

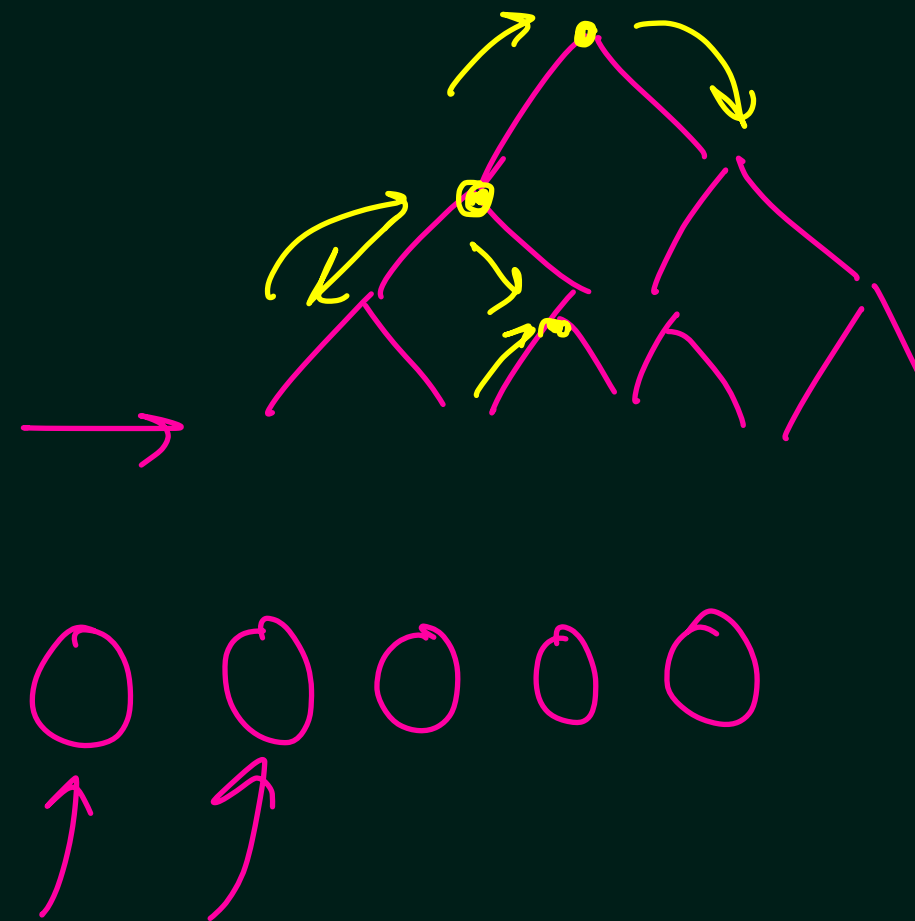
