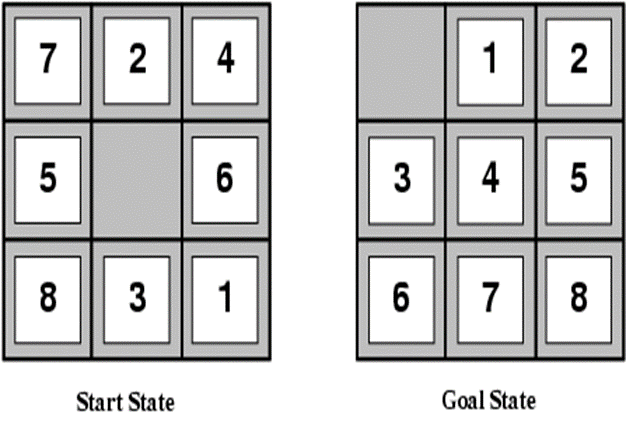
**Informed Search:-**

Also known as heuristic search, is a class of search algorithms that utilize problem-specific information or heuristics to guide the search process.

**Heuristic Functions:-**

Denoted as h(n), estimates the cost of the cheapest path from the state at node n to the goal state. It provides an informed estimate of how close a state is to the goal.

E.g., for the 8-puzzle:

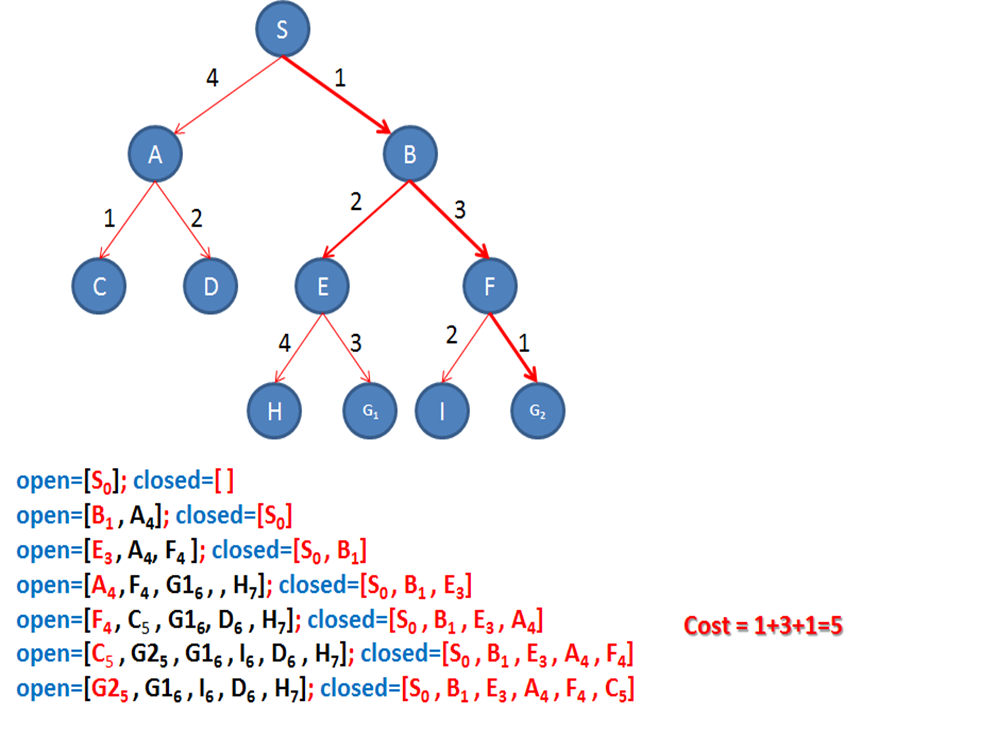
h1(n) = number of misplaced tiles

h2(n) = total Manhattan distance (i.e., no. of squares from desired location of each tile) (cityblock, D4 distance)

h1(S) = 8

h2(S) = 3+1+2+2+2+3+3+2 = 18

**Best First Search** prioritizes nodes based on an evaluation function, aiming to find the most promising path to the goal state. However, it may encounter issues such as infinite loops and suboptimal solutions, depending on the choice of evaluation function and the structure of the search space.

Algorithm:-

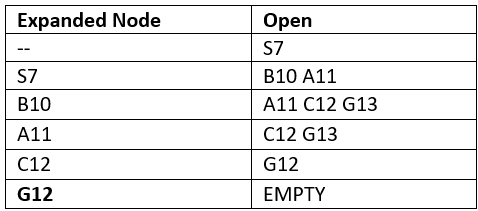
Initialize a priority queue "fringe" with the initial state.

