intelligence:-

* Natural (human) intelligence is creative.
* Natural intelligence use sensory information.
* Natural Intelligence is reasonable.
* Natural Intelligence is more powerful but limited.

**How AI Works:-**

* **Think Well** means develop a formal method of representation and bear reasoning, learning, memory and problem solving for guaranteeing an optimal solution.
* **Act Well** means generate an output for an input that must get the job done. Heuristic (Understanding for yourselves) offers a solution which is good most of the time.
* **Think like humans** means behave cognitively (like humans), focus on reasoning, it also tells us how something has been done. Its example is ELIZA ( a computer program) that interacts with humans like a doctor and suggest medicines.
* **Act like humans** means similar to human behavior and it is exemplified by the Turing test.

**Turing Test:-**

In Turing test, there are three rooms, one contains a human, the other contains the robot and the last contains an interrogator (the one who asks questions). Then,

* The interrogator can communicate with the other two.
* The interrogator tries to find which the person is and which the machine is.
* The machine tries to fool the interrogator into believing that it is the person.
* If the machine succeeds, then we conclude that the machine can think.

Examples of AI are:-

* Computer Vision
* Robotics
* Natural Language Processing (NLP)
* Expert Systems
* Games

**Problems and Problems Spaces:-**

There are 3 types of problems in AI:-

1. Single-agent path finding problems.
2. Two-player games.
3. Constraint satisfaction problems.

To solve a problem:-

* Define the problem precisely.
* Analyze the problem.
* Isolate and represent the task knowledge.
* Choose the best problem-solving technique and apply it.

**Single Agent Path Finding Problems:-**

* We have a single problem-solver making the decisions.
* The task is to find a sequence of steps from the initial location to the goal location.
* Its examples are Rubik’s Cube, Sliding-Tile Puzzle and Navigation – Travelling Salesman Problem.

**Two Player Games:-**

* In a two-player game, one must consider the moves of an opponent and the ultimate goal is a strategy for a win whenever possible.
* Its example is Chess computer game.

**Constraint Satisfaction Problems:-**

* We have a single-agent making all the decisions but we do not focus on the steps rather we will focus on the solution.
* The task is to find the state of a problem and all limitations of the problem are satisfied.
* Its example is Eight Queens Problem.

**The Problem Spaces:-**

* A problem space contains a set of states of problem and a set of operators that change the state.
* **State** is a symbolic structure that represents a single configuration of the problem in a sufficient detail to allow problem solving to proceed.
* **Operator** is a function that takes a state and maps it to another state.

**A Problem (Problem Instance):-**

It consists of:-

* A problem space
* An initial state
* A set of goal states
* A solution ( A path through initial state to final state )

**Representing States:-**

* At any moment, the relevant world is represented as a state.
* Initial State is represented with **S.**
* **State Transition** means change of state by an operation.
* An action can be applied if and only if its preconditions are met by the current state.
* **Goal State** is a state which passes the goal test.
* **Dead-end state** means a non-goal state to which no action is applicable.
* The state-space representation of a problem is a triplet (**I**, **O**, **G**) where **I** is an Initial state, **O** is a set of operators and **G** is goal state.
* A state space can be organized as a graph: Nodes: states in the space

Arcs: actions/operations

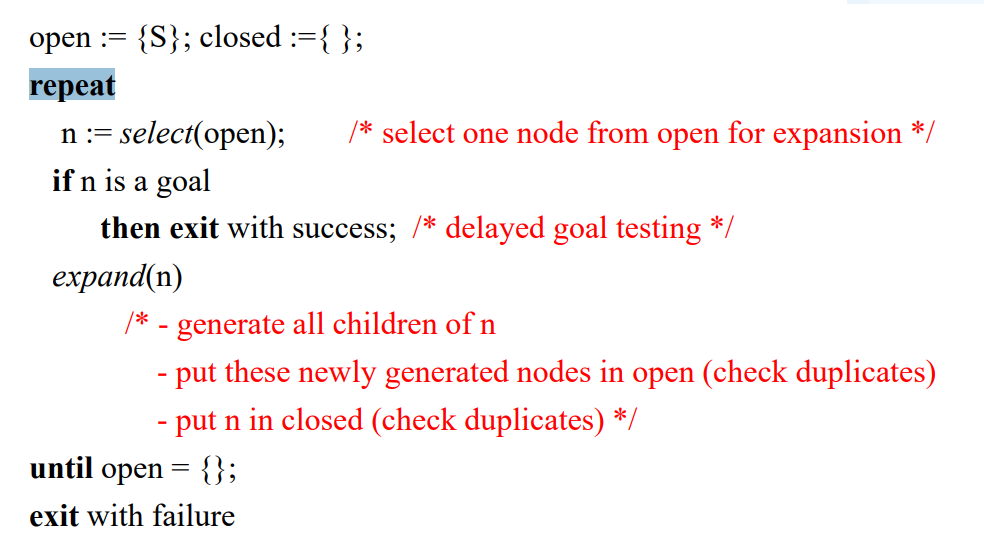
* The size of a problem is usually described in terms of:

The number of states.

