Travelling Salesman problem (30 pages; 22/10/26)

Definitions

Walk: (a sequence of arcs, where the end of one arc is the start of the next) arcs and nodes can be repeated

Tour: ?

Trail: arcs can't be repeated, but nodes can be

Path: arcs and nodes can't be repeated

Cycle: a closed path (arcs and nodes can't be repeated)

Hamiltonian (or sometimes Hamilton) cycle: ?

Route (as in Route Inspection): includes every arc

Eulerian cycle: A cycle that includes every arc exactly once

Walk: (a sequence of arcs, where the end of one arc is the start of the next) arcs and nodes can be repeated

Tour: a closed walk that visits every node at least once (need not include all arcs)

Trail: arcs can't be repeated, but nodes can be

Path: arcs and nodes can't be repeated

Cycle: a closed path (arcs and nodes can't be repeated)

Hamiltonian (or Hamilton) cycle: a closed walk that visits every node exactly once (need not include all arcs)

Route (as in Route Inspection): includes every arc

Eulerian cycle: A cycle that includes every arc exactly once

Travelling Salesman Problem

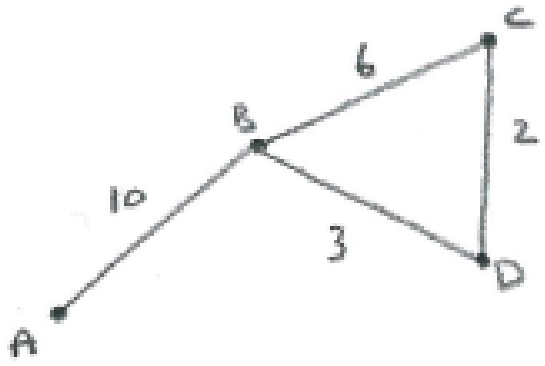
'Classical problem': Each node is visited exactly once

ie finding the shortest Hamiltonian cycle (need not include all arcs)

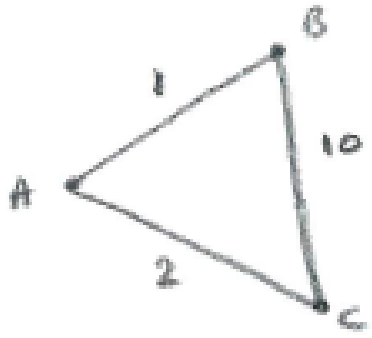
(In a complete network [where all nodes are directly connected to each other], a Hamiltonian cycle is guaranteed to exist.)

'Practical Problem': where repeated nodes are allowed.

Either because a Hamiltonian cycle does not exist:

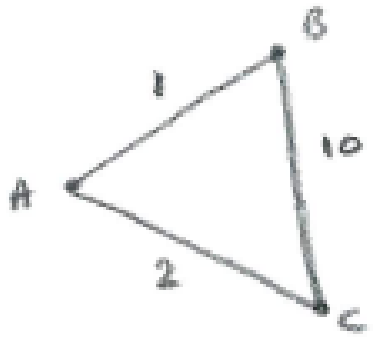


Or, if one does exist, because it doesn't give a good solution:

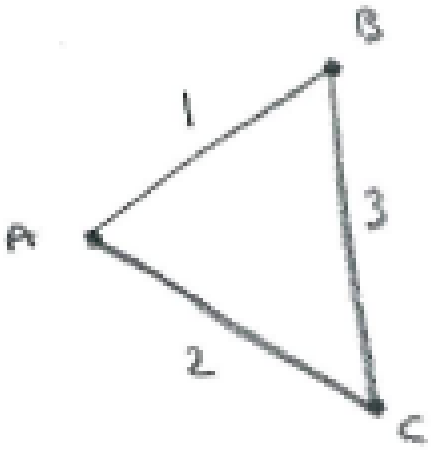


(the tour ABACA will usually be preferable to the Hamiltonian cycle ABCA)

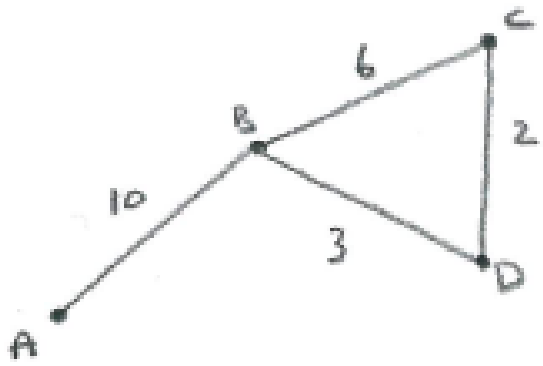
The Practical problem can always be converted to the Classical problem by creating a network of shortest distances.

