Assignment No: 2

1. **Title of Assignment:** Implement A* Algorithm for 8 puzzle game search problems.
2. **Prerequisite:** Basic knowledge of Graph, Tree , informed search, uninformed search, best first search etc.
3. **Objective:** In this experiment, we will be able to do the following:
   - To understand Informed Search Strategies.
   - To make use of Graph and Tree Data Structure for implementation of Informed Search strategies.
   - Study how A* Algorithm is useful for implementation of 8 puzzle game search problems

   Outcome: Successfully able to implement 8 puzzle game search problem using A* Algorithm
4. Software and Hardware Requirement:

Open Source C++ Programming tool like G++/GCC, python, Java and Ubuntu.

5. **Relevant Theory / Literature Survey:**

Informed search

- Informed search algorithm contains an array of knowledge such as how far we are from the goal, path cost, how to reach the goal node, etc.
- This knowledge helps agents to explore less of the search space and find the goal node.
- The informed search algorithm is more useful for large search spaces.
- Informed search algorithms use the idea of heuristic, so it is also called Heuristic search

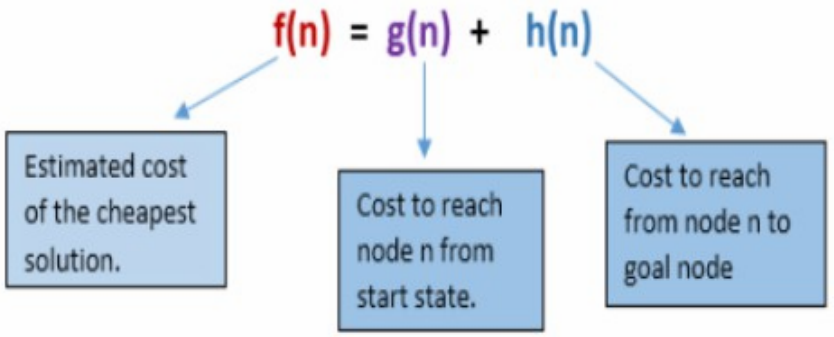
Heuristics function:

- 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 the 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.

A* Search Algorithm:

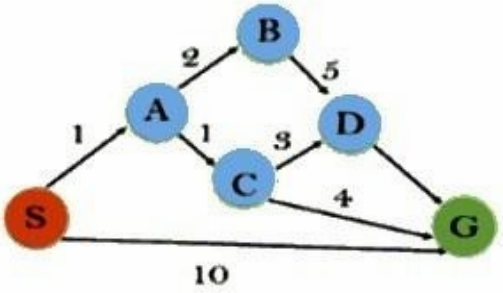
- A* search is the most commonly known form of best-first search.
- It uses the heuristic function h(n), and costs to reach the node n from the start state g(n).
- It has combined features of UCS and greedy best-first search, by which it solves the problem efficiently.
- A* search algorithm finds the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



In this example, we will traverse the given graph using the A* algorithm. The heuristic value of all states is given in the below table so we will calculate the f(n) of each state using the formula f(n)= g(n) + h(n), where g(n) is the cost to reach any node from start state.

Here we will use OPEN and CLOSED list.



