Travelling Salesman Problem (15/1/2014)

Finding the shortest sequence of arcs that visits each node at least once, returning to the starting point.

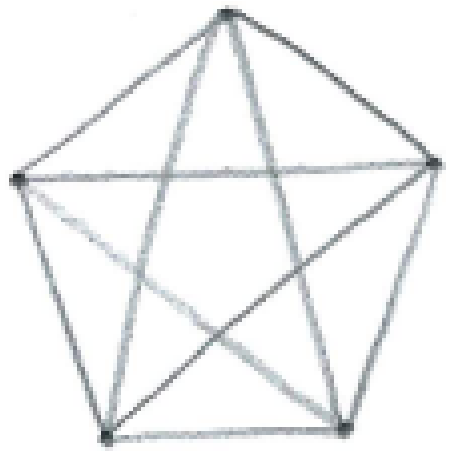

Figure 1

For a complete graph, such as K_5 in Figure 1, it is always possible to find a cycle (ie a closed path that doesn't repeat arcs or nodes) that visits each node **exactly once**. Such a cycle is referred to as a Hamiltonian cycle, and the travelling salesman problem is often expressed in the stronger form of requiring the sequence of arcs to be a Hamiltonian cycle. A Hamiltonian cycle is also called a 'tour'.

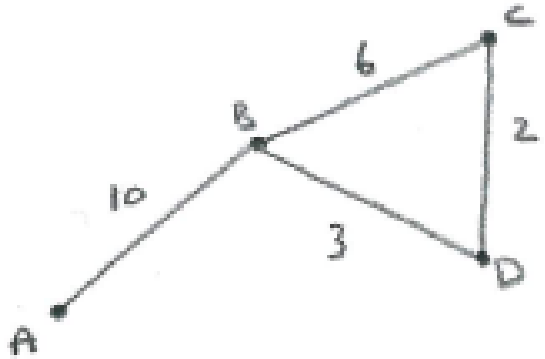

Figure 2

Figure 2 shows a graph (or 'network', as it has weights) which contains no Hamiltonian cycles. In this case, the travelling salesman problem can only be solved by permitting both arcs and nodes to be repeated.

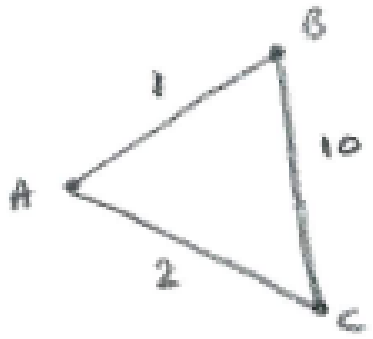

Figure 3

For figure 3, a Hamiltonian cycle exists, but it clearly doesn't give the best solution. It would be better to voluntarily repeat arcs and nodes.

If we are insisting on only following Hamiltonian cycles then the problem is termed the 'Classical' problem; otherwise it is referred to as the 'Practical' problem. In the latter case, the only requirement is for each node to be visited at least once (as well as returning to the starting point).

Some of the methods described below for finding the best solution may only be applicable to one or other of these types of problem. However, it will be seen that the Practical problem can always be expressed in the form of a Classical problem, thereby making extra methods available.

If a graph is complete and has n nodes, then there will be $\frac{1}{2}(n-1)!$ possible tours. (We can choose to start at any node

(since the tour is circular), and there will be $n-1$ ways of choosing the next node to proceed to (and so on). We divide by 2 because reversing the order gives the same tour; eg ABCDEA is the same as AEDCBA.

The travelling salesman problem thus has factorial complexity, and computers cannot simply examine all the possibilities if n is even of moderate size:

$n = 5$: 12 tours

$n = 10$: 181 440 tours

$n = 15$: 4.4×10^{10} tours

Unfortunately no algorithm exists for the travelling salesman problem which can determine the optimal solution. The best that can be done is to establish upper and lower bounds for the shortest length, and then to try and refine these, so that the optimal value is shown to lie within a small enough interval; ie such that no significant improvement will be obtained from further work.

