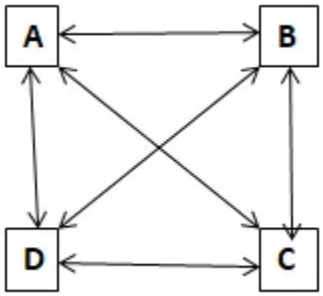
Implementation:

Travelling Salesman Problem



Cost matrix:

	A	B	C	D
A	0	10	15	20
B	5	0	9	10
C	6	13	0	12
D	8	8	9	0

Problem Formulation:

(1) Initial state:

The starting city selected by the salesman.

Let us assume that the starting city here is A.

(2) Actions or successor function or operators:

- Choose the next node such that the cost is minimum.
- Traverse through all the nodes and come back to the starting node.
- The path taken by the salesman should have the minimum cost.

(3) Goal test

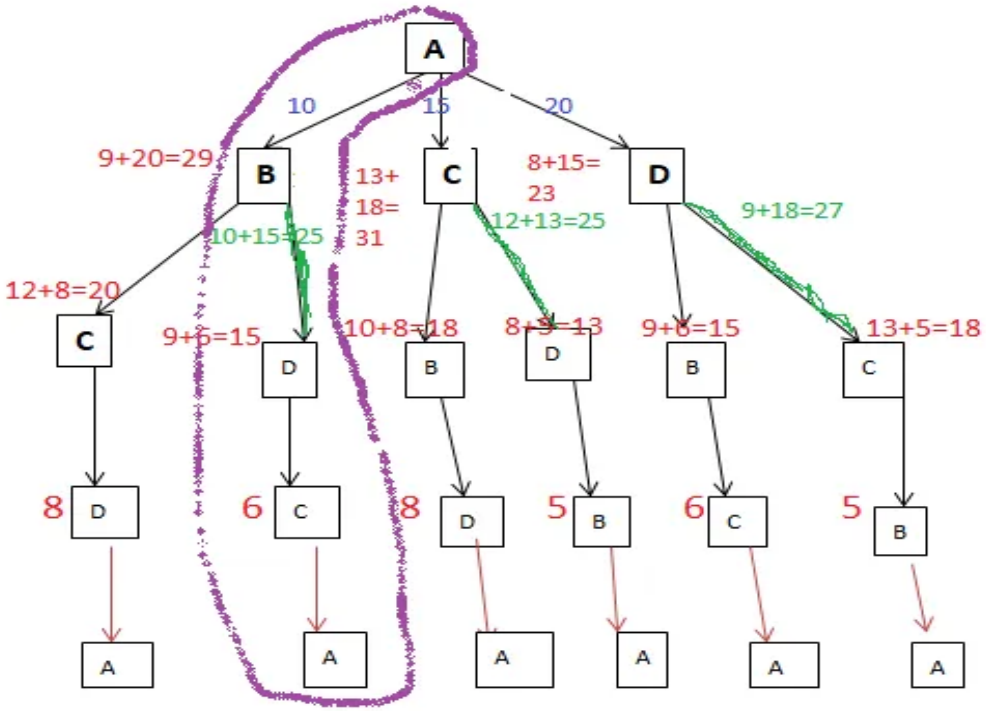
- **The salesman has to select a starting point (starting city), then has to visit all the cities and has to return to the starting point (where he started).**
- **Minimise the total path (length of path) or travel cost travelled by the salesman.**

(4) Path cost function

$g(i,S) = \min [w(i, j) + g(j,S-\{j\})]$

The process of this formula is the same as we have solved the tree using brute force approach. But when we have a large search space, we cannot apply brute force search because time complexity will be very much. Therefore, a dynamic approach is used which is the best solution for this problem.

(5) Solution (Please include state space search graph)



Code and Output:

```
from sys import maxsize
from itertools import permutations
V = 4

def tsp(graph, s):
    vertex = []
    for i in range(V):
        if i != s:
            vertex.append(i)

    min_path = maxsize
    next_permutation = permutations(vertex)
    for i in next_permutation:
        current_pathweight = 0
        k = s
        for j in i:
            current_pathweight += graph[k][j]
            k = j
        current_pathweight += graph[k][s]
        min_path = min(min_path, current_pathweight)

    return min_path

if __name__ == "__main__":
    graph = [[0, 10, 15, 20], [5, 0, 9, 10],
             [6, 13, 0, 12], [8, 8, 9, 0]]
    s = 0
    print("The minimum cost is:")
    print(tsp(graph, s))
```

```
The minimum cost is:
35
```

Result and Discussion: The travelling salesman problems abide by a salesman and a set of cities. The salesman has to visit every one of the cities starting from a certain one (e.g., the hometown) and to return to the same city. The challenge of the problem is that the travelling salesman needs to minimise the total length of the trip. In this experiment, I formulated the Travelling Salesman Problem. The implemented python program for Travelling Salesman Problem takes the cost matrix as the input and gives minimum path cost as the output.

Learning Outcomes: Students should have the ability to

LO1: Formulate the problem using AI Approach.

LO2 : Solve the problem using AI Approach.

Course Outcomes: Upon completion of the course students will be able to evaluate various problem solving methods for an agent to find a sequence of actions to reach the goal state.

Conclusion: Every problem should be properly formulated in Artificial Intelligence. Problem formulation is very important before applying any search algorithm. By performing this experiment, we were able to understand the steps in formulation of any problem. I formulated the Travelling Salesman Problem and implemented it using python to get the desired output.

Viva Questions:

Q.1 Define a problem solving agent with an example.

Q.2 Illustrate 3 problems can be solved using the problem formulation approach.

Q.3 Discuss various items or elements to be considered during problem formulation.

For Faculty Use

Correction Parameters	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				