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

Initialization: {(S, 5)}

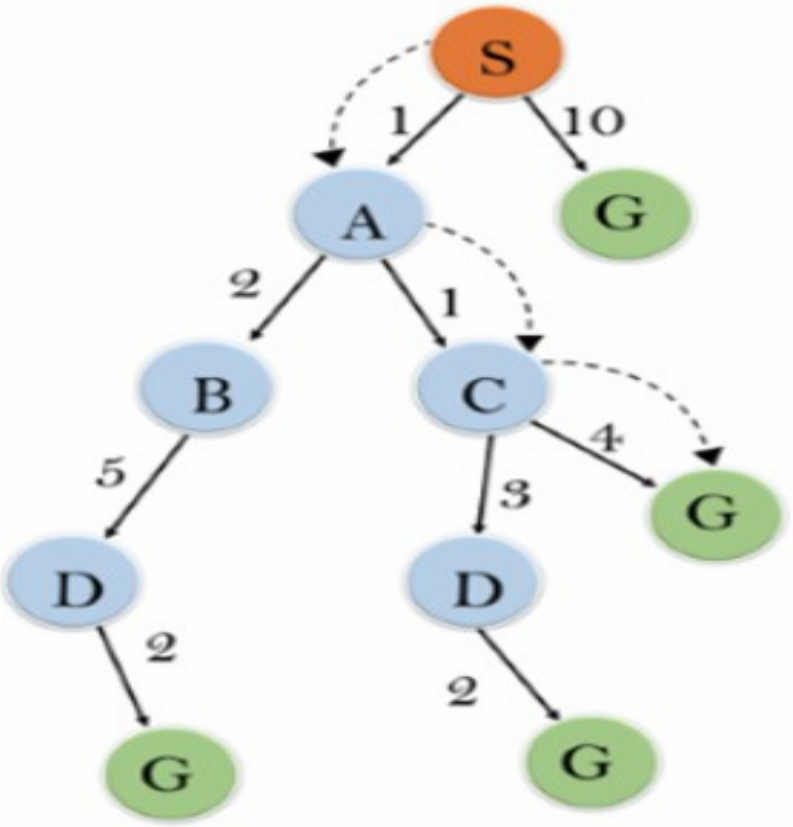
Iteration1: {(S--> A, 4), (S-->G, 10)}

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.

Solution:



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.

A* search Algorithm Disadvantages:

- 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.

Complete: A* algorithm is complete as long as:

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

Optimal: 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.

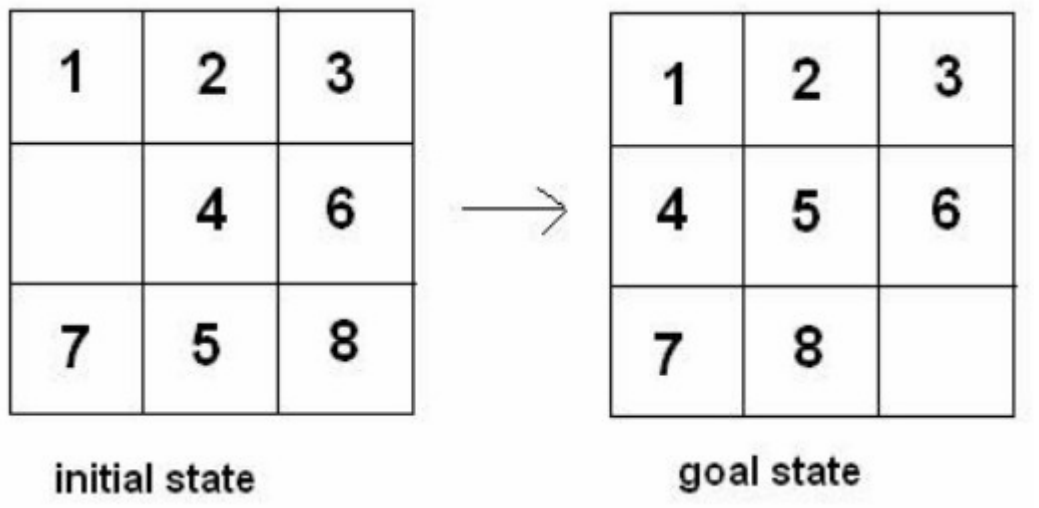
If the heuristic function is admissible, then A* tree search will always find the least cost path.