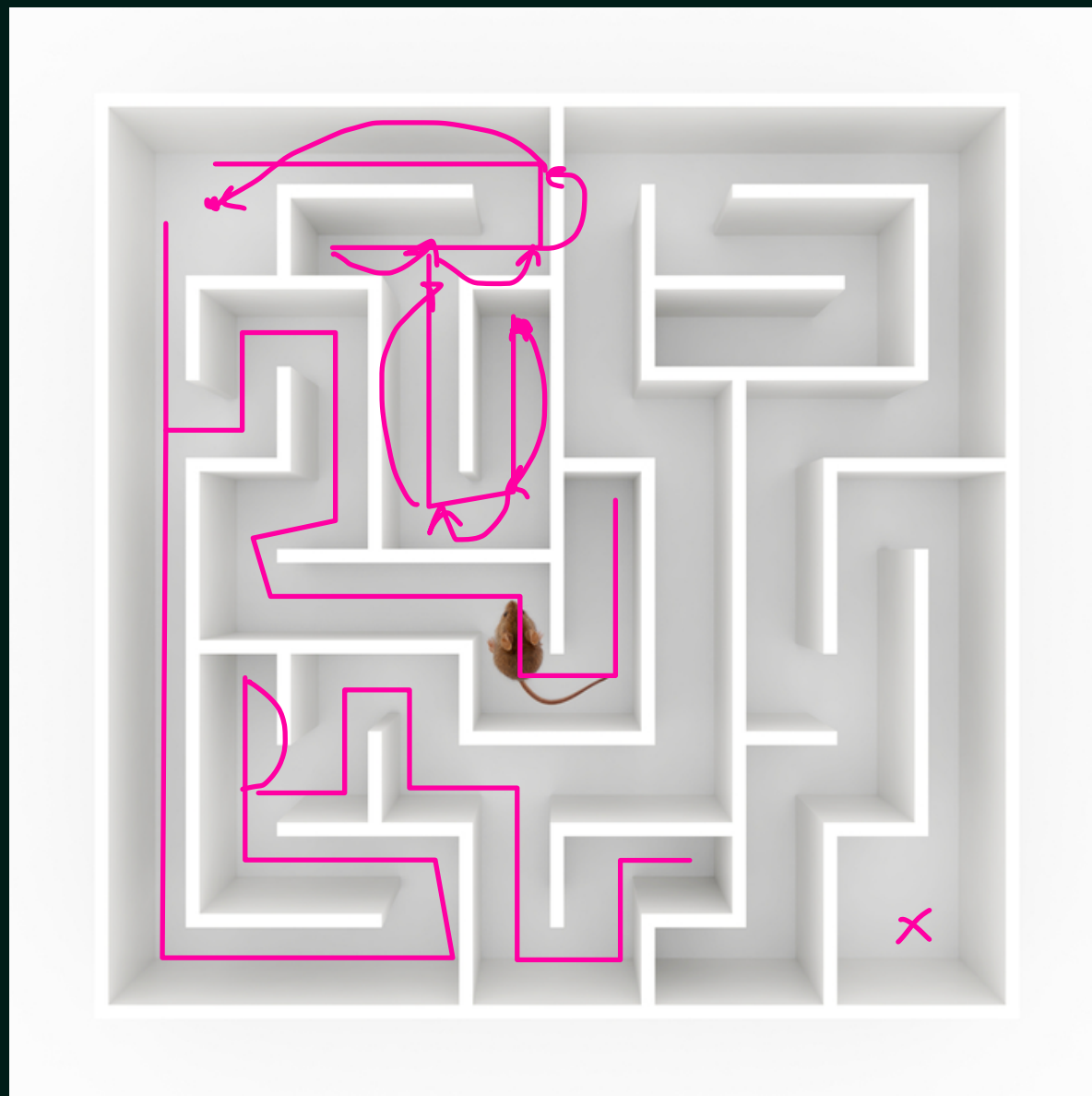
Backtracking - 1