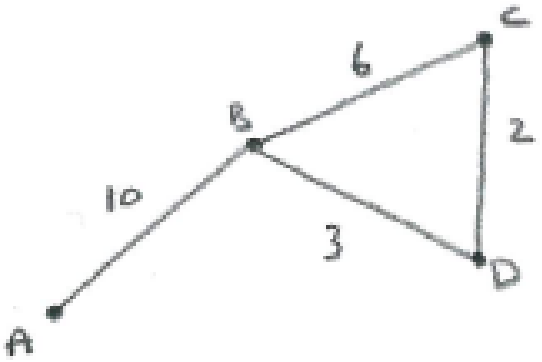


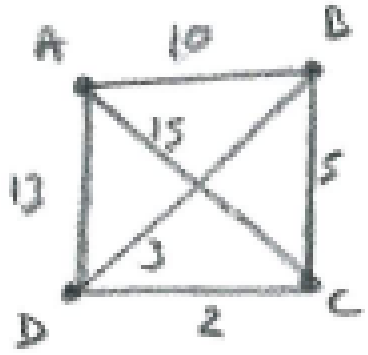
becomes



Create a network of shortest distances for







Note that the solution found will need to be converted back to the original form (eg any reference to AD in the network of shortest distances above will be replaced with ABD in the original network).

‘Triangle Inequality’

A network of least distances can theoretically be obtained by considering every triangle of the form ABC (where there are direct links between A & B, A & C, and B & C). The distance AC can be replaced with $AB + BC$ if this is smaller.

(Usually use inspection; or Dijkstra’s algorithm, or Floyd’s algorithm.)

No feasible algorithm exists for the travelling salesman problem which can determine the optimal solution.

Instead we can:

- establish an upper bound (by finding a distance U that enables all nodes to be visited)

- establish a lower bound (showing that a distance of at least L is required, in order to visit all the nodes)

- refine these (finding smaller upper bounds and/or bigger lower bounds)

- until we are satisfied that the gap between L and U is small enough for our purposes

- U can then be taken as the answer (as this is the smallest known distance that enables all nodes to be visited)

Note: Some of these algorithms rely on the network being complete, or may not produce a Hamiltonian cycle.

However, if we work with the network of shortest distances, this will be complete.

The following algorithms are used to find upper or lower bounds:

Upper Bound methods

Inspection

Doubling the Minimum Connector (with improvements)

Nearest Neighbour algorithm

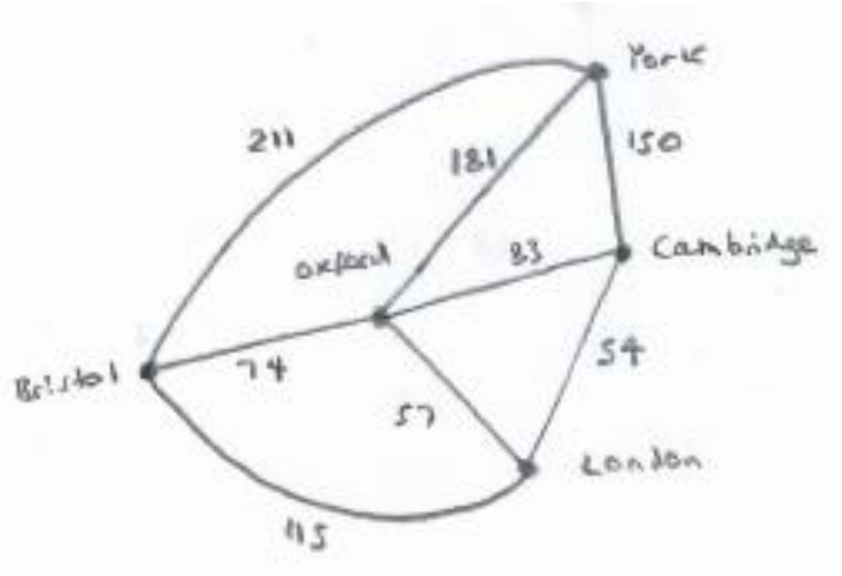
Tour improvement algorithm

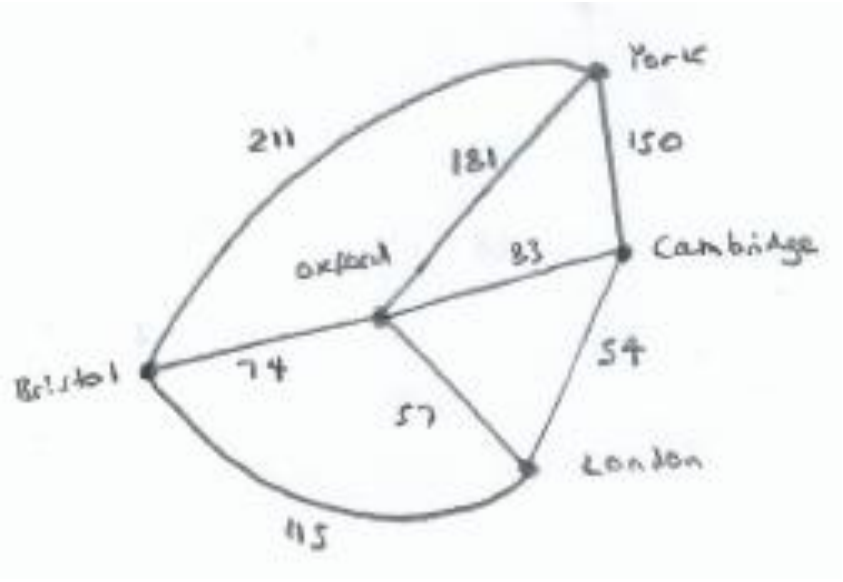
Lower Bound method

Lower Bound algorithm (using Minimum Connector)

