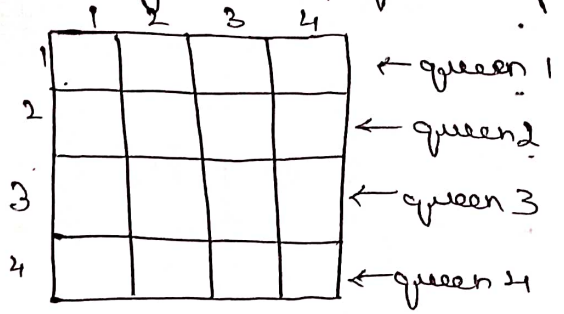
Assignment - 03

DAA (18CS42)

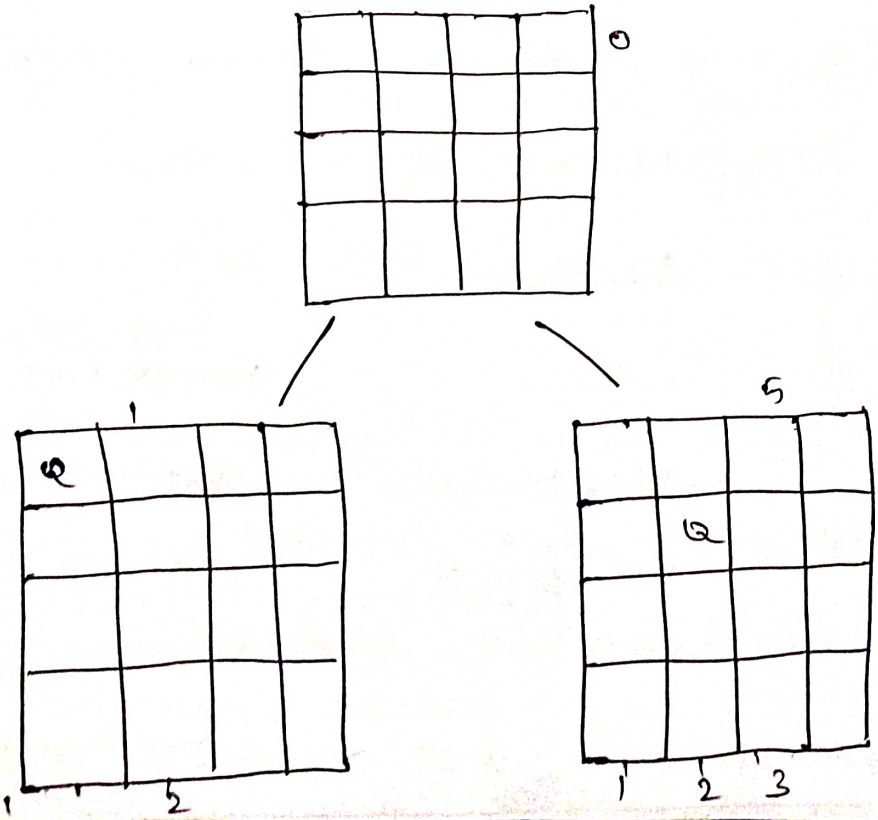
Shahul Hameed S
1KN18CS097
CSE 'A' Sec
4th Sem