Example: Chess has about states in a typical game.

**Searching Strategies:-**

A General State-Space Search Algorithm:-



Search Directions:-

* The goal of search procedure is to find a path through problem spaces from an initial state to a goal state. The search can proceed into two directions:-
* **Forward**, from that start states
* **Backward**, from the goal states

Forward Search:-

* It is also known as Data-directed / Data-driven reasoning / Forward Chaining and it uses knowledge and constraints found in the given data to search along lines known to be true.
* This search starts from the available information & tries to draw conclusion regarding the goal attainment.
* This process continues until it generates a path that satisfies the goal condition.

Backward Search:-

* It is also known as Goal directed / driven / Backward Chaining and it uses knowledge of the goal to guide the search.
* This search starts from expectations of what the goal is or what is to happen and then it seeks evidence that supports those expectations.
* Initially, we will find the goal to be solved then we will find rules to generate that goal and conditions under which these rules are true.
* These conditions are actually sub goals.
* The search continues until a path is generated that leads back to the facts of the problem.
* We use Backward Search (Goal-Driven Search):-
* When a goal is given or a goal can be easily formulated.
* When there are a large number of rules that match the facts of problem and thus can produce an increasing number of goals.
* When the problem data are not given.
* We use Forward Search (Data-Driven Search):-
* When all or most of the data is given in the initial state.
* When there are a large number of potential goals.
* When it is difficult to form a goal.

Evaluating Search Strategies:-

* **Completeness** guarantees to find a solution whenever it exits.
* **Time Complexity** means how long does it take to find a solution and it is measured in terms of the number of nodes expanded.
* **Space Complexity** means how much space is used by the algorithm and it is measured in terms of the maximum size that the **“OPEN”** list becomes during the search.
* **Optimality/Admissibility** means when a solution is found is it guaranteed to be an optimal one?

**Uninformed Search Strategies:-**

* It is also known as **Blind Search**.
* Breadth-First search
* Depth-First search
* Uniform-Cost search
* Depth-First Iterative Deepening search

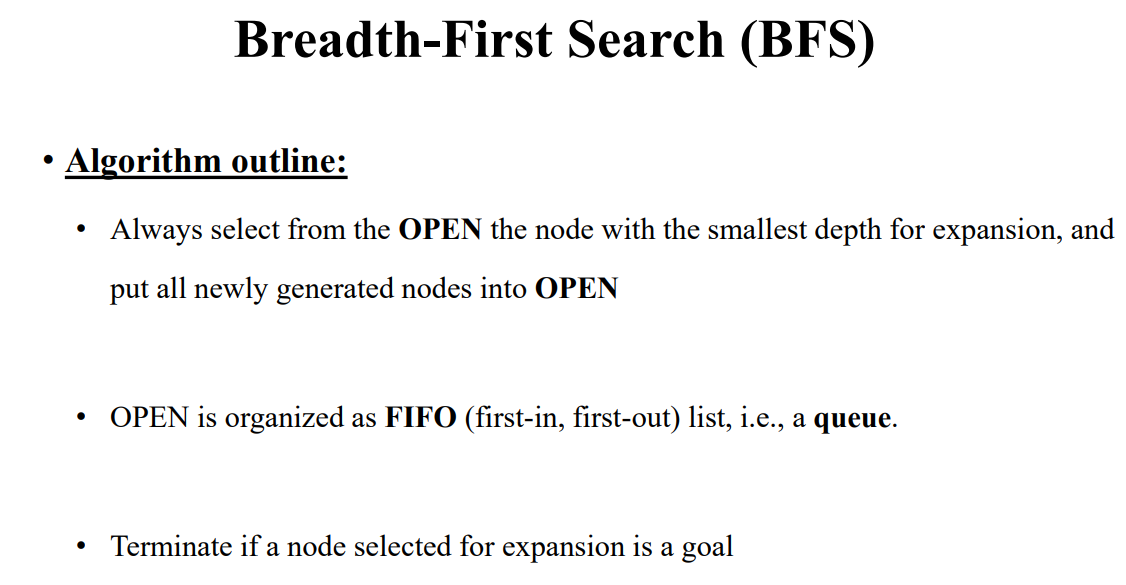
**Informed Search Strategies:-**

* It is also known as **Heuristic Search.**
* Hill climbing
* Best-first search
* Greedy Search
* Beam search
* Algorithm A
* Algorithm A\*

