

Artificial Intelligence Lab - 2

Aim : To develop agent programs for Real World Problems, i.e., Travelling Salesman Problem (TSP)

Algorithm :

1. Consider city 1 as the starting and ending point.
2. Generate all $(n-1)!$ Permutations of cities.
3. Calculate cost of every permutation and keep track of minimum cost permutation.
4. Return the permutation with minimum cost.

Code :

```
#TSP

from sys import maxsize
from itertools import permutations
V = 4

def travellingSalesmanProblem(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], [10, 0, 35, 25],
             [15, 35, 0, 30], [20, 25, 30, 0]]
    s = 0
    print("Weight for the min path is",travellingSalesmanProblem(graph,
s))

```

Output :

The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a directory structure with files like 'Lab1_ToyProblem.py', 'Lab2_TSP.py', and 'Lab3.py'. The code editor displays the Python script from the previous block. The terminal shows the output of the script: 'Weight for the min path is 80'. The terminal also shows the command 'RA1911003010646/Lab2_TSP.py' and the runner 'Python 3'.

```

33
34
35
36
37
38
39
40
41
42
43
44
33
34
35
36
37
38
39
40
41
42
43
44
graph = [[0, 10, 15, 20], [10, 0, 35, 25],
         [15, 35, 0, 30], [20, 25, 30, 0]]
s = 0
print("Weight for the min path is",travellingSalesmanProblem(graph, s))

Weight for the min path is 80

Process exited with code: 0

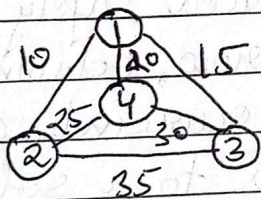
```

AI Lab-2

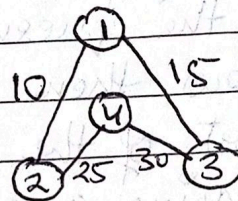
Aim - To develop agent programme for real world problem, i.e., Travelling Salesman Problem

Problem Formulation

For a given complete graph with n vertices & weight function defined on the edges, the objective is to construct a tour, i.e., a circuit that passes through each vertex only once and returns back to the starting point with minimum total weight.



Initial State

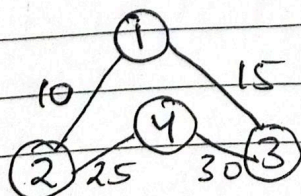


Final State

Problem Solving

We start at vertex 1 and find the minimum cost path with 1 as starting point, i as ending point & all vertices appearing exactly once.

Now first we find a path for the same condition and try various permutations to find the min path out of various path. For this example it is 1-2-4-3!



Min. weight = 80