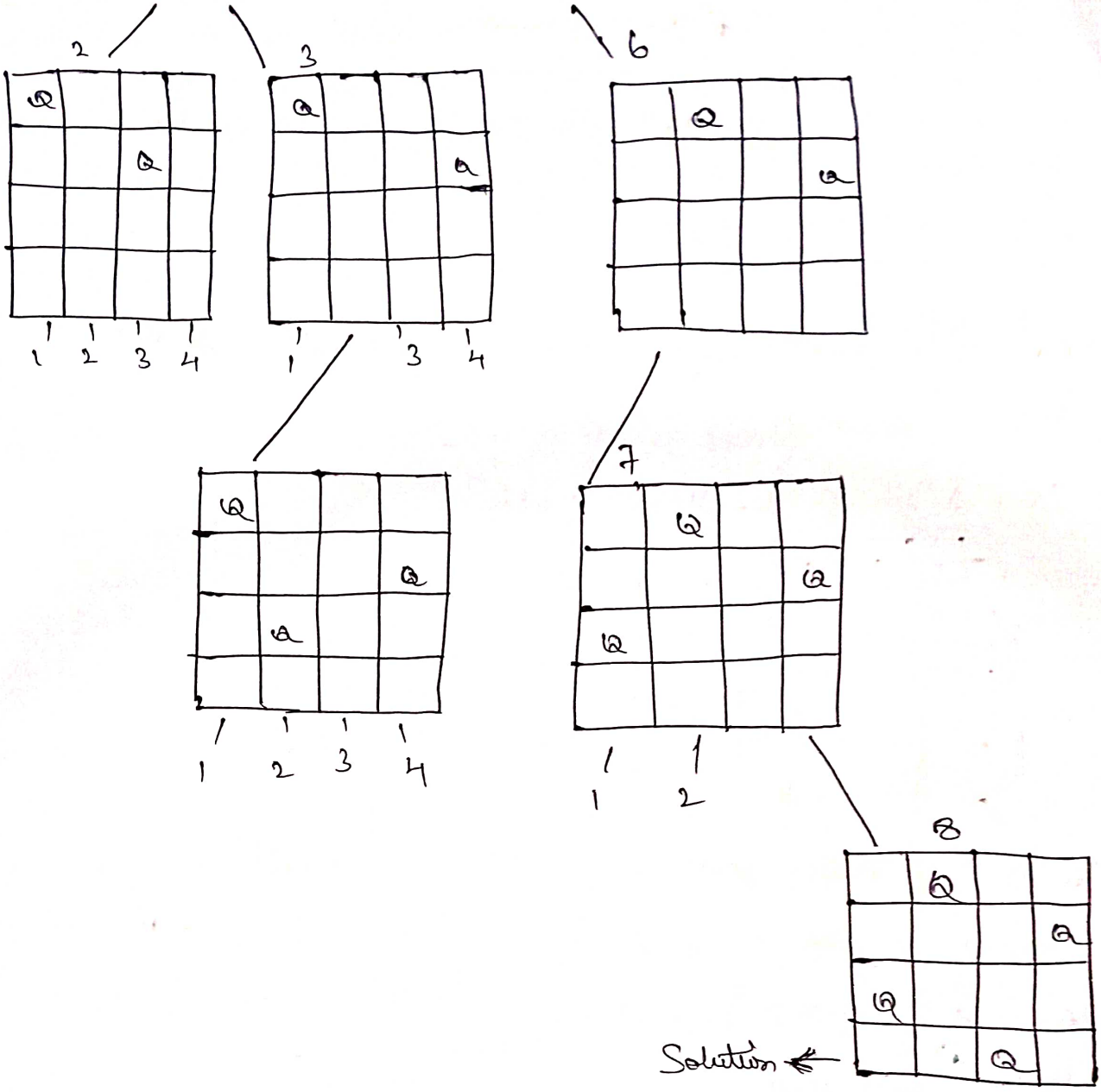
① Draw the state space tree to generate solutions to 4 queens problem.

Board for the 4-queens problem,



In the following state space tree, x-denotes an unsuccessful attempt in place a queen in the indicated column, the numbers above the nodes indicates the order in which the nodes are generated.





(7) Apply backtracking to solve subset sum problem for the instance $n=6$, $d=30$ $S=\{5, 10, 12, 13, 15, 18\}$

> Initially Subset $\{\}$ Sum $=0$

5	5	then add next element
5, 10	15 ∵ $15<30$	add next element
5, 10, 12	27 ∵ $27<30$	add next element
5, 10, 12, 13	40	sum exceeds $d=30$
5, 10, 12, 15	42	sum exceeds $d=30$ ∴ Backtrack

5, 10, 12, 15	45	Sum exceeded not feasible hence backtrack.
5, 10		
5, 10, 13	28	
5, 10, 13, 15	33	not feasible ∴ backtrack.
5, 10		Solution obtained as Sum = 30 = d
5, 10, 15	30	

(3) Explain with an example about travelling sales person Problem using Dynamic programming.

