Search Algorithms in AI(Artificial Intelligence)

Md. Alamgir Hossain

Senior Lecturer

Dept. of CSE, Prime University

Mail: *alamgir.cse14.just@gmail.com*

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.

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.

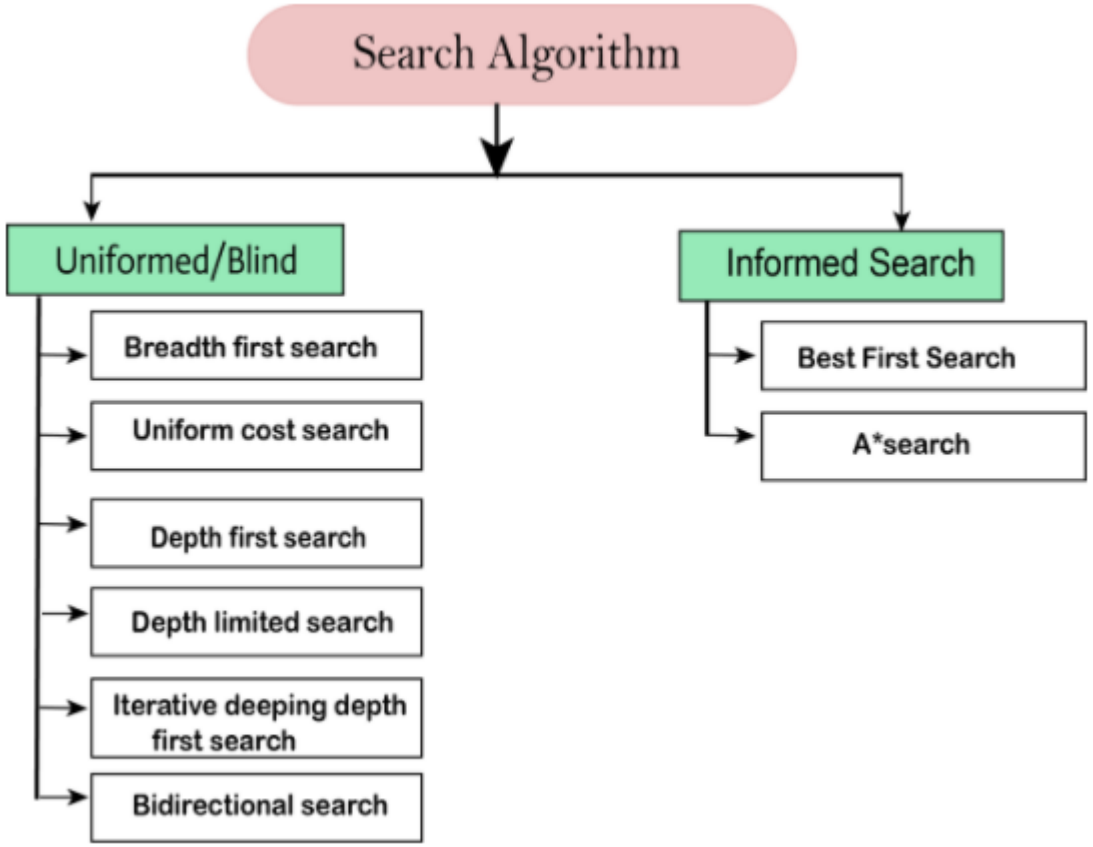
Properties of Search Algorithms

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

- Based on the search problems we can classify the search algorithms into uninformed (Blind search) search and informed search (Heuristic search) algorithms.



Informed Search Algorithms

- 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.

Informed Search Algorithms::Heuristic 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.
- 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.
- Admissibility of the heuristic function is given as: 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.

Informed Search Algorithms:: 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 until 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**