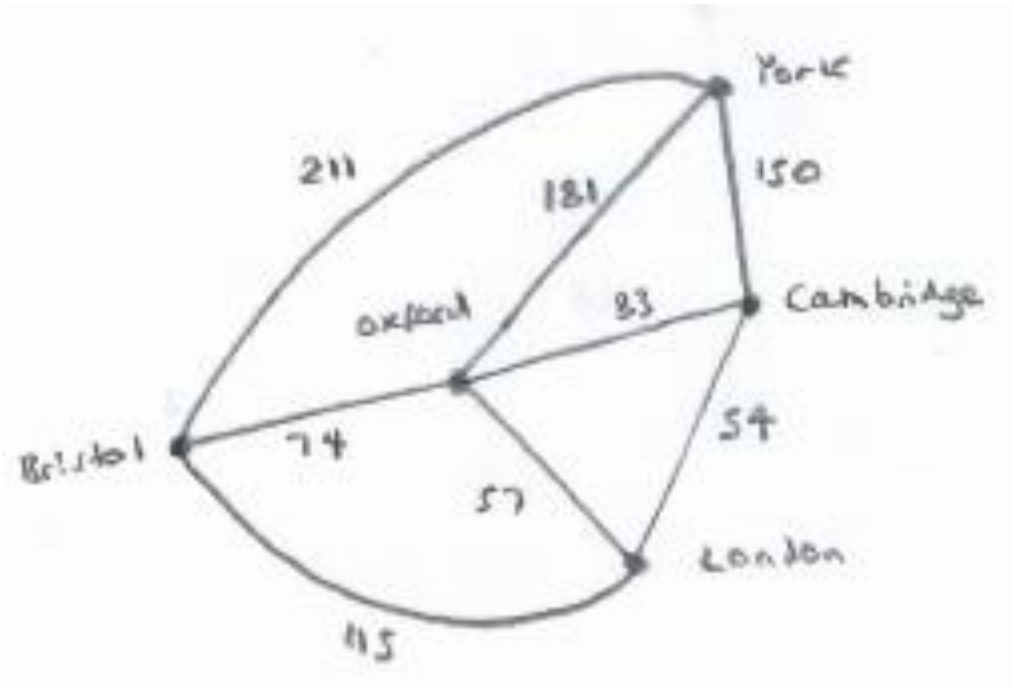

Figure 4

Referring to the cities example in Figure 4, three possibilities (with their total lengths) are:

BYOCLB (211+181+83+54+115=644)

BYCOLB (211+150+83+57+115=616)

BOYCLB (74+181+150+54+115=574)

The following systematic methods can be applied to tackle the problem.

Upper Bound

For the Practical problem, we can start by establishing an upper bound for the shortest length.

The steps are as follows:

(1) Find a minimum connector for the whole network.

(2) Duplicate each of the arcs.

This is carried out below for the cities network in Figure 4.

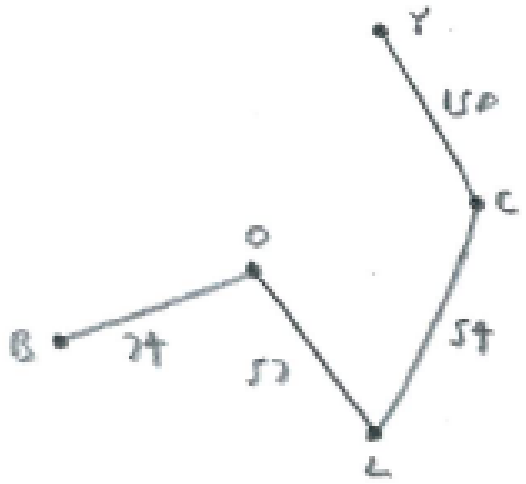

Figure 5

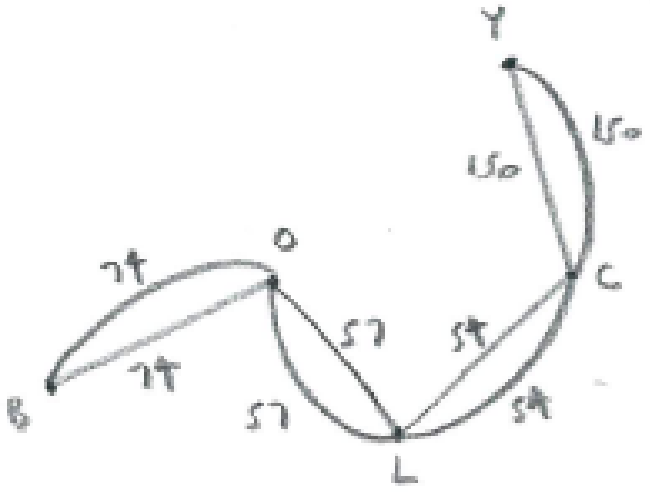

Figure 6

Upper bound = (74+57+54+150)x2=670

Lower Bound

[This method is only guaranteed to work if the network is complete.]

Referring to Figure 4, any Hamiltonian cycle will consist of 2 arcs from (eg) B, together with 3 arcs linking O, L, C & Y.

The minimum total length of 2 arcs from B is 74+115=189

To find the minimum total length of arcs linking O, L, C & Y, we can find the minimum connector for these nodes. This is done below by three methods.

Prim's Algorithm

eg starting at O:

OL(57)

OL(57)+LC(54)

OL(57)+LC(54)+CY(150) = 261

Kruskal's Algorithm

LC(54)

LC(54)+LO(57)

LC(54)+LO(57)+CY(150) = 261

Removing the longest arcs:

(181+150+83+57+54)-181-83 = 261

Hence the lower bound is 189+261=450

Further lower bounds can be established by dividing up the nodes differently. Using Kruskal's algorithm gives:

O + BLCY

(57+74)+(54+115+150)=450

L + BOCY

(54+57)+(74+83+150)=418

C + BOLY

(54+83)+(57+74+181)=449

Y + BOLC

(150+181)+(54+57+74)=**516**

The figure of 516 supersedes the other, lower figures (although it is still true that the shortest tour can't be lower than 418, it is also true that it can't be lower than 516).