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)**

8 Puzzle Algorithm:-

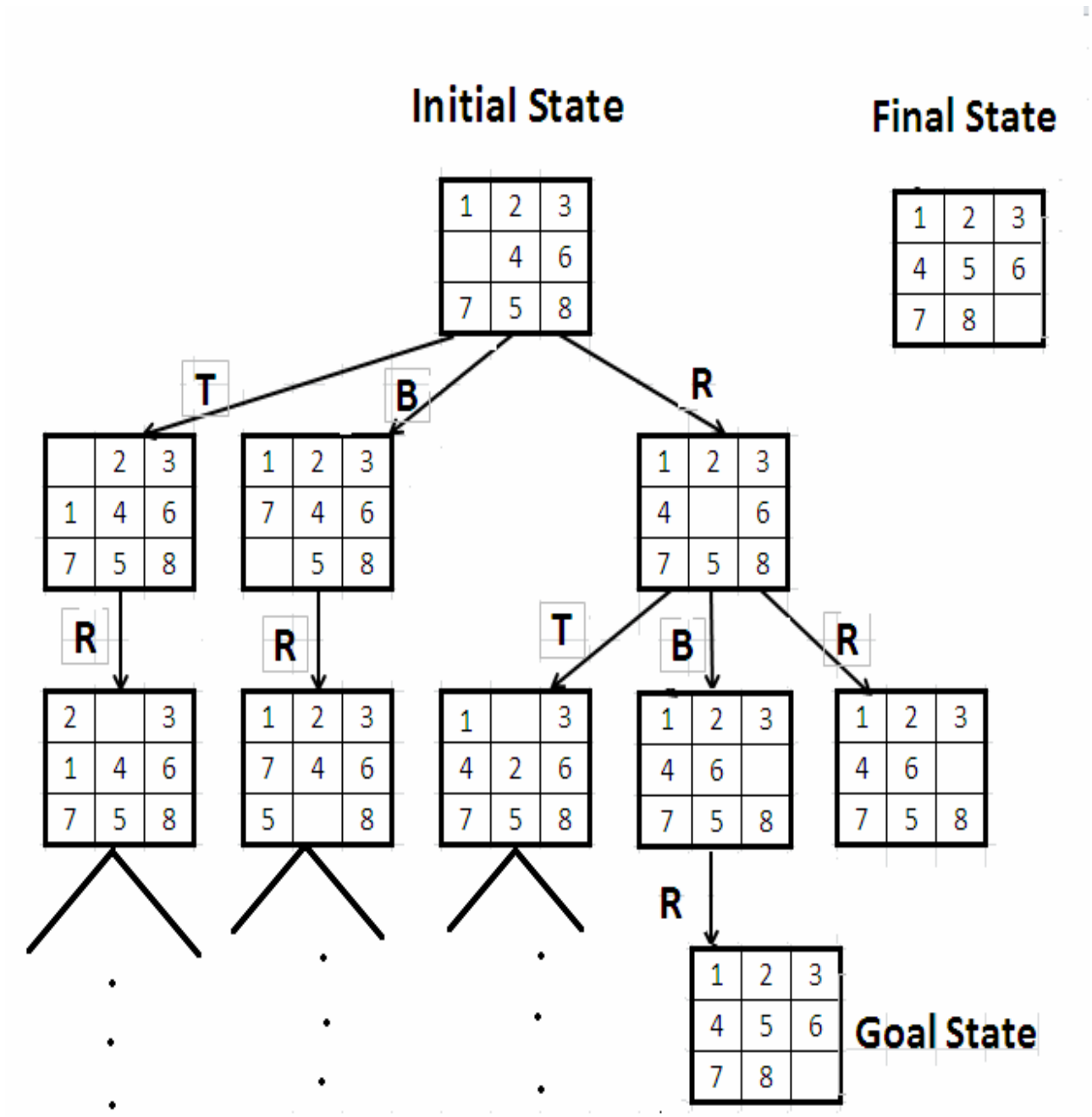
The 8-puzzle problem is a puzzle invented and popularized by Noyes Palmer Chapman in the 1870s. It is played on a 3-by-3 grid with 8 square blocks labeled 1 through 8 and a blank square. Your goal is to rearrange the blocks so that they are in order. You are permitted to slide blocks horizontally or vertically into the blank square.