Informed Search Algorithms:: A* Search

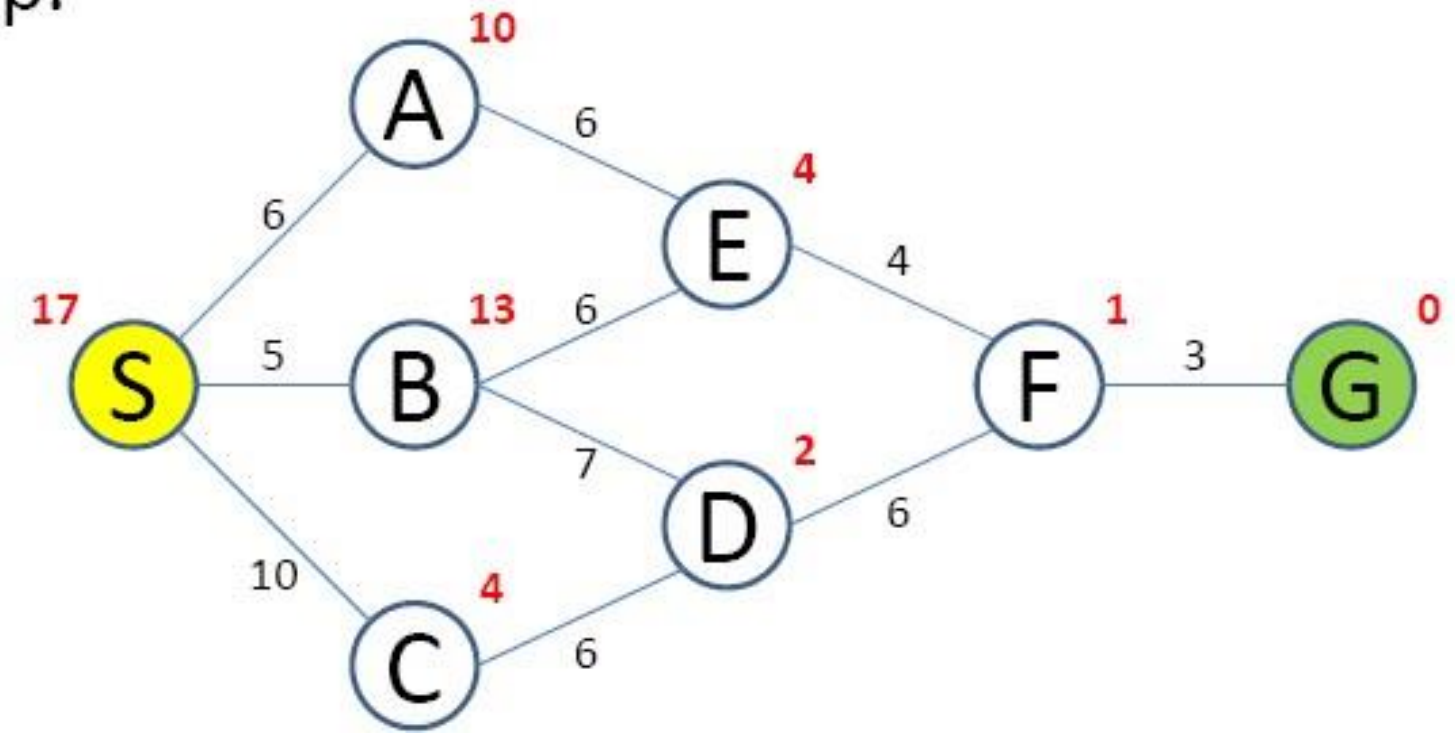
- A* search finds the ***shortest path through a search space to goal state using heuristic function***. This technique finds minimal cost solutions and is directed to a goal state called A* search.
- The A* algorithm also finds the ***lowest cost path between the start and goal state***, where changing from one state to another requires some cost.
- A* requires heuristic function to evaluate the cost of path that passes through the particular state. This algorithm is complete if the branching factor is finite and every action has fixed cost.
- It can be defined by following formula: f(n) = g(n) + h(n), Where,
  - ✓ g(n) = The actual cost path from start state to current state
  - ✓ h(n) = The actual cost path from current state to goal state
  - ✓ f(n) = The actual cost path from the start state to the goal state

Algorithm of A* Search

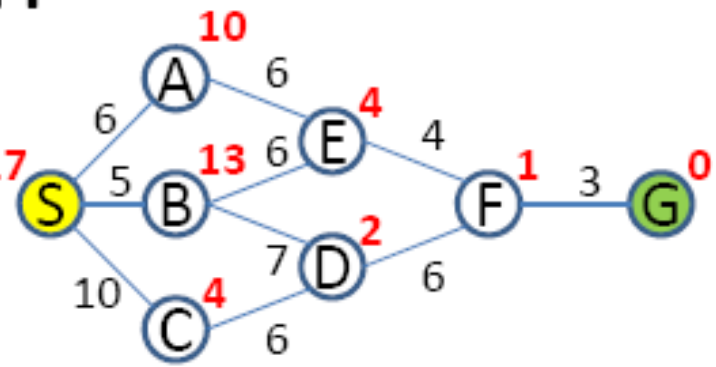
- **Step1:** Place the starting node in the OPEN list.
- **Step 2:** Check if the OPEN list is empty or not, if the list is empty then return failure and stops.
- **Step 3:** Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise
- **Step 4:** Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute evaluation function for n' and place into Open list.
- **Step 5:** Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.
- **Step 6:** Return to **Step 2**.

Problem

- Perform the A* Algorithm on the following figure. Explicitly write down the queue at each step.



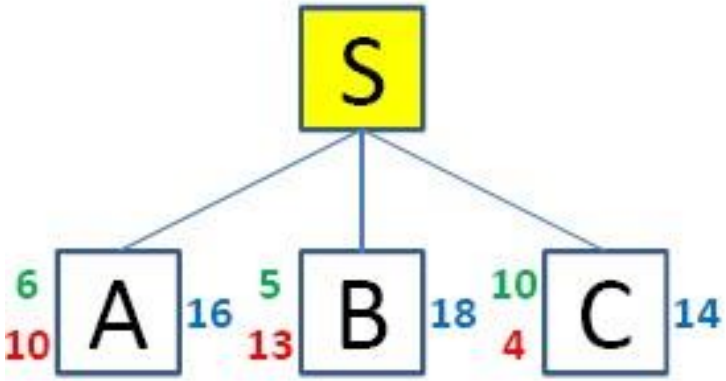
A* Search

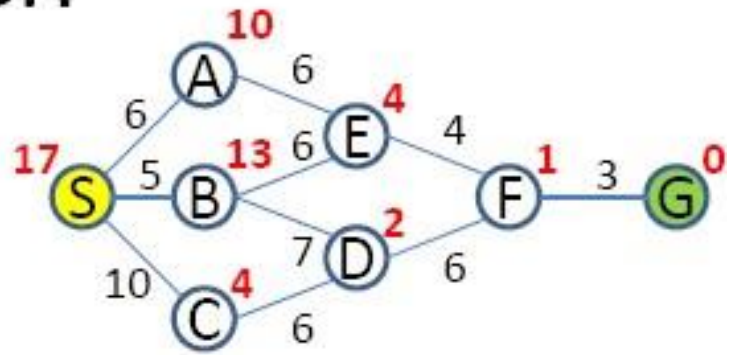


QUEUE:

S

A* Search





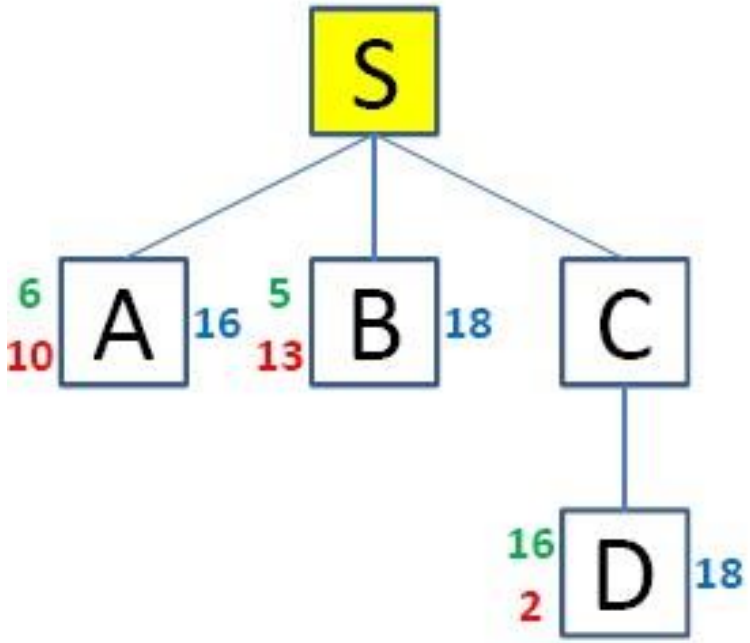
QUEUE:

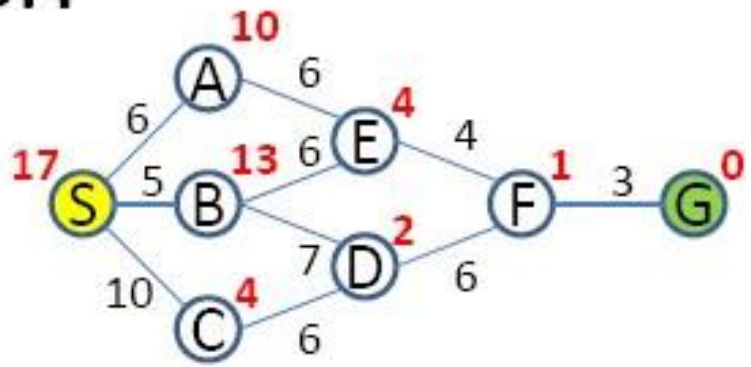
SC

SA

SB

A* Search





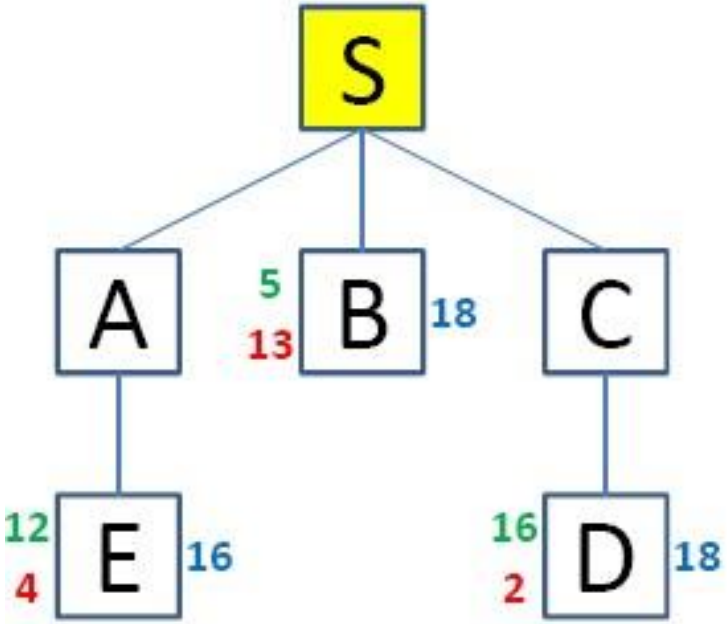
QUEUE:

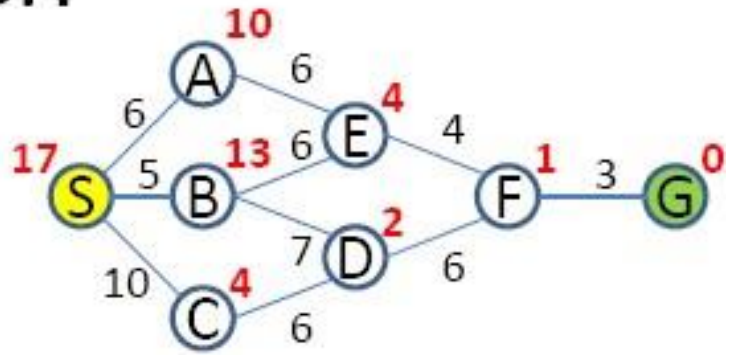
SA

SCD

SB

A* Search





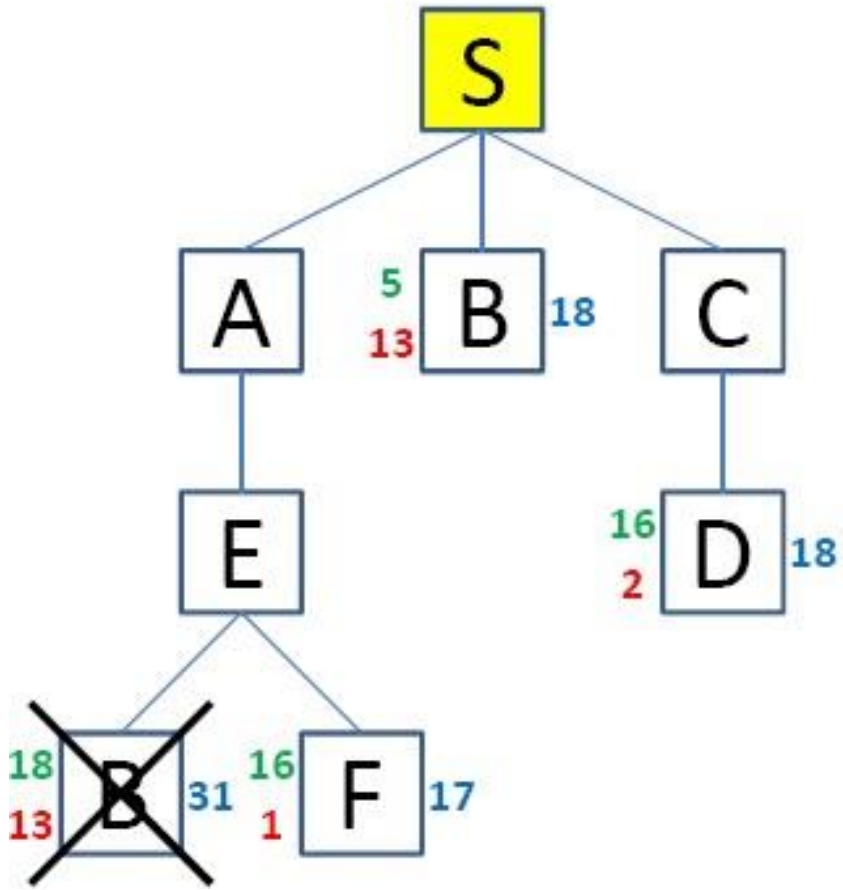
QUEUE:

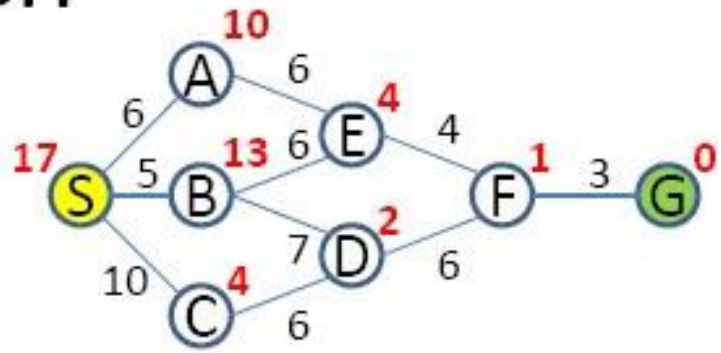
SAE

SCD

SB

A* Search





QUEUE:

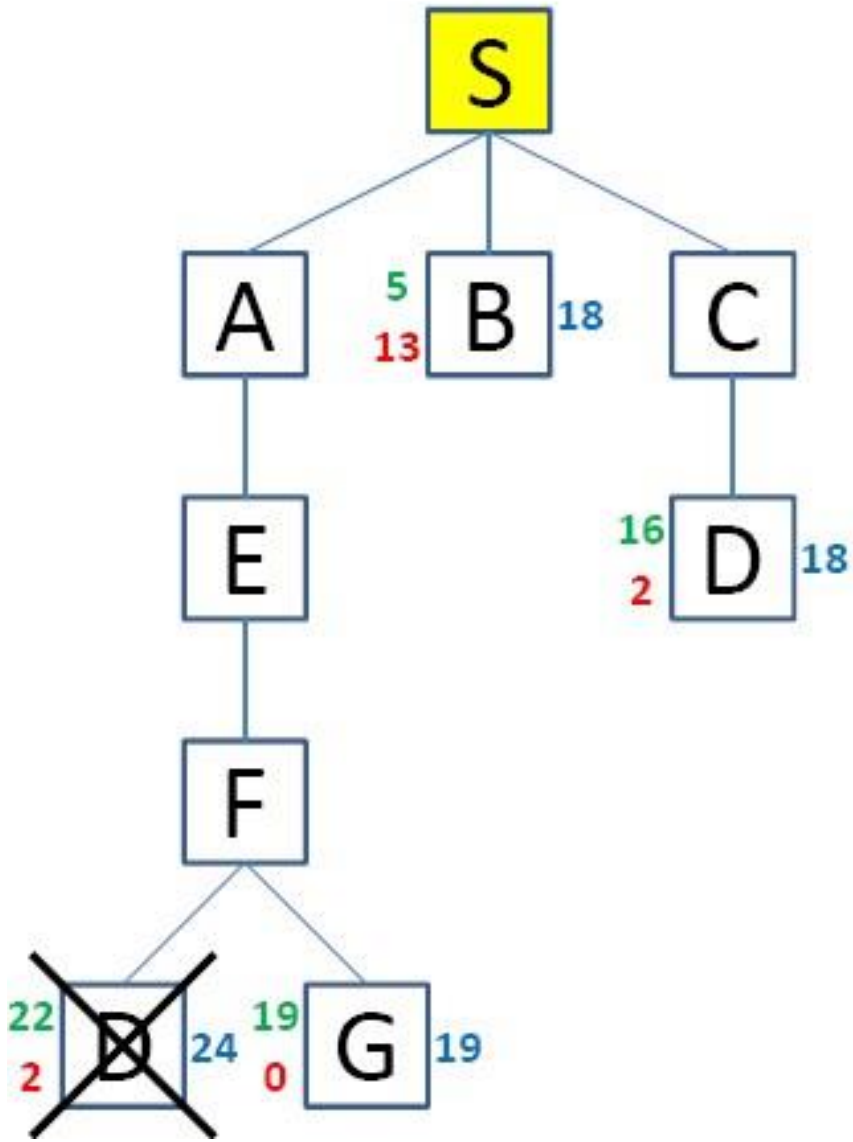
SAEF

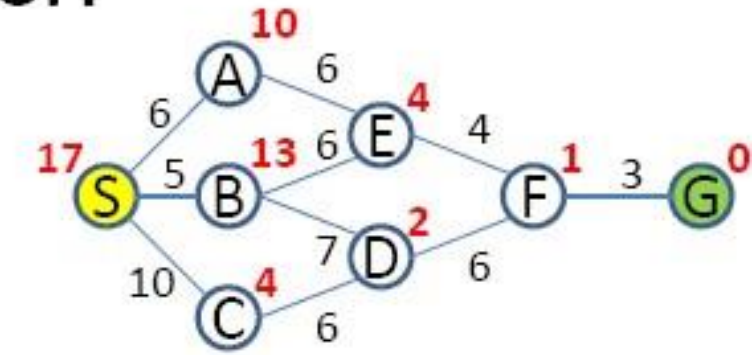
SCD

SB

SAEB

A* Search





QUEUE:

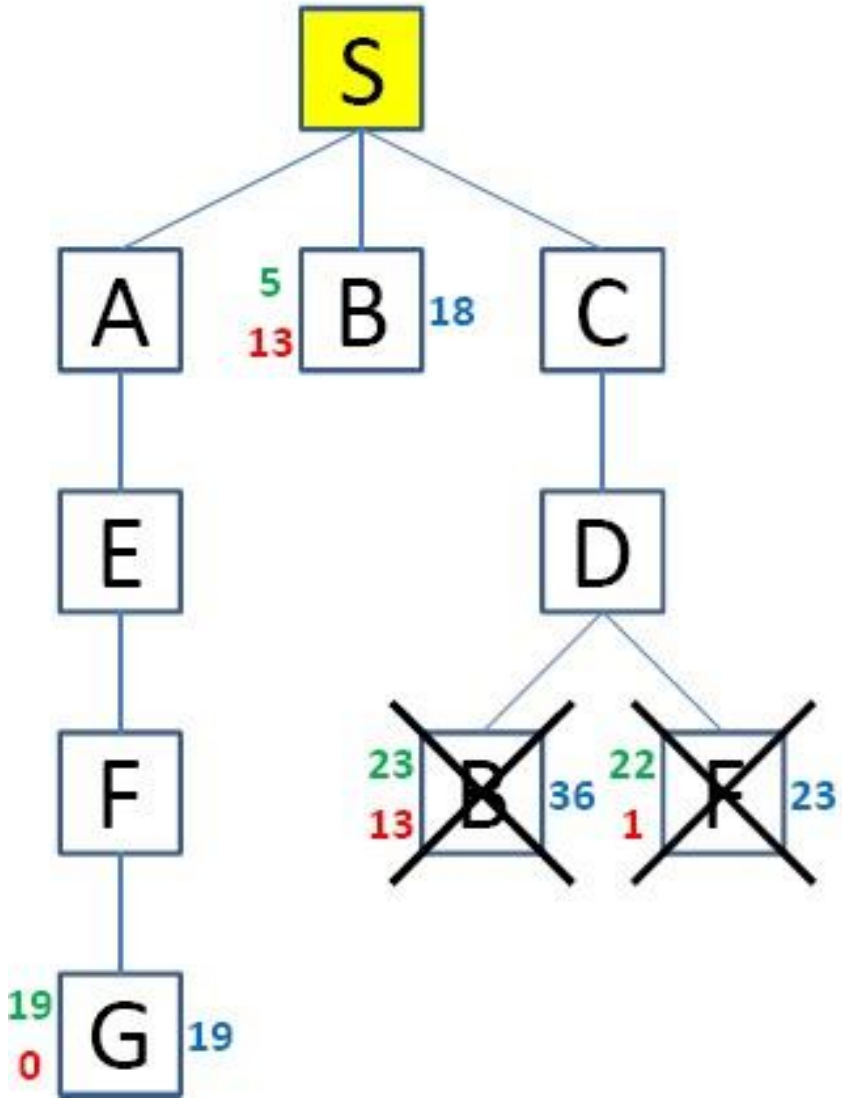
SCD

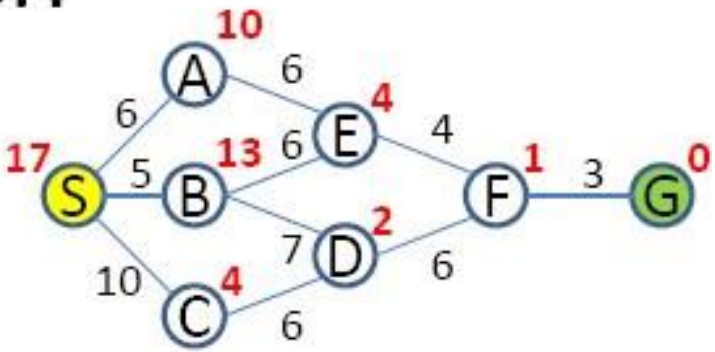
SB

SAEFG

SAEFD

A* Search





QUEUE:

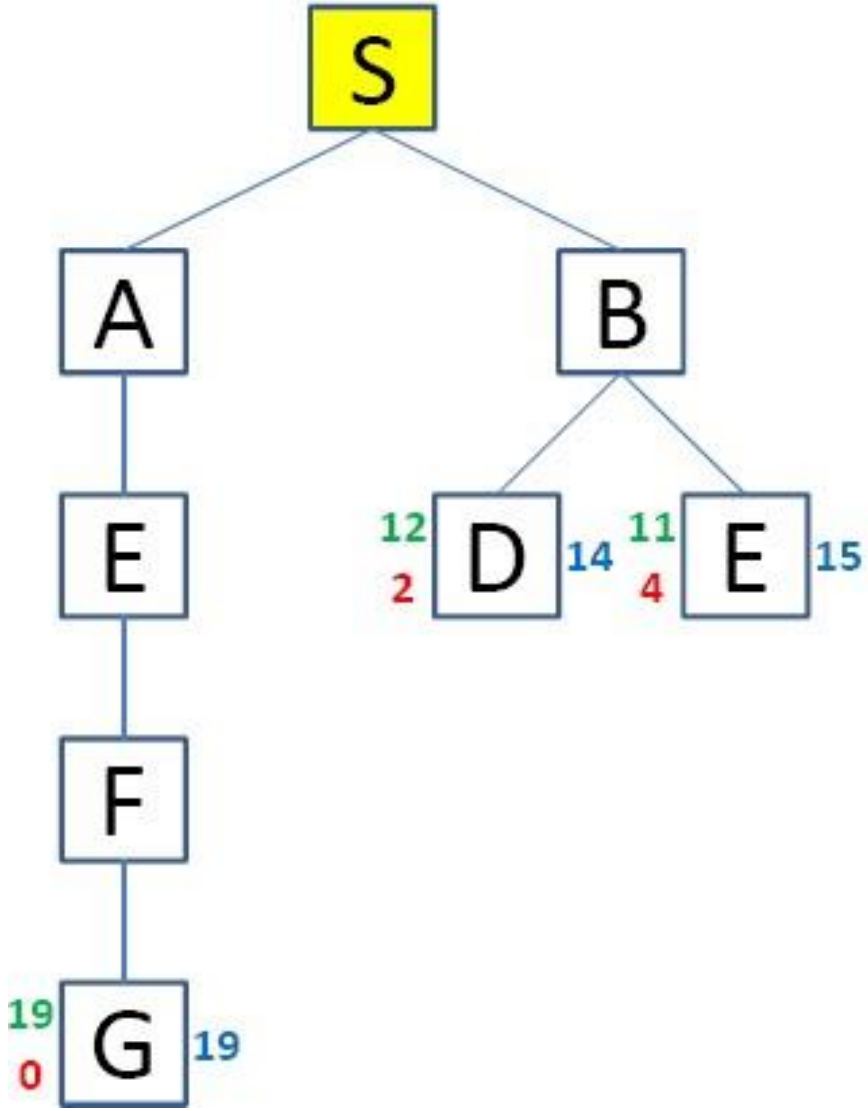
SB

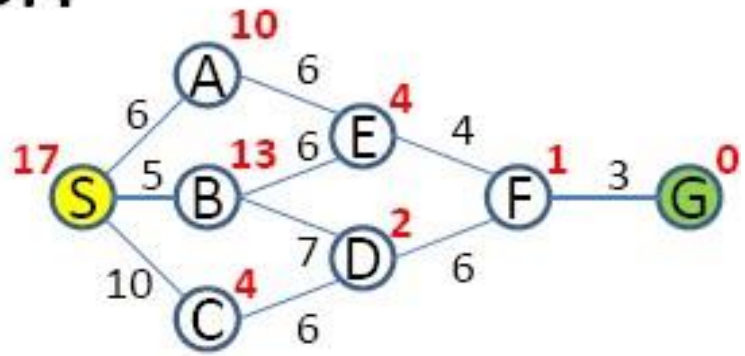
SAEFG

SCDF

SCDB

A* Search





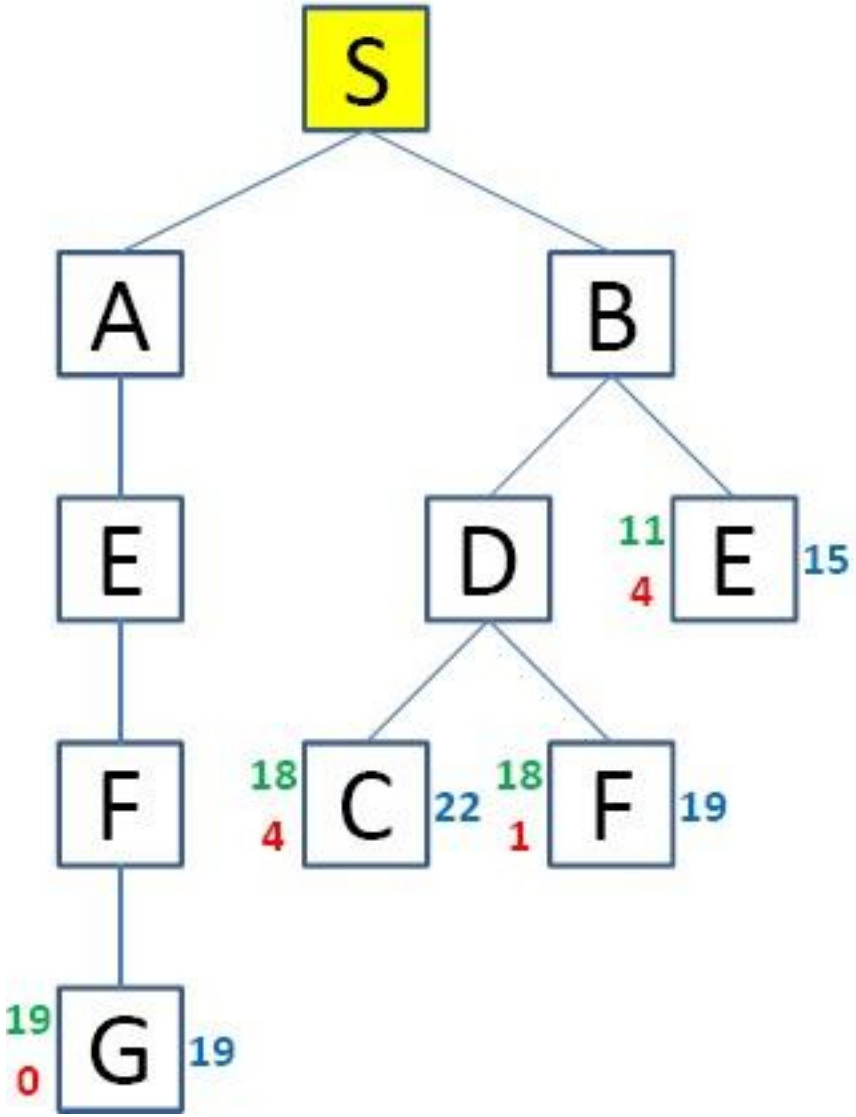
QUEUE:

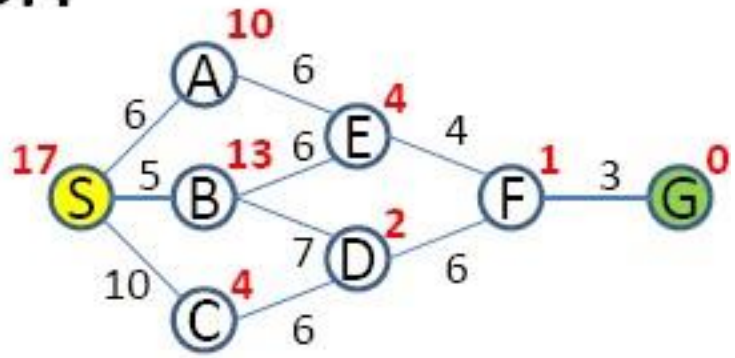
SBD

SBE

SAEFG

A* Search





QUEUE:

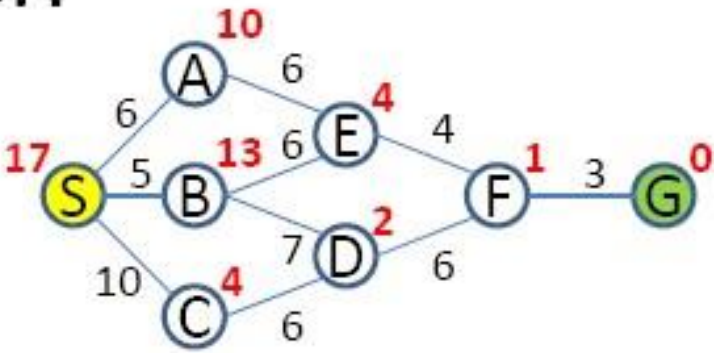
SBE

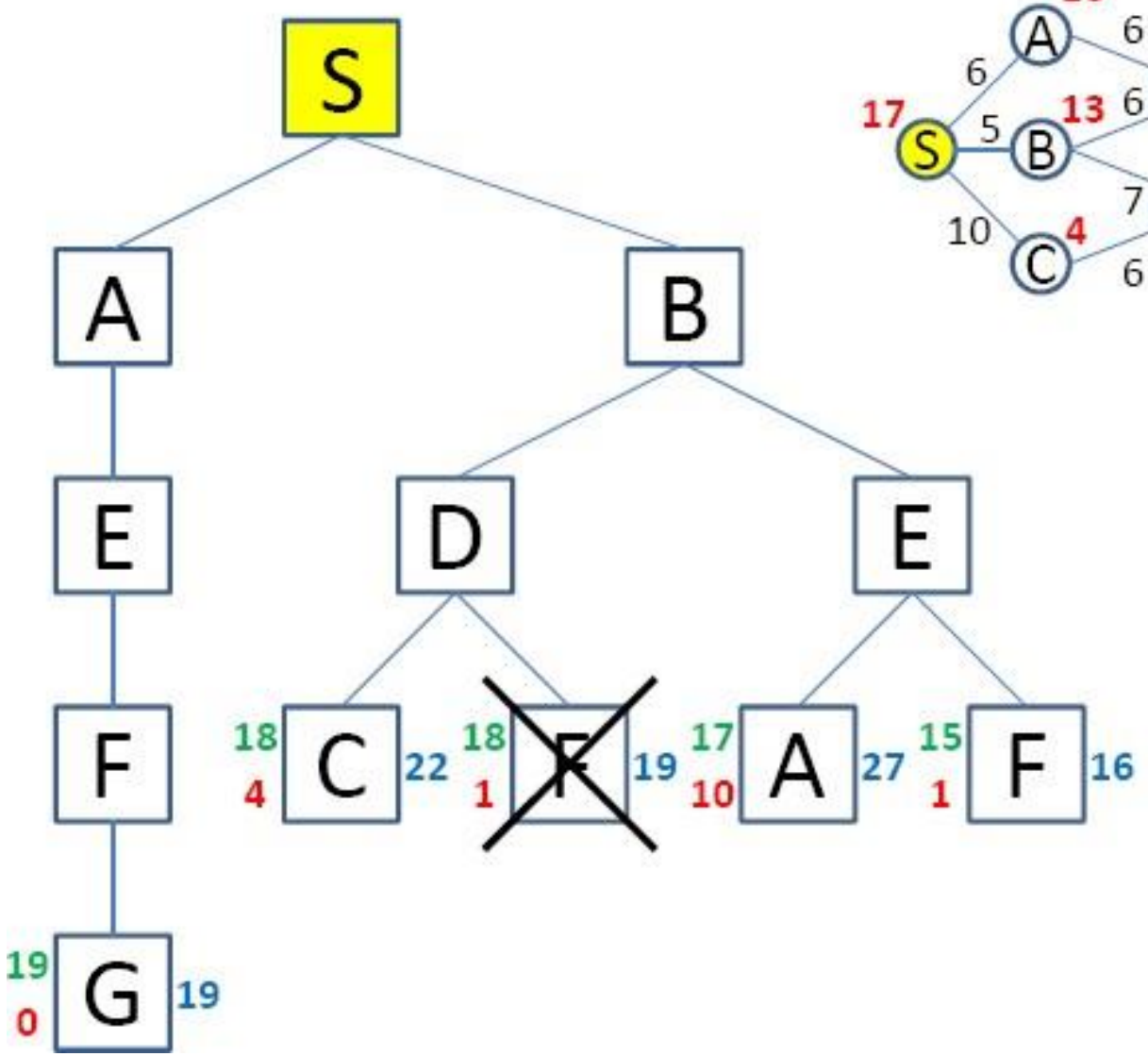
SBDF

SAEFG

SBDC

A* Search





QUEUE:

SBEF

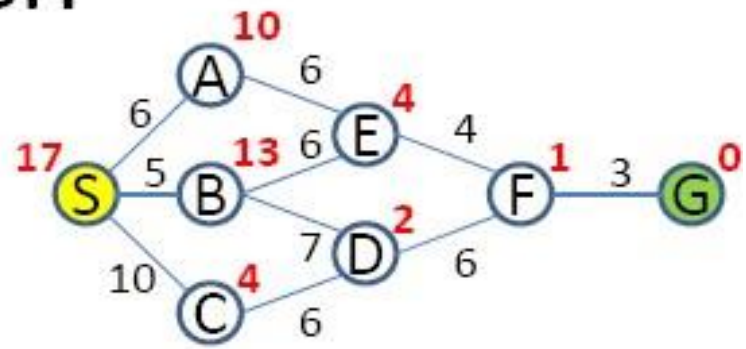
SAEFG

SBDF

SBDC

SBEA

A* Search



QUEUE:

SBEFG

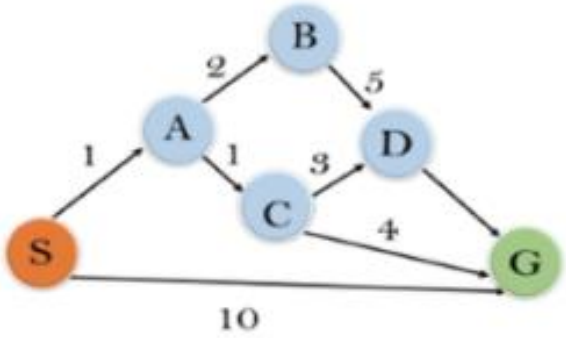
SAEFG

SBDC

SBEFD

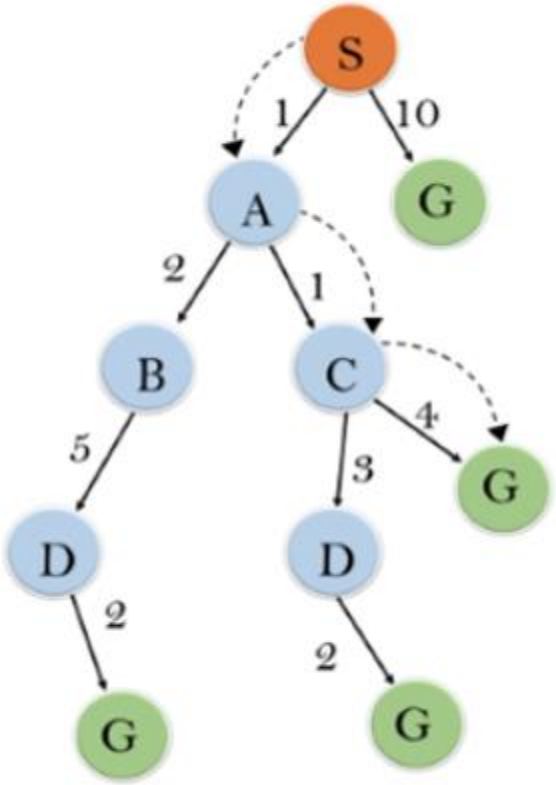
SBEA

A* Search:: Example-02



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

Solution



Solution

- **Initialization:** {(S, 5)}
- **Iteration1:** {(S--> A, 4), (S-->G, 10)}
- **Iteration2:** {(S--> A-->C, 4), (S--> A-->B, 7), (S-->G, 10)}
- **Iteration3:** {(S--> A-->C--->G, 6), (S--> A-->C--->D, 11), (S--> A-->B, 7), (S-->G, 10)}
- **Iteration 4** will give the final result, as **S--->A--->C--->G** it provides the optimal path with cost 6.

Points to Remember for A* Search Algorithm

- A* algorithm returns the path which occurred first, and it does not search for all remaining paths.
- The efficiency of A* algorithm depends on the quality of heuristic.
- A* algorithm expands all nodes which satisfy the condition f(n).

A* algorithm is complete as long as:

- Branching factor is finite.
- Cost at every action is fixed.

Points to Remember for A* Search Algorithm

A* search algorithm is optimal if it follows below two conditions:

- **Admissible:** the first condition requires for optimality is that h(n) should be an admissible heuristic for A* tree search. An admissible heuristic is optimistic in nature.
- **Consistency:** Second required condition is consistency for only A* graph-search.
- If the heuristic function is admissible, then A* tree search will always find the least cost path.
- **Time Complexity:** The time complexity of A* search algorithm depends on heuristic function, and the number of nodes expanded is exponential to the depth of solution d. So the time complexity is O(b^d), where b is the branching factor.
- **Space Complexity:** The space complexity of A* search algorithm is **O(b^d)**

Advantages & Disadvantages of A* Search Algorithm

Advantages:

- A* search algorithm is the best algorithm than other search algorithms.
- A* search algorithm is optimal and complete.
- This algorithm can solve very complex problems.

Disadvantages:

- It does not always produce the shortest path as it mostly based on heuristics and approximation.
- A* search algorithm has some complexity issues.
- The main drawback of A* is memory requirement as it keeps all generated nodes in the memory, so it is not practical for various large-scale problems.

Thank You