Upper Bound: Inspection

Find some tours for the following network (as short as possible; ideally not repeating nodes):





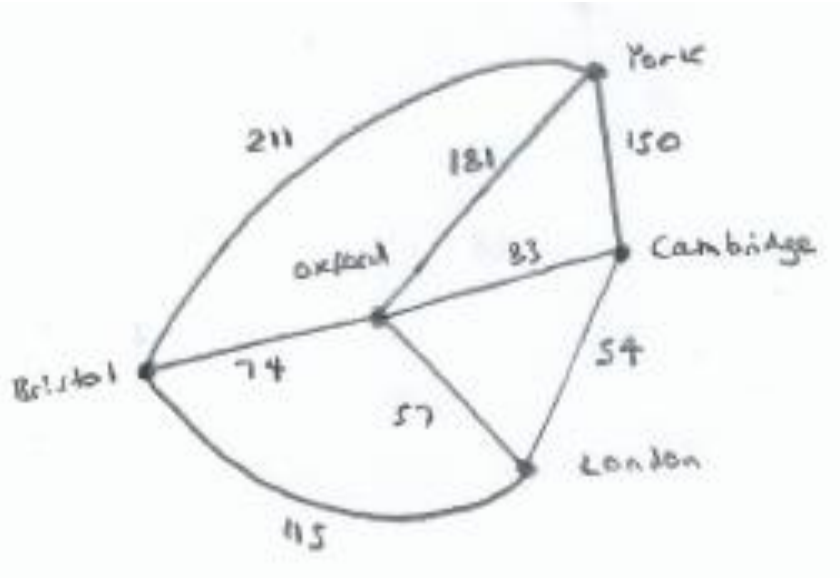
BYOCLB (211+181+83+54+115=644)

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

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

Upper bound so far: 574.

Upper Bound: Doubling the Minimum Connector



The method for finding a tour is as follows:

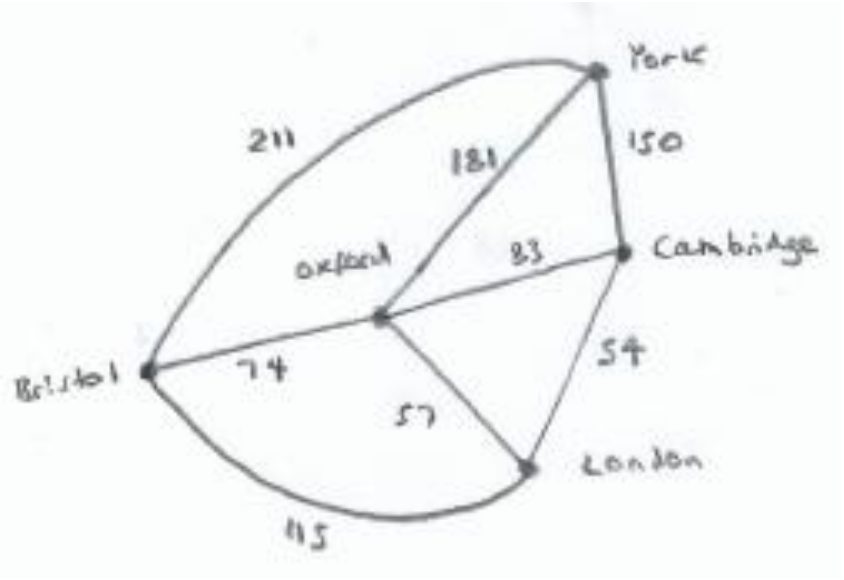
(i) Find a minimum connector for the network.

(ii) Duplicate each of the arcs.

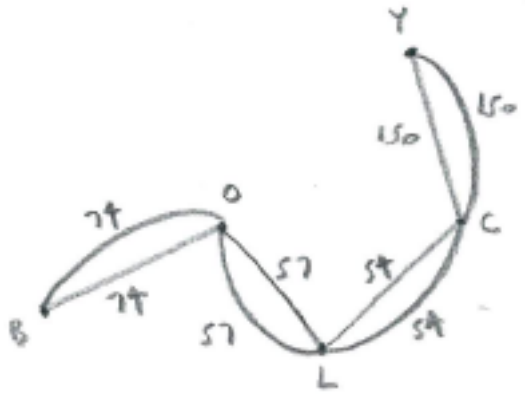
(iii) Use the duplicated minimum connector to construct a tour.

(Clearly nodes are being repeated though.)

Have a go at this, starting the tour at Bristol.