Nearest Neighbour Algorithm

This is another way of producing an upper bound, as the algorithm finds a tour that works; ie the shortest possible tour will have a length less than or equal to the value obtained.

The method is only guaranteed to work if the network is complete.

Referring to Figure 4,

(1) Start at any node (eg B)

(2) Add the shortest arc leading to a new node: BO

(3) Repeat the process, to give: BO+OL+LC+CY

(4) Return directly to the start node, to give the cycle:

BOLCYB (74+57+54+150+211=546)

(5) Repeat the algorithm, with other starting points:

OLCYBO (57+54+150+211+74=546)

LCOBY: can't return to L

CLOBY: can't return to C

YCLOBY (150+54+57+74+211=546)

In general we would take the lowest of the figures obtained.

The Nearest Neighbour algorithm resembles Prim's algorithm. However Prim's algorithm joins the nearest new node to **any** existing node, whereas the Nearest Neighbour algorithm joins it to the last node obtained. Also, Prim's algorithm is designed to produce a tree and we don't return to the start node.

Notes on the Nearest Neighbour Algorithm

(i) It is a 'greedy' algorithm: it doesn't look ahead, and just maximises the short-term gain (by selecting the nearest node). This is similar to a chess player who sees that a piece can be taken, without considering the longer-term consequences.

(ii) As seen above, it doesn't always work.

(iii) Although the solution will usually be much better than for the "Upper Bound" method described earlier, the Nearest Neighbour algorithm doesn't usually give the best possible solution.

Improving on the Upper Bound

The 'Upper Bound' method can usually be improved on significantly by taking one or more shortcuts.

Referring to Figure 6, we start with BOLCYCLOB:

$(74 + 57 + 54 + 150) \times 2 = 335 \times 2 = 670$

This can be replaced with

BOLCY**CO**B: $335 + 150 + 83 + 74 = 642$

or with BOLC**YO**B: $335 + 181 + 74 = 590$

or with BOLC**YB**: $335 + 211 = 546$

In this case, we have succeeded in bringing the value down to that obtained by the Nearest Neighbour algorithm.

So far we have 'trapped' the shortest possible tour between 516 and 546. However there is no guarantee that a shorter tour than 546 exists.

Tour improvement algorithm

This can be illustrated by referring to Figure 4 again. The three possibilities that were mentioned earlier were:

BYOCLB (211+181+83+54+115=644)

BYCOLB (211+150+83+57+115=616)

BOYCLB (74+181+150+54+115=574)

Note that BYOCLB is improved by swapping O and C, or by swapping Y and O.

The algorithm consists of examining each sequence of 4 nodes, and seeing if an improvement can be obtained by swapping the middle two nodes.

Thus, for the above example, YCOL gives a shorter distance than YOCL, and BOYC gives a shorter distance than BYOC.

Using computer language, the algorithm could be written as:

For i = 1 to n

If $d(N_i, N_{i+2}) + d(N_{i+1}, N_{i+3}) < d(N_i, N_{i+1}) + d(N_{i+2}, N_{i+3})$ then swap N_{i+1}& N_{i+2}

Next i

(where N_i denotes the ith node, and d denotes distance).

Converting from the practical problem to the classical problem

For the network in Figure 2, no Hamiltonian cycle exists. We may wish to convert this network into a complete network; eg in order to be able to use the Nearest Neighbour algorithm (which is only guaranteed to work if the network is complete).

This can be done by creating the network of shortest distances, as in Figure 7 below.

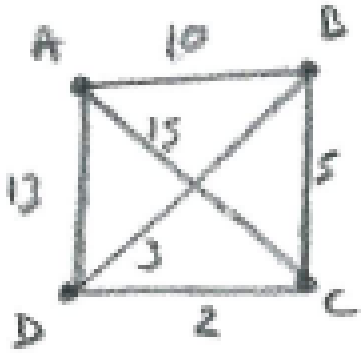

Figure 7

(Thus A is now connected directly to C by an arc of weight equal to the shortest distance between A and C in the original network.)

Note that, although the Practical problem has been converted to the Classical problem, we are not pretending that the solution obtained won't involve repeating nodes in the original network.

We may also wish to 'voluntarily' use the network of shortest distances. For example, in Figure 3 we saw that the Hamiltonian cycle ABC does not give the best solution. Figure 8 below shows the equivalent network of shortest distances, to which the Classical problem can be applied (though clearly in this simple case there is no need to apply any of the methods described in this note).

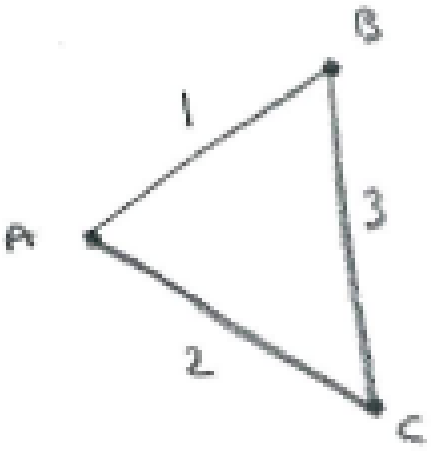

Figure 8