Route Inspection (Chinese Postman) Problem

(5 pages; 12/9/17)

The aim is to find the shortest route that covers all the arcs at least once, returning to the start node.

A good example of this is a gritting lorry wanting to cover all the roads in a particular area, with the minimum of duplication, and returning to its starting point (assuming that the lorry can grit both sides of the road at the same time).

First of all, here is a summary of the relevant results for graphs (and hence networks – which are graphs with weights attached) concerning 'odd' nodes (nodes with an odd number of arcs):

(i) if there are no odd nodes, then it is possible to cover each arc once only, returning to the starting point ['Eulerian' graph]

(ii) if there are 2 odd nodes, then it is possible to cover each arc once only, but not being able to return to the starting point ['Semi-Eulerian' graph]

(iii) there will always be an even number of odd nodes

<u>Note on terminology</u>

The use of the term 'Eulerian' is fairly confusing (Euler is normally pronounced 'Oiler', by the way):

An 'Eulerian trail' is a trail in a graph which visits every arc exactly once, whilst an 'Eulerian cycle' (or 'Eulerian circuit') is an Eulerian trail which starts and ends on the same node.

An 'Eulerian graph' is thus a graph with an Eulerian cycle, whilst a 'Semi-Eulerian' graph is a graph with an Eulerian trail, but not an Eulerian cycle!

A 'traversable' graph is one that is either Eulerian or Semi-Eulerian (ie has an Eulerian trail, but may or may not have an Eulerian cycle!!

[Warning: The MEI D1 book includes the question “Explain why a graph with odd vertices cannot be traversable” (3rd Ed, page 54), but in D2 there is the statement “For a network to be traversable it must have zero or two odd nodes” (3rd Ed, page 67).]

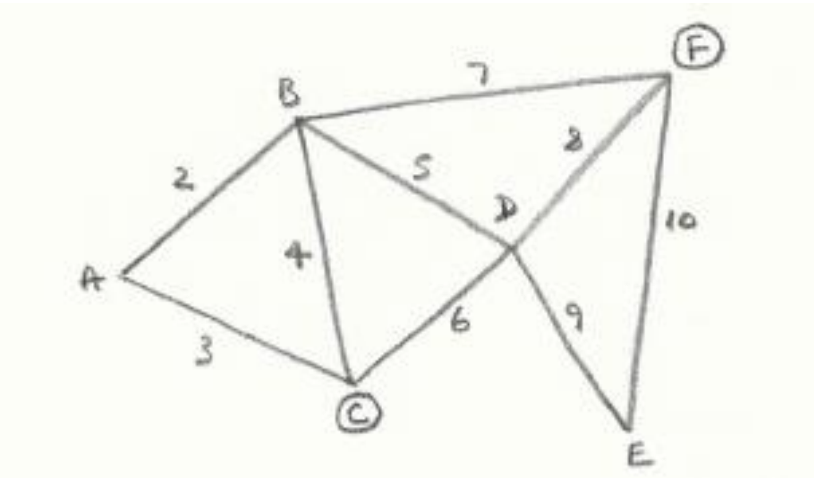
[See also the Graphs Glossary note.]

For our problem, situation (i) is ‘trivial’: the length of the shortest route is just the total of the arc lengths.

Situations (ii) and (iii) are dealt with by repeating arcs in the network in such a way that all the odd nodes are removed; thus turning it into situation (i). This is achieved by linking the odd nodes by the shortest possible routes.

For situation (ii), where there are just 2 odd nodes, Dijkstra’s algorithm can be used to establish the shortest route between the 2 odd nodes (often this can be done by inspection though). Note that these 2 nodes may or may not be directly linked in the network. Even if they are directly linked, the shortest route between them may not be the length of the arc joining them.

Example



Here there are 2 odd nodes: C and F. The shortest route between them is CBF, and this path is repeated in the network, in order to enable the starting point to be returned to.

Since the total of the original arcs is 2+3+4+5+6+7+8+9+10 = 54, the effect of adding on the repeated arcs is to give a figure of 54+4+7= 65 for the shortest route allowing all arcs to be covered at least once, and returning to the starting point.

For example, starting at A, a possible route is:

ABFEDFBDCBCA (which repeats CB and BF).

For situation (iii), more work is involved. We need to decide how to pair up the odd nodes in such a way that the total of the shortest routes between the paired nodes is minimised.

To do this, first of all find the shortest path between each pair of odd nodes (in the case of 4 nodes: AB, AC, AD, BC, BD, CD).

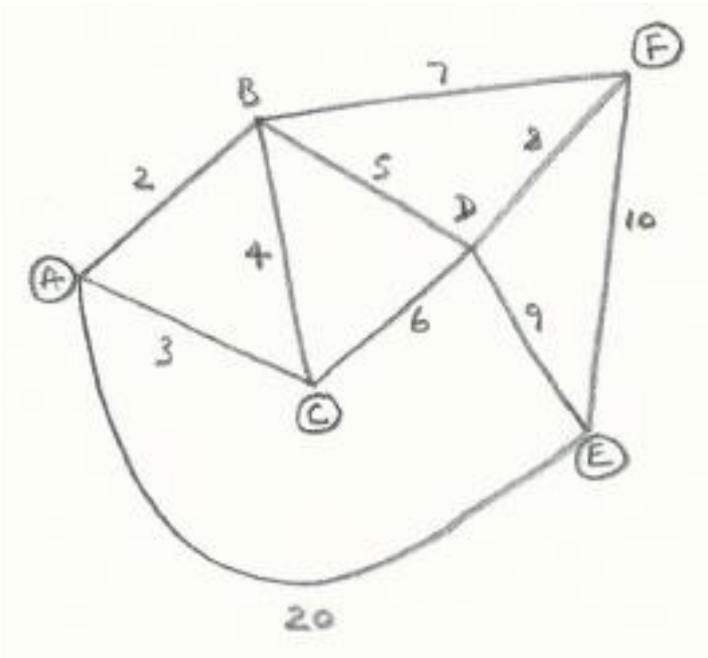
Then establish all the possible ways of pairing up the odd nodes (in the case of 4 nodes, there will be 3 possibilities: AB CD, AC BD & AD BC).

Then choose the combination of pairings that gives the shortest total path (eg AC BD, if AC+BD is the smallest possible total).

This total is the additional distance that has to be added on to the original total of all the arcs, to give the shortest route.

In an exam question, it is highly unlikely that there will be more than 4 odd nodes, since in the case of 6 odd nodes, for example, it can be shown that there are 15 possible ways of pairing up the odd nodes.

Example



The 4 odd nodes are A, C, E & F.

The possible pairs, together with the shortest distances between them (by inspection), are:

AC 3

AE 16 (ABDE)

AF 9 (ABF)

CE 15 (CDE)

CF 11 (CBF)

EF 10

The possible ways of pairing up these nodes, together with the total distances in each case, are:

AC EF 3+10=13

AE CF 16+11=27

AF CE 9+15=24

The combination that gives the shortest total distance is thus AC EF. Since the total of the original arcs is 2+3+4+5+6+7+8+9+10+20 = 74, the effect of adding on the repeated arcs is to give a figure of 74+13=87 for the shortest route allowing all arcs to be covered at least once, and returning to the starting point.

A possible route is: ABFDBCDEFEACA (which repeats EF and AC).