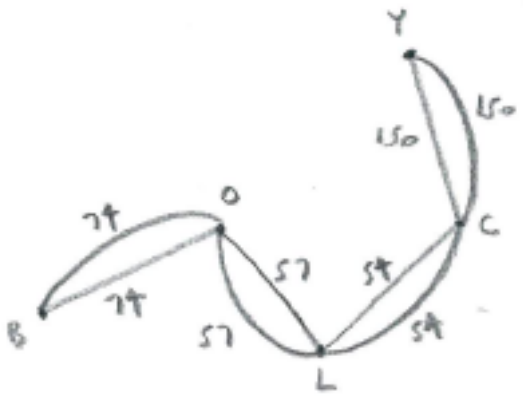
This produces a length of $(74 + 57 + 54 + 150) \times 2 = 670$

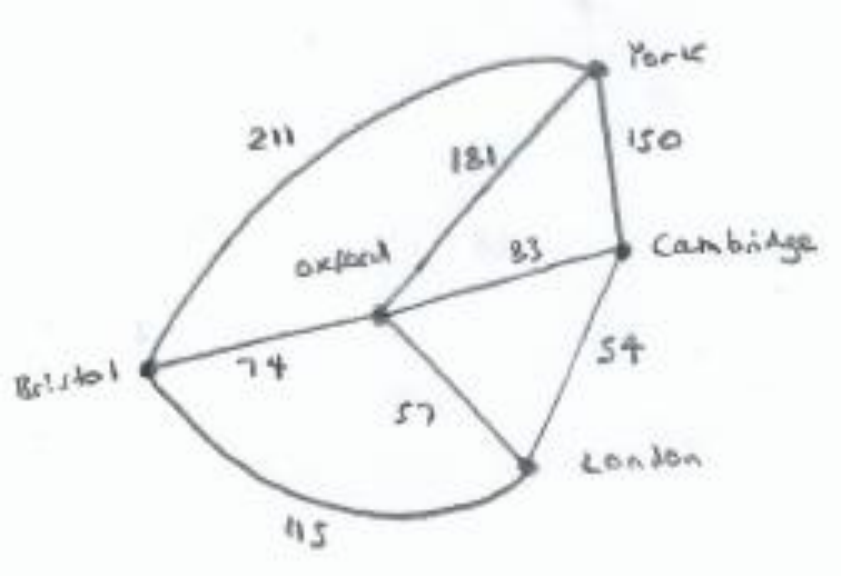
with the tour being BOLCYCLOB.

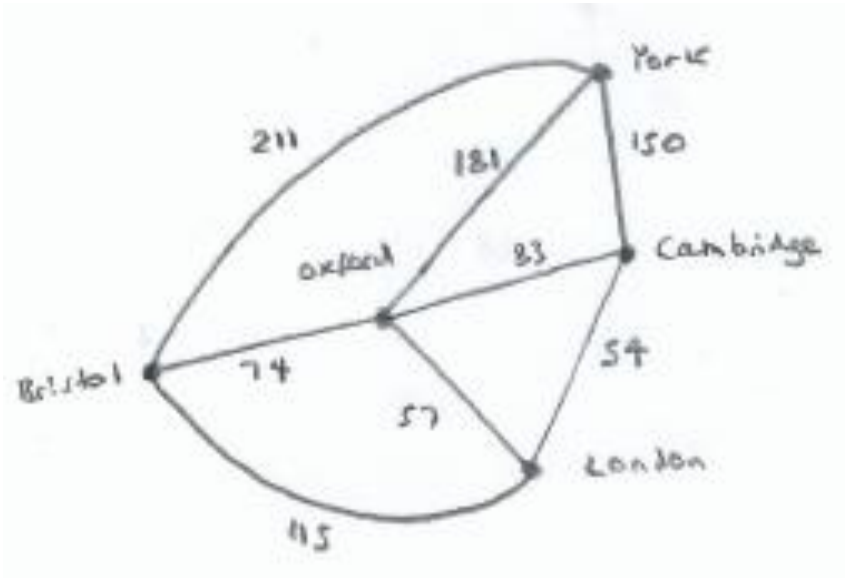
This isn't an improvement on the existing upper bound of 574.

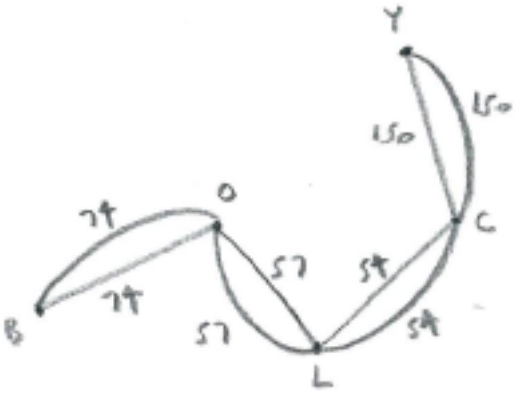
The Minimum Connector method can usually be improved on significantly by taking one or more shortcuts:

Can you spot any shortcuts?









Initially we have BOLCYCLOB (length: 2×335)

Systematic improvements can be made as follows:

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

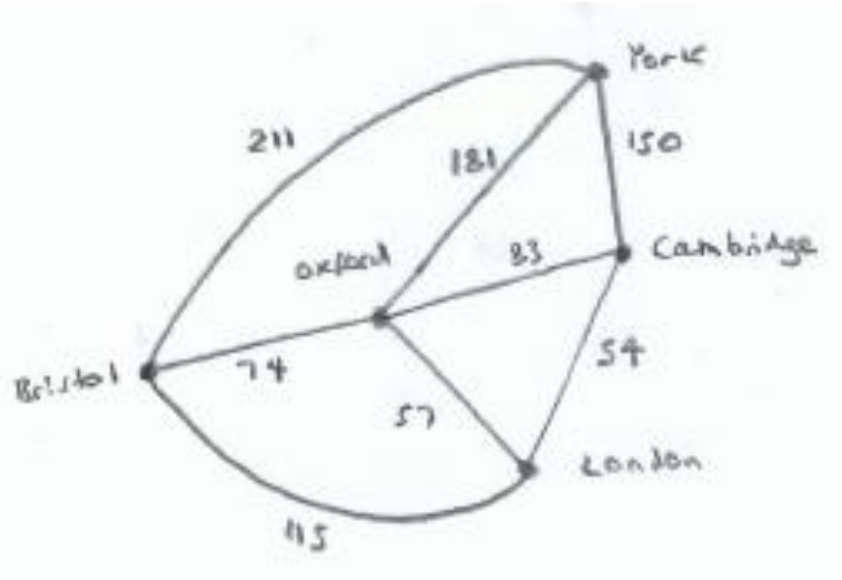
BOLC**YO**B: $335 + 181 + 74 = 590$

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

Thus the improved upper bound is 546. This is an improvement on the 574 we found by inspection.

Upper Bound: Nearest Neighbour Algorithm

This is the standard algorithm for producing an upper bound.



(i) Start at any node (eg B)

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

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

(iv) Return to B **by a direct arc**, to give the cycle:

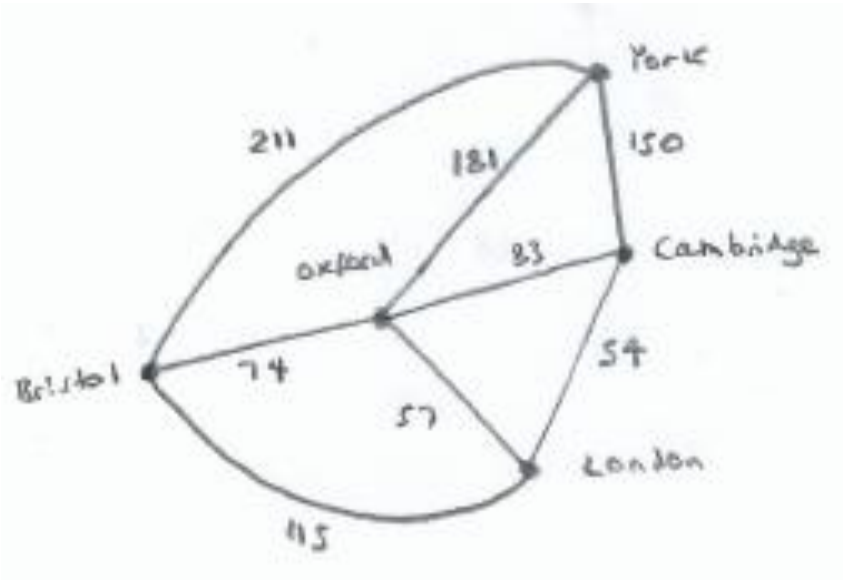
BOLCYB (74+57+54+150+211=546)

How does this differ from Prim's Algorithm?

- Prim's algorithm joins the nearest new node to any existing node, whereas the Nearest Neighbour algorithm joins it to the last node obtained.

- Prim's algorithm is designed to produce a tree and we don't return to the start node.

Repeat the algorithm, with other starting points

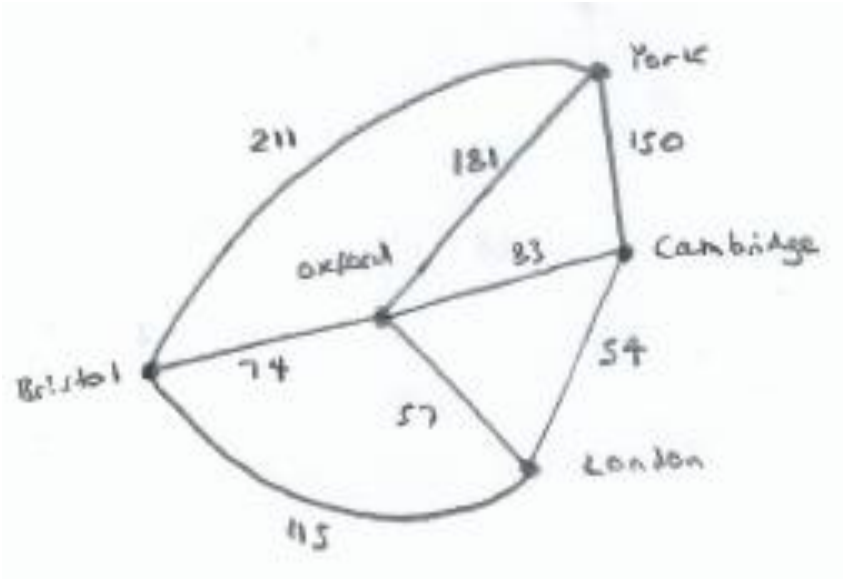


(i) Start at any node.

(ii) Add the shortest arc leading to a new node.

(iii) Repeat the process.

(iv) Return to the starting point by a direct arc, to give a cycle.



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

LCOBYL (54+83+74+211+204=626)

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

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

This gives an upper bound of 546, but we already found this value earlier (by doubling the Minimum Connector, and looking for shortcuts).

