10/24

8-puzzle

Algorithms

```
// Initialize goal state, visited set, priority queue, & maps
goal = [[0,1,2],[3,4,5],[6,7,8]]
vis = empty set.
q = empty priority queue
parents_map = empty dictionary
move_map = empty dictionary.

// Calculate Manhattan distance.

function manhattan(curr):
    ans = 0
    pos = {goal[i][j] : (i,j) for i in range(3) for
                                  j in range
    for i in range(3):
        for j in range(3):
            x,y = pos[curr[i][j]]
            ans += abs(i-x) + abs(j-y)
    return ans

def moves(curr):
    (x,y) = find the position of 0 in curr
    poss = [[0,-1,'left'],[-1,0,'up'],[1,0,'down'],
                                  [0,1,'right']]

for each (dx,dy,direction) in poss:
    nx = x+dx
    ny = y+dy

if nx and ny are valid coordinates:
    curr1 = copy of curr.
```

swap curr1[x][y] with curr1[nx][ny]
tuple_curr1 = convert curr1 to tuple form

if tuple_curr1 not in vis:
push(manhattan(curr1), curr1) to q
mark tuple_curr1 as visited

parent_map[tuple_curr1] = curr
move_map[tuple_curr1] = direction

function display(board)
print current board.

Start with the initial puzzle 'c'
Run dfs(c) to solve the puzzle.

Track solution path and moves from the goal back to start.

Output:

Step 0:

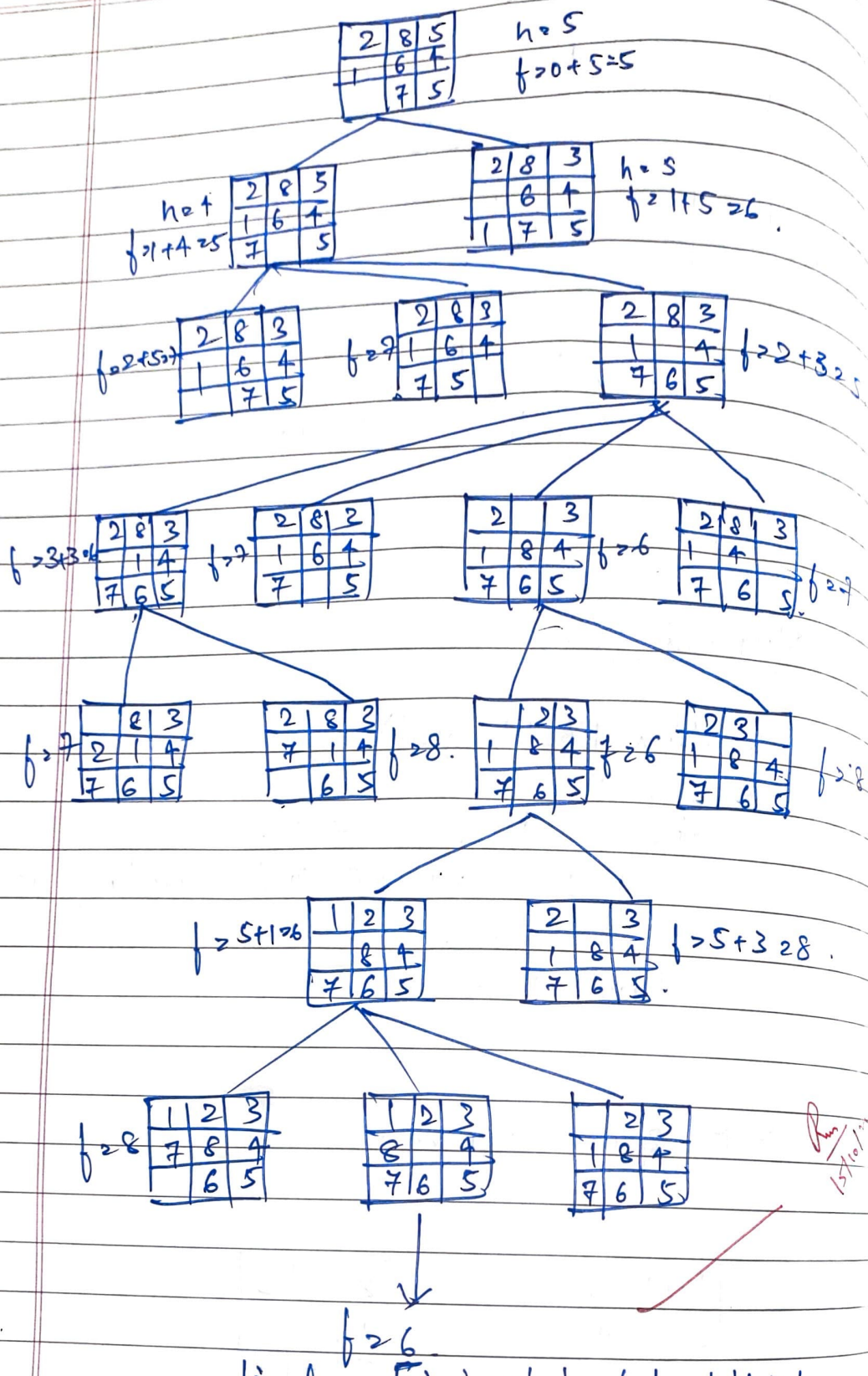
| | | |
|---|---|---|
| 3 | | 2 |
| 5 | 6 | 7 |
| 8 | 4 | 1 |

Initial step.

----->

Step 45:

| | | |
|---|---|---|
| | 1 | 2 |
| 3 | 4 | 5 |
| 6 | 7 | 8 |


h=5
f=0+5=5
h=4
f=1+4=5
h=5
f=1+5=6.
f=2+5=7
f=7
f=2+3=5
f=3+3=6
f=7
f=6
f=7
f=7
f=8.
f=6
f=8
f=5+1=6
f=5+3=8.
f=8
f=6
final state = ['right', 'up', 'up', 'left', 'down', 'right'].