Unit 8: Algorithms on Graphs II (part 2)

Return to Overview

SPECIFICATION REFERENCES

3.2 The practical and classical Travelling Salesman problems. The classical problem for complete graphs satisfying the triangle inequality.

Determination of upper and lower bounds using minimum spanning tree methods.

The nearest neighbour algorithm.

PRIOR KNOWLEDGE

Covered so far

- Tours and Hamiltonian cycles (for the Travelling Salesman problem) (Unit 1a)
- Introduction to graph theory (Unit 1c)
- Dijkstra's algorithm (Unit 2b)
- Eulerian and semi-Eulerian graphs (for the Route Inspection Problem) (Unit 1c)

KEYWORDS

Traversable, odd valency, Eulerian, Semi-Eulerian, minimum weight, upper bound, lower bound, nearest neighbour algorithm, complete network, triangle inequality, walk, tour.

8a. Travelling salesman problem (3.2)

Teaching time
9 hours

OBJECTIVES

By the end of the sub-unit, students should:

- understand the travelling salesman problem and that there is no simple algorithm to solve it for complex networks;
- be able to use the nearest neighbour algorithm to find upper bounds for the problem;
- be able to find lower bounds for a problem;
- understand that not all upper and lower bounds give a solution to the problem;
- know how to identify the best upper and lower bounds;
- be able to solve the travelling salesman problem and interpret this solution in the context of the problem.

TEACHING POINTS

Explain the difference between the classical and practical problems, and how if you convert a network into a complete network of least distances, the practical and classical problems are the same.

Mention the triangle inequality: the longest side ≤ the sum of the two shorter sides.

Explain how to find the upper bound (and that the aim is to make this as low as possible):

- Minimum spanning tree method – find the minimum spanning tree (make clear which method you're using) and double it. You may need to then seek shortcuts, identified in the original table of weights.
- Nearest neighbour algorithm – select each vertex in turn as the starting point and complete the nearest neighbour algorithm. Repeat until all vertices have been used as the start vertex and select the shortest tour. Do not confuse this with Prim's algorithm!

Explain how to find the lower bound (and that the aim is to make this as high as possible):

- Residual minimum spanning tree method – remove each vertex (with its arcs) in turn, find the RMST (normally using Prim's), then reconnect the deleted vertex using its two shortest arcs. NB – this doesn't usually make a tour, but if it does, you have found an optimal solution.

Represent the interval in which the solution is contained using an inequality: L.B. $< d \leq$ U.B.

Notation again is the key to success. For the upper bound students must identify the tour which they are using not just give a numerical answer. For lower bounds students must identify which edges they are choosing and not just give a numerical answer. It is useful for students to sketch the edges they are using to check a cycle is not formed.

Students can be asked to solve these problems from a matrix rather than a network and must be aware of the correct notation for doing this.

Students should be exposed to problems where the upper bound is not just following the direct path from vertex to vertex but involves going via another vertex.

In the exam, students are not likely to find an optimal solution, but instead find a range of values that the optimum route must lie between. At the same time, they should recognise when an optimal solution has been obtained.

OPPORTUNITIES FOR REASONING/PROBLEM SOLVING

Ask questions such as:

How can you represent the different routes in a way that makes sure you don't miss any?

If I add in an extra city, how does that affect the number of different possible routes?

COMMON MISCONCEPTIONS/ EXAMINER REPORT QUOTES

Examiner comments highlight the importance of good, clear notation:

“Most students knew the required algorithm for finding a lower bound but a significant minority still lost marks by not clearly detailing which edges were involved. Lettered labelling is required, not just numbers.”

“When finding the upper bound the students who elected to write their tour down, vertex by vertex, underneath the table usually earned full marks. Those who elected to work entirely in the table often failed to make clear the order in which vertices had been selected and thus that they had actually used the nearest neighbour algorithm. However when specifically asked to work on the table students must show a method for clearly showing the nearest neighbour algorithm has been used.”

Students should be warned against the similarities between Prim’s algorithm (see 2a) and the nearest neighbour algorithm, and ensure that they do not confuse the two.

NOTES

The use of shortcuts to improve the upper bound is included.

The conversion of a network into a complete network of shortest ‘distances’ is included.