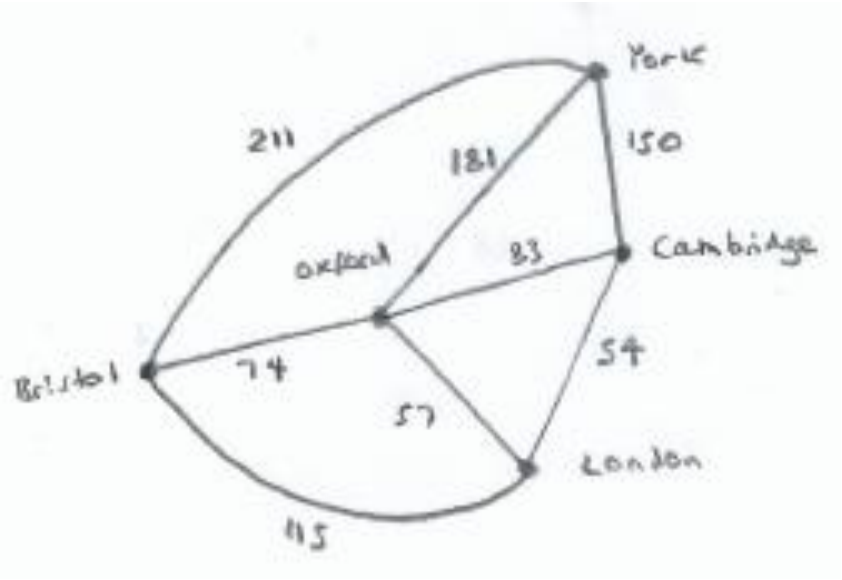
Notes

(i) The Nearest Neighbour Algorithm is a 'greedy' algorithm: it doesn't look ahead, and just maximises the short-term gain (by selecting the nearest node).

(ii) Although the solution will usually be much better than for the Minimum Connector method, the Nearest Neighbour algorithm doesn't usually give the best possible solution.

(iii) An improvement on the algorithm would be to return by the shortest route (not necessarily directly) for the last step.

Upper Bound: Tour improvement algorithm



The 3 possibilities that were discovered by inspection were:

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

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

(obtained from BYOCLB by swapping O & C)

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

(obtained from BYOCLB by swapping Y & O)

The Tour improvement 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

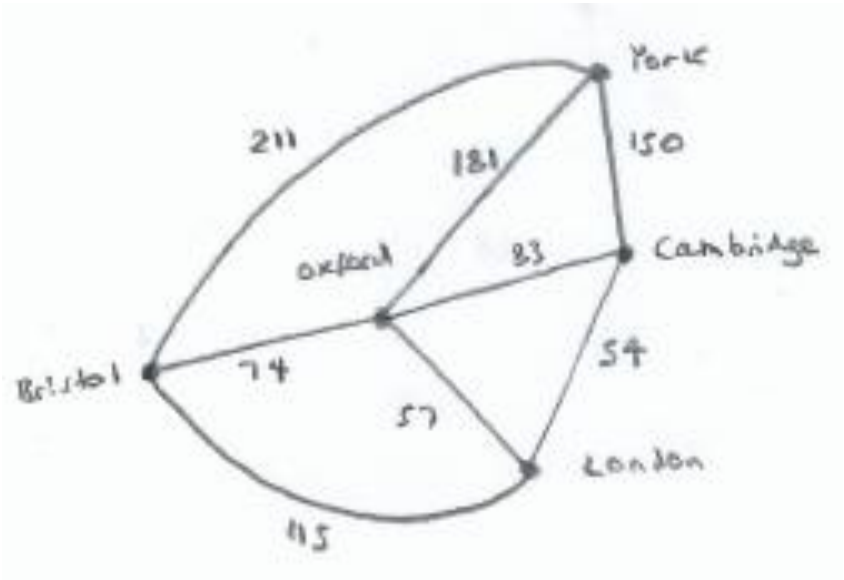
(where N_i denotes the ith node, and d denotes distance).

Note: ABCD will be replaced by ACBD if

$AC + CB + BD < AB + BC + CD$;

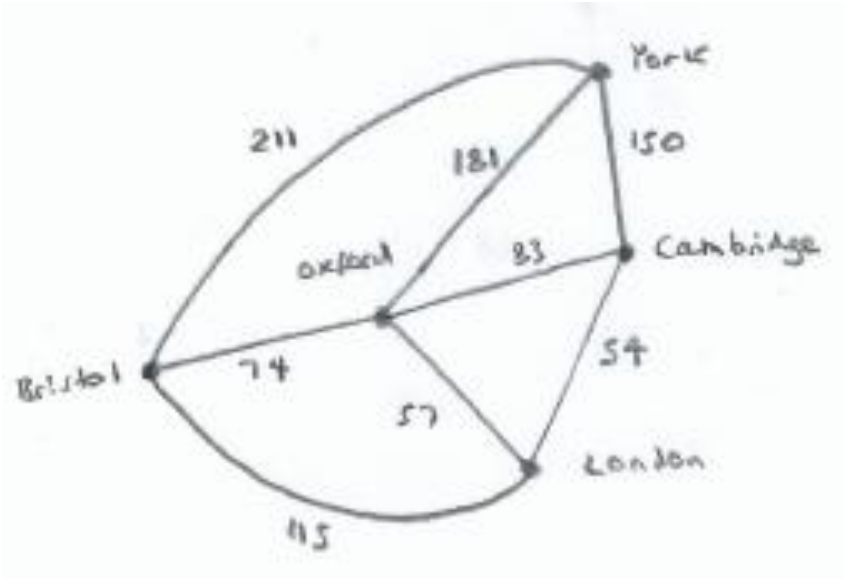
ie if $AC + BD < AB + CD$

Lower Bound Algorithm (using Minimum Connector)