**Uninformed Search Strategies:-**

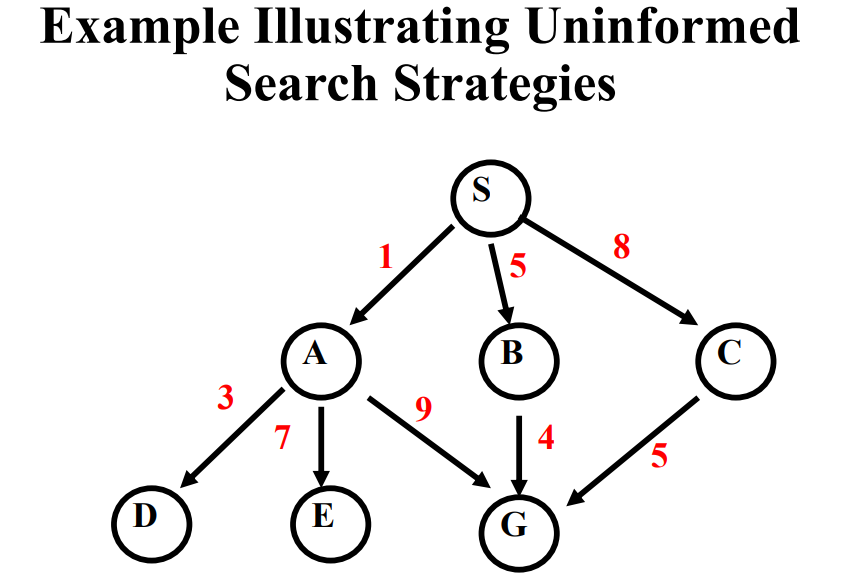
Breadth First Search Algorithm (BFS):-

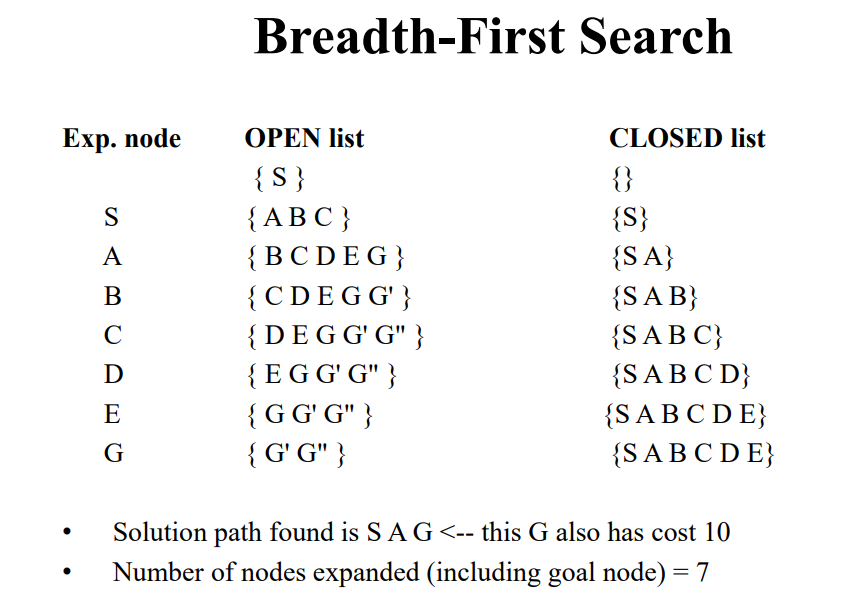
* In this algorithm, when a state is examined, all of its siblings are examined before any of its children.
* The space is searched level-by-level, proceeding all the way across one level before going down to the next level.
* BFS is suitable for problems with shallow solutions.