7 Step 1 :- Let the function $c(i, v-(i))$ is the total length of the tree terminating at 1. the objective of TSP problem is that the cost of this tree should be minimum.

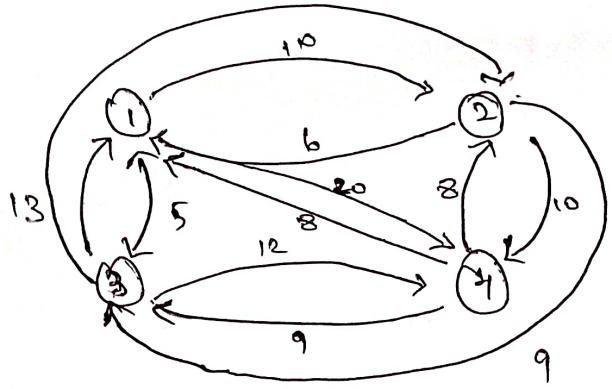
Let $d[i, j]$ be the shortest path b/w 2 vertices and i and j.

Step 2 :- Let $v_1, v_2 \ldots v_n$ be the sequence of vertices followed in optimal tree. Then $(v_1, v_2 \ldots v_n)$ must be a shortest path from v_1 to v_n which passes through each vertex exactly once. Here the principle of optimality is used. The path $v_i, v_{i+1} \ldots v_j$ must be optimal for all paths begining at $v(i)$, ending at $v(j)$ and passing through all the intermediate vertices $\{(v_{i+1}) \ldots v(i-1)\}$ once.

Step 3 :- following formula can be used to obtain the optimum

Cost [i,s] :

$cost(i,S) = \min \{ d[i,j] + cost[j, S-\{j\}] \}$ where $j \in S$ and $i \notin S$. Consider one eg to understand Solving of TSP using dynamic programming approach.



⇒ The distance matrix can be given by, first we will select any arbitary vertex say select 1.

to → / from ↓	1	2	3	4
1	0	10	15	20
2	5	0	9	10
3	6	13	0	12
4	8	8	9	0

Now, process for intermediate sets with increasing size

Step 1:- Let $S = \phi$ then,

$Cost(2, \phi, 1) = d(2,1) = 5$

$Cost(3, \phi, 1) = d(3,1) = 6$

$Cost(4, \phi, 1) = d(4,1) = 8$

That means we have obtained list(2,1) list(3,1) and list(4,1).

Step 2:- Candidate (S) = 1 Applying formula

$Cost(i,S) = \min \{ d[i,j] + cost(j, S-\{j\}) \}$

Hence from router 2 to 1, router 3 to 1 and router 4 to 1 by considering intermediate ~~with~~ path lengths we will calculate total optimum cost.

$$cost(2,\{3\},1) = d(2,3) + cost(3,\phi,1)$$

$$= 9+6 = 15 //.$$

$$cost(2,\{4\},1) = d(2,4) + cost(4,\phi,1)$$

$$= 10+8 = 18$$

$$cost(2,\{2,4\},1) = d(3,2) + cost(2,\phi,1)$$

$$= 13+5 = 18 //.$$

$$cost(3,\{4\},1) = d(3,4) + cost(4,\phi,1)$$

$$= 12+8 = 20$$

$$cost(4,\{2\},1) = d(4,2) + cost(2,\phi,1)$$

$$= 8+5 = 13$$

$$cost(4,\{3\},1) = d(4,3) + cost(3,\phi,1) = 9+6 = 15$$

Step 3: Consider Candidate (S) = 2

$$cost(2,\{3,4\},1) = \min\{[d(2,3) + cost(3,\{4\},1)], \{[d(2,4) + cost(4,\{3\},1)]\}$$

