Submitted By:

Suraj Pandey
MT18025
Assignment 1

Observation and Analysis

For Part 1 Assignment:

I used the tree approach in which each state has at the most four children as according to move.

I have visited listed in which I have record of states I have traced.

Now in Program:

Case 1:

BFS Search Algorithm:

BFS is worked as level wise ,it first trace all states in level 1, level 2 and so on, until it get final state or goal state.

So, For this I used queue in which as states come queued into queue and the deleted as per required and if it is visited firstly then ,inserted also into visited to take trace of states in searching .

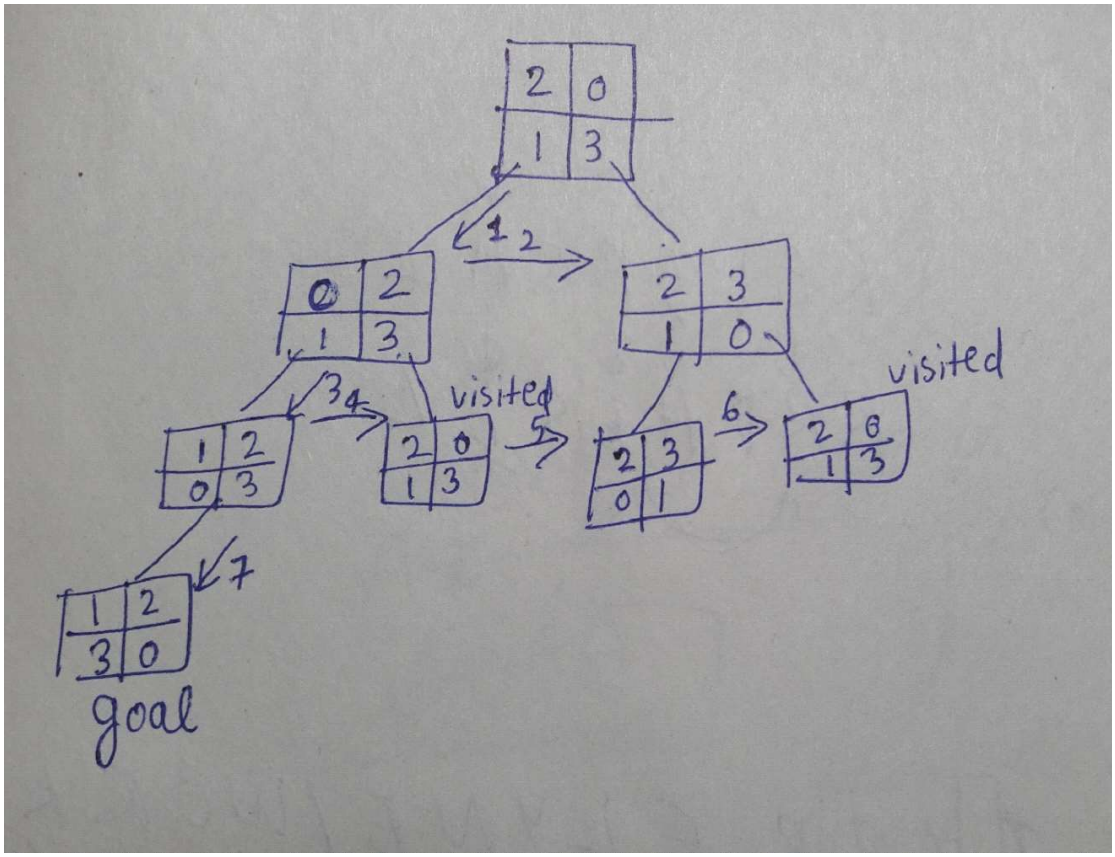
BFS is not give optimal result .But surely give result if present.

It will take enough time for n>3, Since it did level-wise.

BFS will be good when we are very close to final

state ,otherwise it will be worse because it will do all traces of each and every state level-wise.

As per my example of n=3 puzzle problem.



Analysis on Complexity:

So, its time complexity will be $O(b^l)$, where b : branching factor ,and l is number of levels. Here, b is 4 (atmost) and level could be large enough to compute . So, its time complexity will be simply as $O(4^l)$, which is exponential.

So, sometimes it will go in infinite kind of running to find the solution.

And it has space complexity as $O(4^l)$ because may be all nodes are traced to find the result.