Any Hamiltonian cycle [nodes included exactly once – so no arcs repeated] will consist of 2 arcs from (eg) B, together with 3 arcs linking O, L, C & Y (the 'residual spanning tree')

What will be the minimum total length that could possibly be achieved (based on these considerations alone)?



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.

Applying Prim's Algorithm, for example, starting at O:

OL(57)

OL(57)+LC(54)

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

[Alternatively, applying Kruskal's Algorithm:

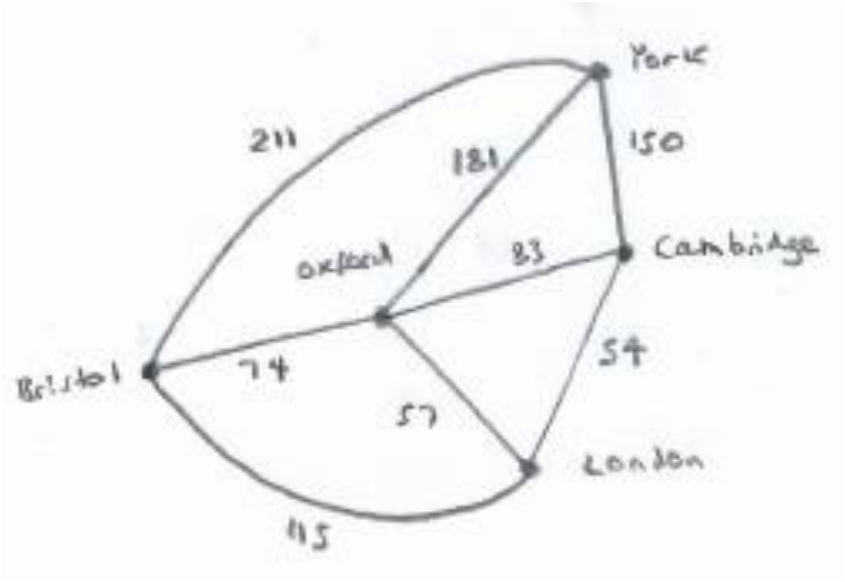
LC(54)

LC(54)+LO(57)

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

Hence the lower bound (so far) is 189+261=450

Improved (ie bigger) lower bounds can be established by dividing up the nodes differently. Try this



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, for example, it is also true that it can't be lower than 516.]