$$= \min\{[9+20],[10+15]\} = 25 //.$$

$$cost(3,\{2,4\},1) = \min\{[d(3,2) + cost(2,\{4\},1)], [d(3,4) + cost(4,\{2\},1)]\}$$

$$= \min\{[13+18],[12+13]\}$$

$$= 25 //.$$

$cost(4,\{2,3\},1) = \min\{[d(4,2) + cost(2,\{3\},1)],$
$[d(\ ,\ ) + cost(\ ,\{\ \},\ )]\}$

Step 4!: Consider Candidate (s) = 3 . i.e cost $(1,\{2,3,4\})$ but as we have choosen vertex 1, initially the cycle should be completed i.e starting and ending vertex should be 1

∴ we will compute,

$cost(1,\{2,3,4\},1) = \min\{[d(1,2) + cost(2,\{3,4\},1)],$
$[d(1,3) + cost(3,\{2,4\},1)],$
$[d(1,4) + cost(4,\{2,3\},1)]$
$\}$

$= \min\{[10+25], [15+25], [10+23]\}$

$= 35$ //.

∴ The Optimal tour is of path length 35.

Now, considering step4, now from vertex 1 we obtain the optimum path as $d(1,2)$. Hence select vertex 2, Now consider step3, in which vertex 2 we can obtain optimum cost from $d(2,4)$. Hence select vertex 4. Now in Step 2 we get remaining vertex 3 as $d(4,3)$ is optimum. Hence optimal tour is 1,2,4,3,1.

---

(4) what is Branch and bound Algorithm? How it is different from backtracking?

4 Branch and bound is an algorithm design paradigm which is generally used for solving Combinatorial optimization problems. These problems are typically exponential in terms of time complexity and may require exploring all possible permutations in worst case. The Branch and bound Algorithm solves these problems relatively quickly.

Backtracking	Branch and bound algorithm
* Backtracking Traverse the state space tree by DFS manner.	* Branch and bound traverse the tree in any manner, DFS/BFS.
* Backtracking involves feasibility function.	* Branch and bound involves a bounding function.
* Backtracking is used for solving decision problem.	* Branch and bound is used for solving optimization problem.
* Backtracking is more efficient	* Branch and bound is less efficient.

5 what is Hamiltonian cycle ? give the Backtracking based algorithm to find the Hamiltonian cycle in the graph. write the functions used to generating next vertex and for finding Hamiltonian cycle.

7 Hamiltonian cycle : A path through a graph that starts and ends at the same vertex and includes every other vertex exactly once also known as ~~tour~~ tour.

```
algorithm :- {
    repeat
    {
     Next vertex (k);
      If [x[k]=0] then
       return ;
      if (k =n) then
      write ( x[1:n])
      Else
         If cycle (k+1);
       } until (false);
      }
Algorithm  Next value (k)
   {
    repeat
    {
    x[k]:= x(x[k]+1) mod(n+1);
    if (x[k]=0 ) then return ;
    for   j:i = 1  to n do
    {
    if [ (eq (k,j) =0 ] and [ x[k] = x[j] ]
     then break ;
    }
    if (j=n+1) then return ;
    }until (false);
   }
```

(6) Write a short note on (i) Hamiltonian problem.

(ii) M-colouring problem.

> (i) Hamiltonian problem: Hamiltonian path is an undirected graph is a path that visits each vertex exactly once. A Hamiltonian cycle (circuit) is a hamiltonian path such that there is an edge from the last vertex to the first vertex of the hamiltonian path. Determine whether a given graph contains hamiltonian cycle or not. If it contains then, print the path.

(ii) M-colouring problem: given an undirected graph and a number m, determine if the graph can be coloured with at most m colours such that no two adjacent vertices of the graph are coloured with the same problem. Here colouring of a graph means the assignments of colour to all vertices.