Case 2:

DFS Search Algorithm:

DFS is worked as one sided ,it first trace all states in one side of state and so on, until it get final state or goal state.

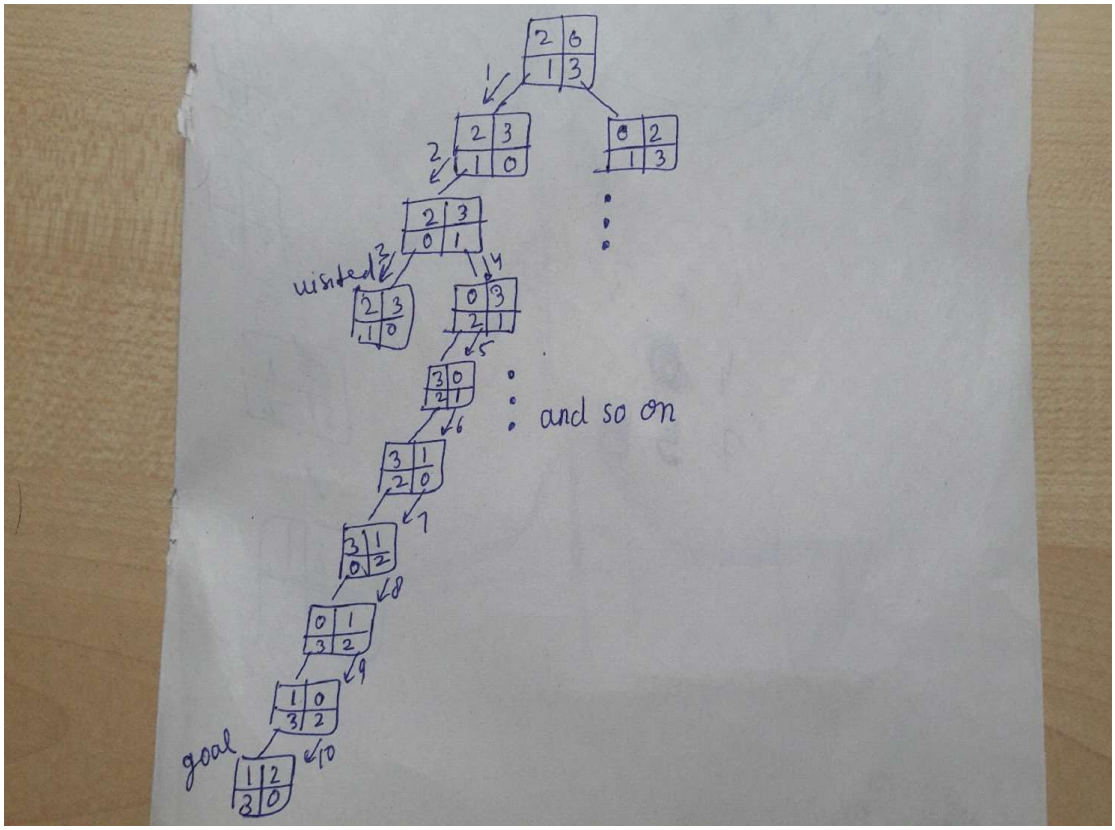
So, For this I used stack in which as states come stacked into stack and the deleted as per required and if it is visited firstly then ,inserted also into visited to take trace of states in searching .

DFS is not give optimal result .But surely give result if present.

It will take enough time for n>3, Since it did one sided-wise.

DFS will be good when we are very close to final state and using same approach is better such that we can go one sided and get the result ,otherwise it will be worse because it will do all traces of each and every node in deep and will fall in infinite depth kind of approach.

As per my example of n=3 puzzle problem.



Analysis on Complexity:

So, its time complexity will be $O(b^l)$, where b : branching factor ,and l is number of levels. Here, b is 4 (atmost) and level could be large enough to compute . So, its time complexity will be simply as $O(4^l)$, which is exponential.

So, sometimes it will go in infinite kind of running to find the solution.

And it has space complexity as O(l) since trace all in deep so max of depth is space complexity.