There are a number of ways by which we can solve 8 puzzle problems.

- Solution without Heuristic Function
- Solution A* Algorithm

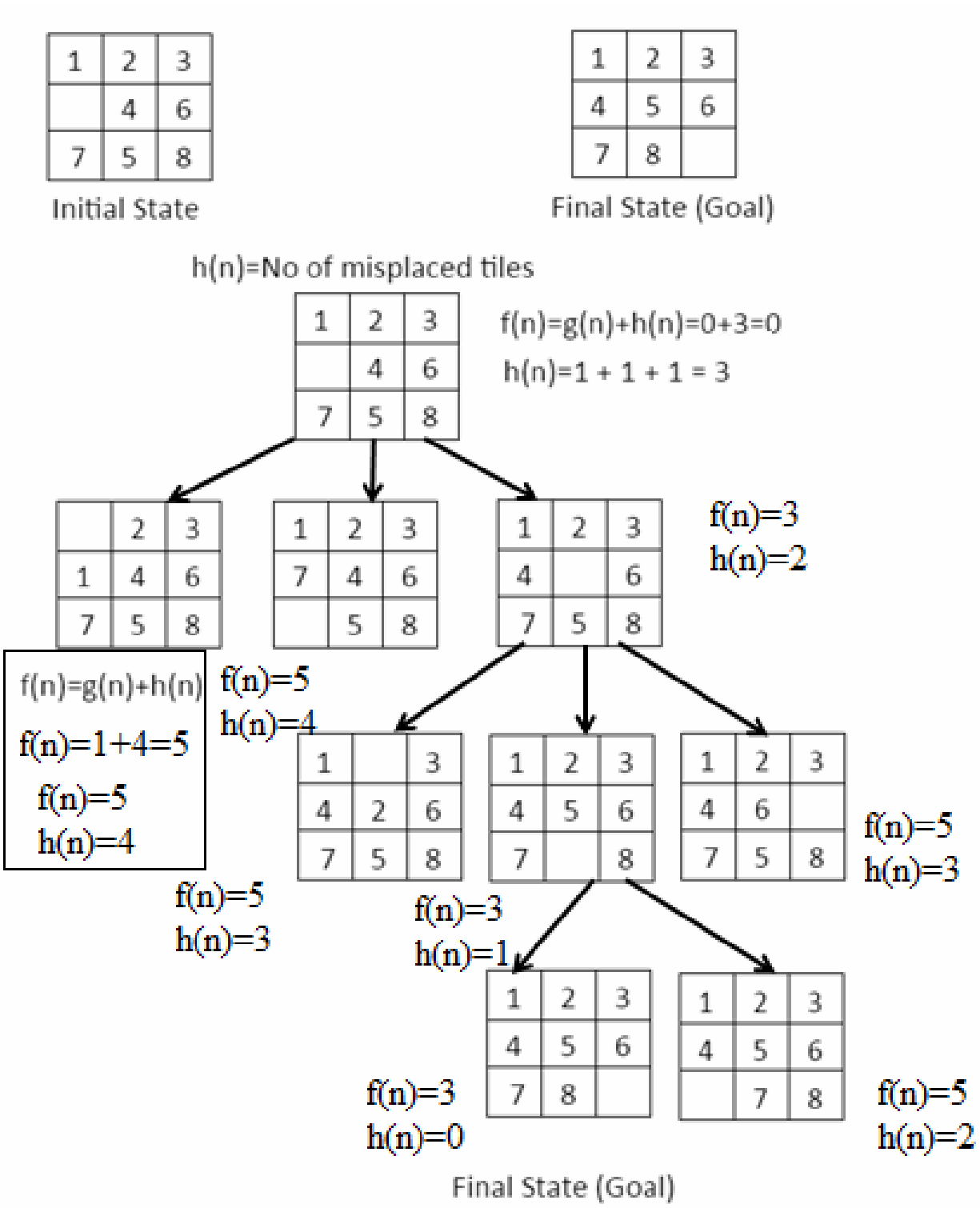
Solution without Heuristic Function



Disadvantages

need to explore each node and in case of failure need to generate its child which is a very time consuming as well as space consuming process.

