Travelling Salesman Problem

1 Definitions

The travelling salesman problem looks for a **walk** that gives the minimum **tour**.

Walk - A finite series of edges so that the end of one vertex is the start of the next

Tour - A walk that visits every vertex and returns to the starting vertex
Add more information about upper and lower bounds here when you understand it more

2 The differences between classical and practical problems

Classical Problem - Must visit each vertex **only once** before returning to the start.
Practical Problem - Must visit each vertex **at least once** before returning to the start

3 Converting a network into a complete network of least distances

If a network is converted into a complete network of least distances, the classical and practical problem are the same.

To create a complete network of least distances, you must ensure the **triangle inequality** holds for all triangles in the network.

Triangle inequality:
The longest side of any triangle $\leqslant$ The sum of the two shorter sides

In a network where the triangle inequality does not hold, replace the longest arc with the sum of the two shorter ones.

4 Using a Minimum Spanning tree to find the upper bound of the travelling salesman problem

Method:

- Find the minimum spanning tree(Prim's or Kruskal's). This guarantees all vertexes are included.
- Double the length of the minimum spanning tree as the route includes going there and back.
- Find "short cuts"(using the non included arcs to bypass repeated edges.

This algorithm gives the initial upper bound

5 Using a minimum spanning tree to find a lower bound

Method:

1. Remove a vertex from the matrix
2. Find the minimum spanning tree for the remaining vertices (Residual Minimum Spanning Tree)
3. Find the two shortest connectors to the removed vertex and add them to the RMST
4. The greatest total from removing a vertex is the lower bound

Notes

- The lower bound should be as large as possible to make the interval between upper and lower bounds smallest
- You have an optimal solution if
  - The lower bound gives a tour, or
  - The lower bound has the same value as the upper bound

6 Nearest Neighbour algorithm

1. Select the vertex chosen to start at
2. Go to the nearest unused vertex
3. Repeat step 2 until all vertices are included
4. Return to the start vertex directly from the end vertex
5. Repeat for other vertices and select the tour with the smallest length as the upper bound

7 Flowchart of Nearest Neighbour