Case 3:

A* Search Algorithm:

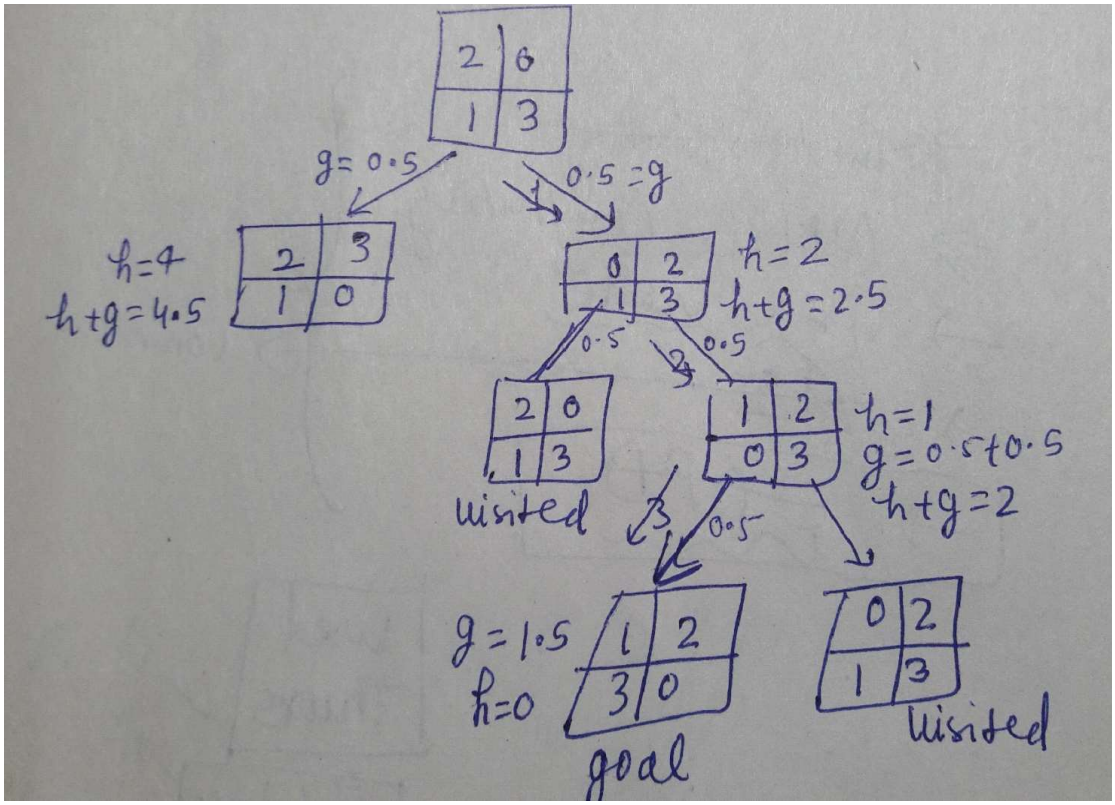
A* is worked as expanding the node and finding the best one to expand again, until it get final state or goal state.

So, For this I do expand state on that whose h+g is min in queued states ,and states are inserted also into visited to take trace of states in searching .

A* will give optimal result .But surely give result if present.

It will take less time compared to BFS and DFS because of intelligence approach of min(h+g) in states present.

As per my example of n=3 puzzle problem.



Analysis on Complexity:

So, its time complexity will be $O(b^l)$, where b : branching factor ,and l is number of levels. Here, b is 4 (atmost) and level could be large enough to compute . So, its time complexity will be simply as $O(4^l)$, which is exponential.

So, sometimes it will go in infinite kind of running to find the solution.

And it has space complexity as $O(b^l)$ since have to store all nodes in worst case.

Case 4:

IDA* Search Algorithm:

IDA* is worked as expanding the node and finding the best one to expand again, until it get final state or goal state as in A* but it will do as iterative increase in cost of h.