In This Lecture

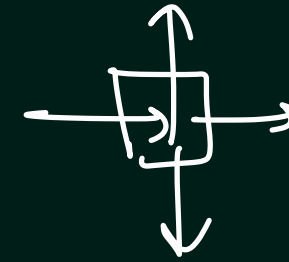
1. What is Backtracking?
2. Rat in a Maze Problem ✓

What is Backtracking?

Backtracking is an algorithmic technique for solving problems recursively by trying to build a solution incrementally, one piece at a time, removing those solutions that fail to satisfy the constraints of the problem at any point in time.



Common structure of Backtracking Solutions

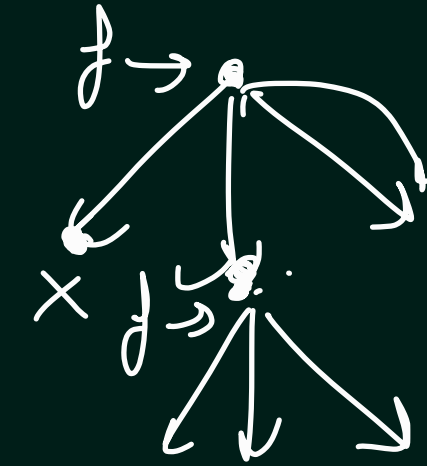


```

boolean findSolutions(n, other params) :
↳ if (found a solution) :
    displaySolution();
    return true;

for (val = first to last) :
    if (isValid(val, n)) :
        ↳ applyValue(val, n);
        ↳ if (findSolutions(n+1, other params))
            return true;

// ↳ removeValue(val, n);
return false;
    
```



```

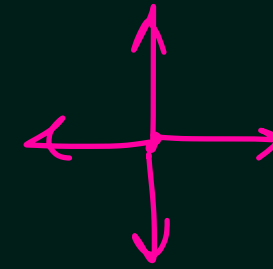
f() {
    →
    add()
    f()
    remove() // backtracking.
}
    
```

Rat in a Maze Problem

Possible Directions: DLRU

$m[][] = \{ \{1, 1, 1, 0\}, \{1, 0, 0, 1\}, \{1, 1, 0, 0\}, \{1, 1, 1, 1\} \}$

(i, j)
3 3



Recursion

→ ① Subproblem
② substructure

- ① Base case → Destination
- ② $D \rightarrow L \rightarrow R \rightarrow U$
- ③ Visited matrix

Output:

DDRRR, DDRRR

Rat in a Maze Problem

Possible Directions: DLRU

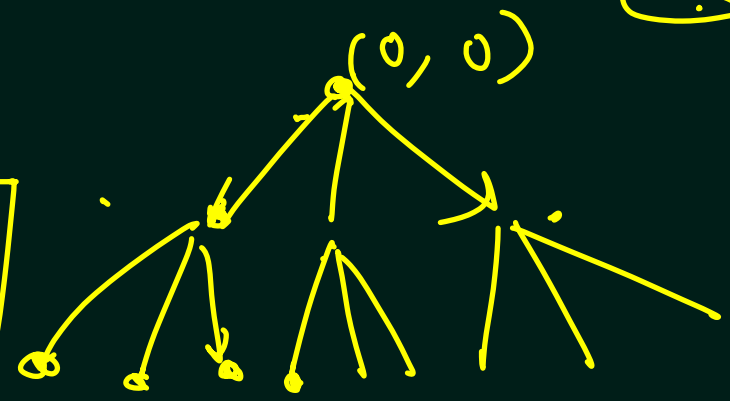
$m[][] = \{ \{1, 1, 1, 0\}, \{1, 0, 0, 1\}, \{1, 1, 0, 0\}, \{1, 1, 1, 1\} \}$
 (Handwritten annotations: 'T' for top, 'D' for down, 'L' for left, 'R' for right, and 'f' for false. Arrows indicate movement from (0,0) to (0,1) and (0,1) to (0,2).)



→ D D D R R R

T.C. $O(3^{n*m})$

S.C. $O(n^2)$



```

→ if(isValid(i+1, j, mat, vis, n, m)) { //D
    vis[i+1][j] = true;
    → ratInAMaze(mat, vis, i+1, j, path: path+'D', n, m);
    vis[i+1][j] = false;
}
↪ if(isValid(i, j-1, mat, vis, n, m)) { //L
    vis[i][j-1] = true;
    ratInAMaze(mat, vis, i, j-1, path: path+'L', n, m);
    vis[i][j-1] = false;
}
- if(isValid(i, j+1, mat, vis, n, m)) { //R
    T vis[i][j+1] = true;
    ← ratInAMaze(mat, vis, i, j+1, path: path+'R', n, m);
    → vis[i][j+1] = false;
}
→ if(isValid(i-1, j, mat, vis, n, m)) { //U
    vis[i-1][j] = true;
    ratInAMaze(mat, vis, i-1, j, path: path+'U', n, m);
    vis[i-1][j] = false;
}
    
```