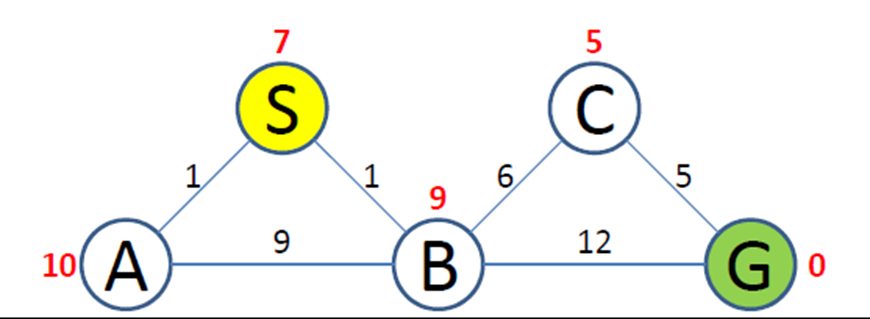
Loop until the fringe is empty (failure):

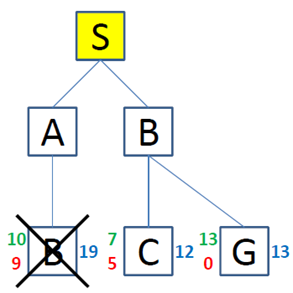
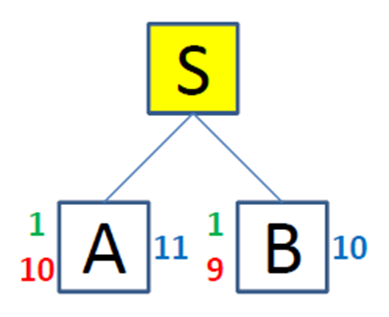
Remove the node with the lowest evaluation function value (f-value) from the fringe.

If the removed node is a goal state, return the path from the initial state to the goal.

Generate successors of the removed node and add them to the fringe based on their f-values.

A\* efficiently finds the optimal path from a start node to a goal node by considering both the cost of reaching a node from the start node (g(n)) and the estimated cost from the node to the goal node (h(n)).





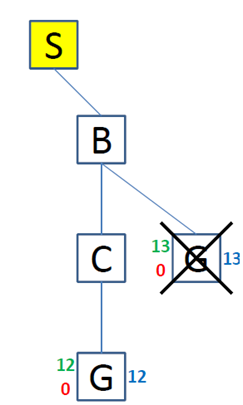
Step 1: Start node S, Successors A & B

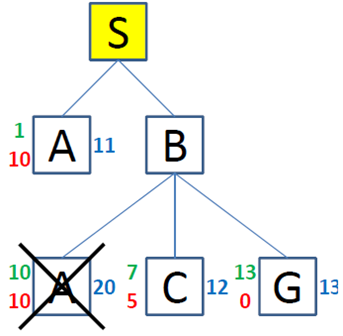
S-A => f(A) = 1+10=11

**S-B => f(B)= 1+9=10**

**Step 2:** S-B, Successors of B are A, C, G

S-B-A => f(A) = (1+9)+10=20....Discard

S-B-C => f(C)= (6+1)+5=12

S-B-G => f(G)= (1+12)+0=13

S-A path from step 1 chosen as min f(n)

**Step 3:** S-A, Successors is B

S-A-B => f(B) = (1+9)+9= 19 ....Discard

S-B-C chosen as min f(n)

**Step 4:** S-B-C , Successors is G

S-B-C-G => f(G) = (1+6+5)+0= 12 ......Discard G13

SBCG optimal path from S to G

**Cryptarithmetic Problem** is a type of constraint satisfaction problem where the game is about digits and its unique replacement either with alphabets or other symbols. In cryptarithmetic problem, the digits (0-9) get substituted by some possible alphabets or symbols. The task in cryptarithmetic problem is to substitute each digit with an alphabet to get the result arithmetically correct.  
There should be only one carry forward while performing the addition operation on a problem.

The problem can be solved from both sides, i.e., lefthand side (L.H.S), or righthand side (R.H.S)

**S E N D + M O R E = M O N E Y**

Where S=9, M=1, E=5, O=0, N=6, R=8, D=7

**Constraint satisfaction problem (CSP)** involves identifying constraints and finding solutions that satisfy those constraints.

Various algorithms, including backtracking, forward checking, and local search, can be used to search the state space and find a solution to the CSP.

Applications: including scheduling, resource allocation, & automated reasoning, Sudoku.

Solving it involves searching for solution in state space of possible assignments to variables.