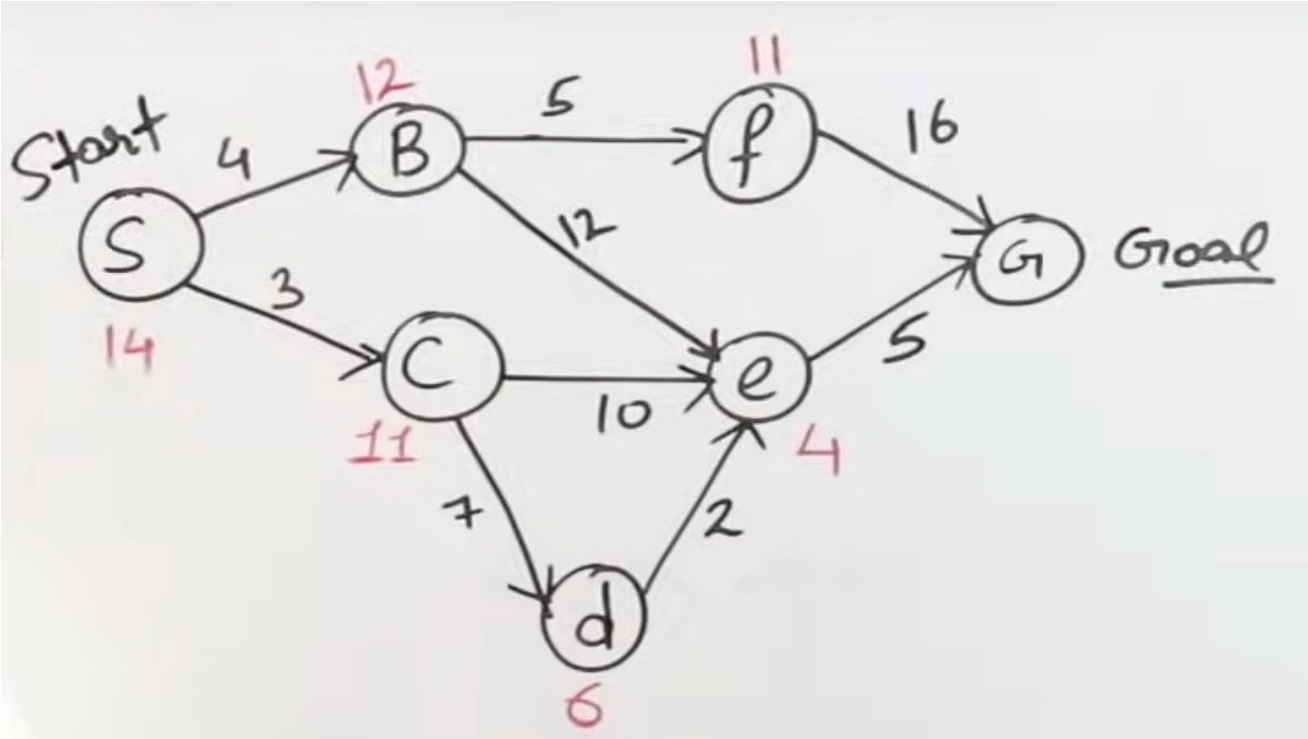
Solution A* Algorithm



6. Questions:

Q 1: Differentiate between Best first search and A* algorithm.

Q 2: Solve this problem using A* algorithm



Q 3: What is the drawback to solve 8 Puzzle problem with a non-heuristic method?

7. **Conclusion:**

In This way we have studied informed search strategy, how to calculate heuristic function and implementation of 8 puzzle game search problems using A* Algorithm.