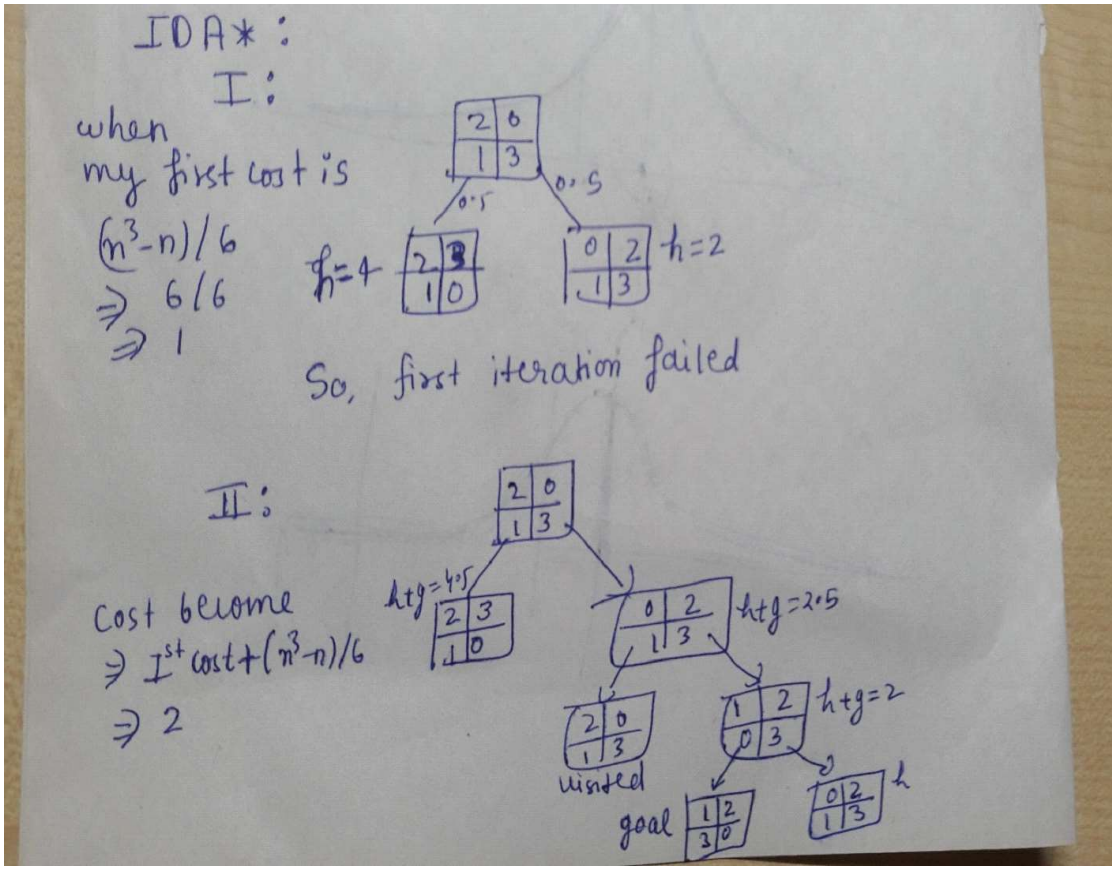
So , that memory load could maintain to some extent.

So, For this I start limit of cost as (n*n*n-n)/6 and then iterating with same limit and other procedure of expand state on that whose h+g is min in queued states ,and states are inserted also into visited to take trace of states in searching is same as in A*.

IDA* will give optimal result .But surely give result if present.

It will take less time compared to BFS and DFS because of intelligence approach of min(h+g) in states present and take less memory than A* (except in worst case).

As per my example of n=3 puzzle problem.



Analysis on Complexity:

So, its time complexity will be $O(b^l)$, where b : branching factor ,and l is number of levels. Here, b is 4 (atmost) and level could be large enough to compute . So, its time complexity will be simply as $O(4^l)$, which is exponential.

So, sometimes it will go in infinite kind of running to find the solution.

And it has space complexity as $O(b^l)$ since have to store all nodes in worst case but better to some extent in normal case.

Findings will be that IDA is much better approach to do this task as it is fast and take lesser memory in normal cases.*

But we can use A if we have no memory problem.*

BFS and DFS as they are blind search algo donot be reliable as take much memory and not as fast and optimality is not guaranteed.

For Part 2 Assignment:

I used two search algorithms as :

- BFS
- DFS

Whenever I get the adjacent same color block, I tried to find the the colour which is not adjacent to it and then ,try to find that colour after selecting random numbers as (h,k) i.e, from (h,k) to (n-1,n-1) and then (0,0) to (h-1,k-1).

If that colour is not in n*n board, then, interchange the below colour from that problem making cell.

In other words, the pattern of state got changed with different patterns of colours but with some frequency of colours .

Do it as we get goal state.

So each state has eight states with these patterns. And so do the same search algorithms for BFS and DFS.

Time complexity and Space Complexity is discussed above.

Findings will be that both BFS and DFS are better in this approach because there is pattern of each state

But sometimes it will be noted that DFS is more better than BFS since it deeply traces states in this approach