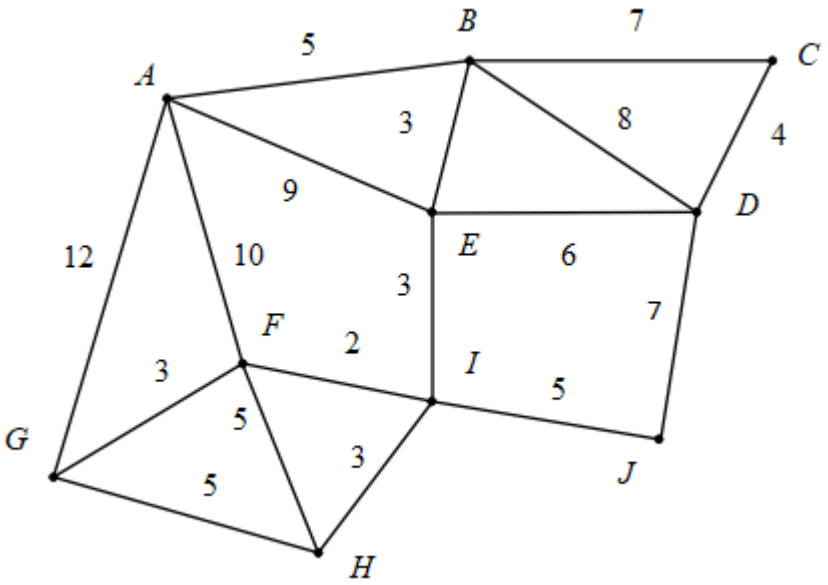
Conclusion

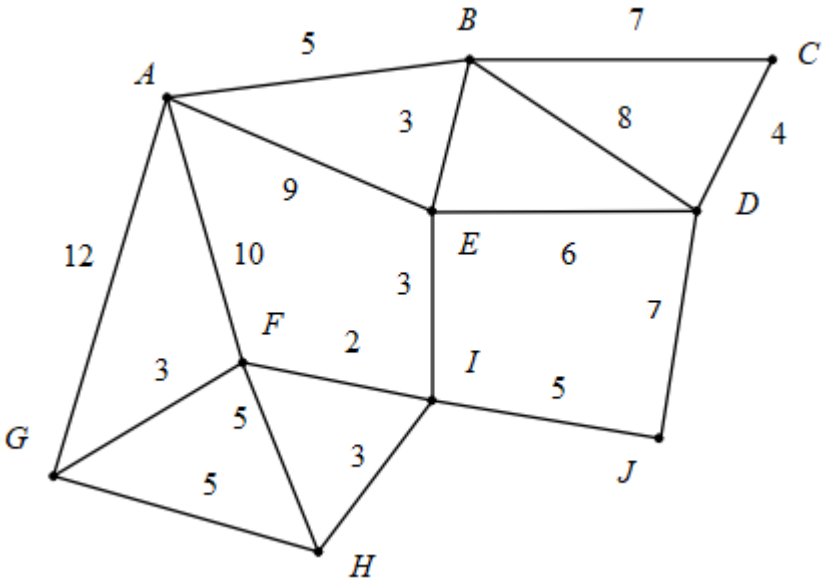
Upper Bound: 546

Lower Bound: 516

Exercise

(i) For the network below, use the Lower Bound [minimum connector] algorithm to find a lower bound for a Hamiltonian cycle, by isolating A.



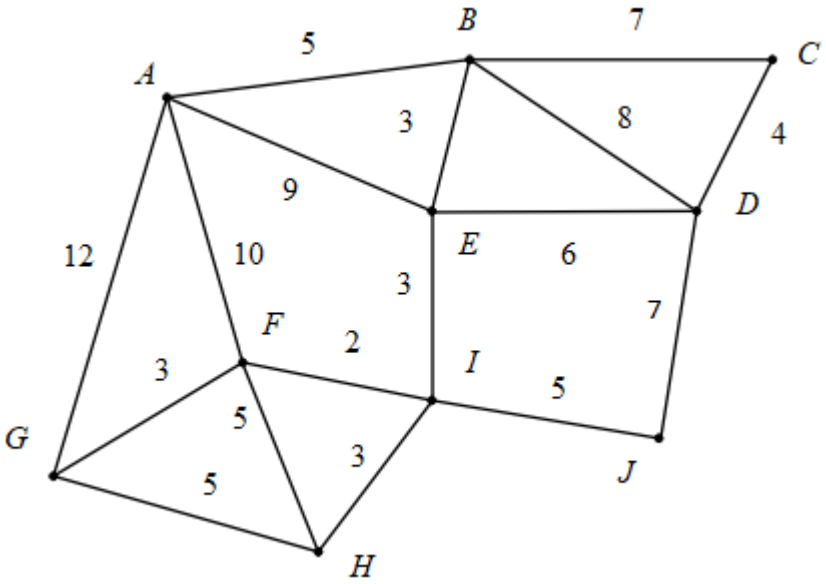


(i) The two shortest arcs leading from A are AB and AE, with total length 14.

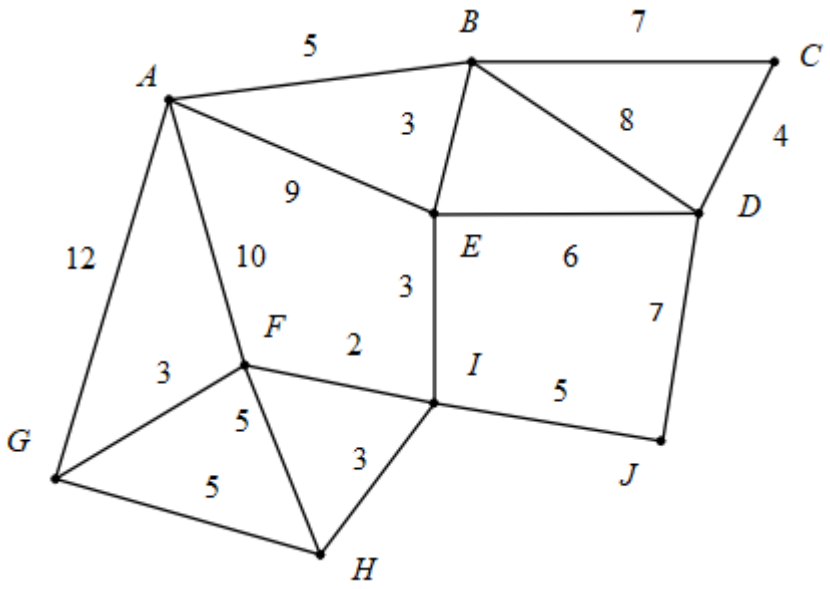
The minimum connector for the remaining nodes can be found as follows, using Prim's algorithm (for example), starting at B:

BE 3, EI 3, IF 2, FG 3, IH 3, IJ 5, ED 6, DC 4 ; total length 29

So the lower bound is $14 + 29 = 45$.



(ii) Use the Nearest Neighbour algorithm to find an upper bound for a Hamiltonian cycle, starting at A.



Solution

(ii) AB(5), BE(3), EI(3), IF(2), FG(3), HI(3), IJ(5), JD(7), DC(4), CB(7), BA(5)

Total length: 47

So we have a lower bound of 45, and an upper bound of 47. Our best tour so far has distance 47.