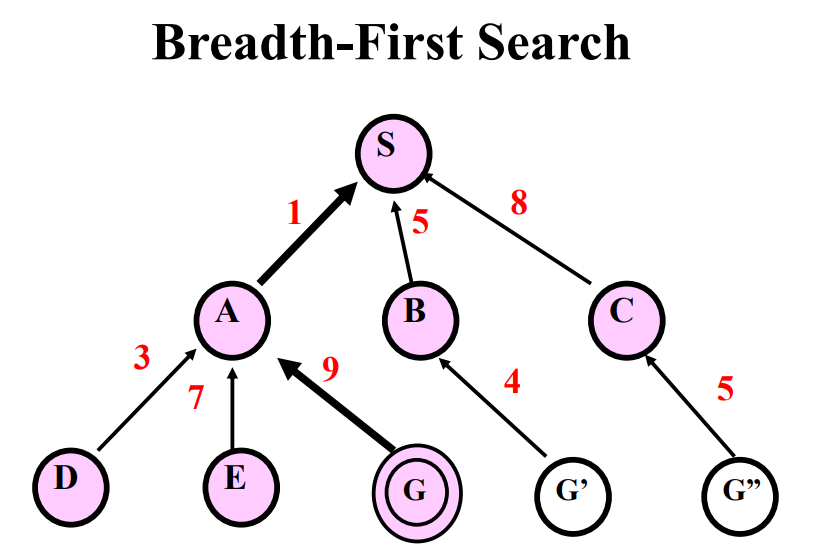


Properties of BFS:-

* It is complete.
* It is Optimal.
* It has exponential time & space complexity.
* It implements FIFO queue.

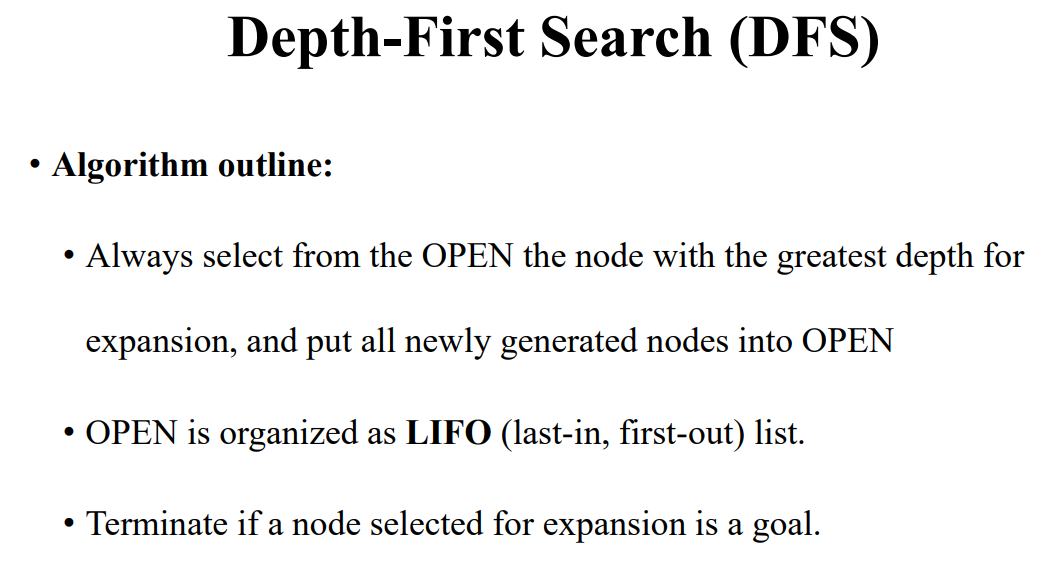






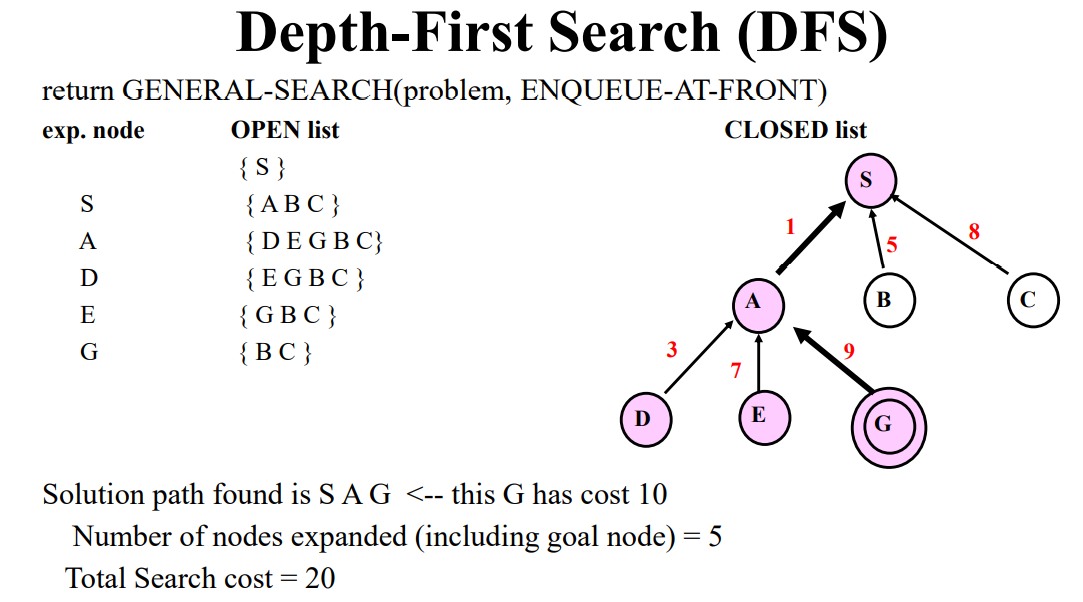
Depth First Search Algorithm (DFS):-

In this algorithm, when we examine a state, we will visit all of its children and their descendants before visiting its siblings.



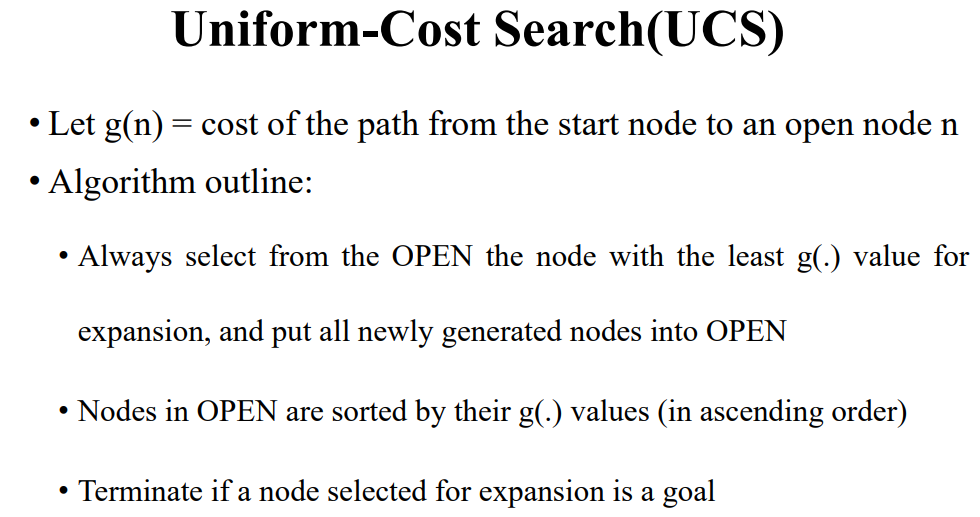
Properties of DFS:-

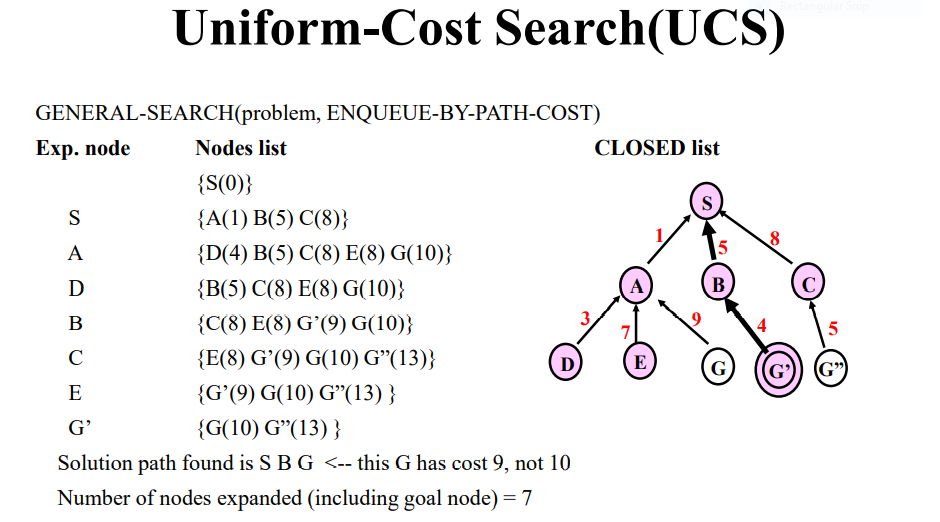
* It may not terminate without a “depth bound”.
* It is not complete.
* It has exponential time complexity and linear space complexity.
* Can find deep solutions quickly.
* It implements LIFO queue.



Uniform-Cost Search (UCS):-

This algorithm uses the lowest cumulative cost to find a path from the source to the destination. Nodes are expanded, starting from the root, according to the minimum cumulative cost.





Properties of UCS:-

* It is complete when cost of each action is not infinitesimal.
* It has exponential time and space complexity.
* In worst cases, it becomes BFS when all arcs cost the same.
* It is optimal/admissible.
* It implements priority queue.

