Unit 2: Algorithms on Graphs I (part 1)

Return to Overview

SPECIFICATION REFERENCES

2.1 The minimum spanning tree (minimum connector) problem. Prim's and Kruskal's (greedy) algorithm.

2.2 Dijkstra's algorithm for finding the shortest path.

PRIOR KNOWLEDGE

Covered so far

- Introduction to graph theory (See Unit 1c)

KEYWORDS

Minimum spanning tree, Kruskal's algorithm, Prim's algorithm, network, distance matrix, Dijkstra's algorithm, working values, final values, directed network, source vertex, destination vertex.

2a. Minimum connectors (spanning trees) (2.1)

Teaching time
4 hours

OBJECTIVES

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

- **understand the meaning of a minimum spanning tree;**
- **be able to apply Kruskal's algorithm to a network to find the minimum spanning tree;**
- **be able to apply Prims algorithm to a network to find the minimum spanning tree;**
- **be able to apply Prim's algorithm to a distance matrix to find the minimum spanning tree.**

TEACHING POINTS

Make links to spanning trees in graph theory (arc, weight, tree, cycle, vertices)

Students are regularly asked to identify the number of edges in a spanning tree with a given number of vertices. In a network of n vertices a spanning tree will always have $(n - 1)$ arcs.

Students must show their working out clearly to make it clear they have used the stated algorithm. The best method is to write the edges used in order (make sure students are writing edges e.g. AB not just vertices e.g. A) with their decision about rejecting or adding them where appropriate.

Students should be aware of the similarities and differences between the two algorithms and be aware they find the same answer.

Students should be given problems where it is possible to find more than one spanning tree.

They must be able to show the use of Prim's algorithm from a matrix, presenting the final labelled table, plus a list of arcs in order, to make their method clear.

OPPORTUNITIES FOR REASONING/PROBLEM SOLVING

Ask questions such as the following:

What does the full network look like?

Would the minimum spanning tree remain the same if this edge was added?

Can you draw a different spanning tree?

Can you find a shorter tree?

What strategy did you use?

Can you write your strategy in a formal way (as an algorithm)?

What are the similarities and differences?

COMMON MISCONCEPTIONS/ EXAMINER REPORT QUOTES

Questions which require Prim's algorithm on a table are sometimes poorly answered. Many students do not present their working on the table and of those that do, they often do not show the order of selection of edges. Students should be encouraged to record the edges as they add them to the tree, rather than after they have completed the algorithm, to avoid making mistakes.

It should be noted that whilst a question on Kruskal often follows one on sorting a list in to ascending order, the sort is not part of Kruskal's algorithm itself and so is not necessary unless asked for in a separate part of the question.

Students often try to merge the methods of Prim and Kruskal, for example by incorrectly showing rejections when using Prim.

Students should be warned against the similarities between Prim's algorithm and the nearest neighbour algorithm (see 3b), and ensure that they do not confuse the two.

NOTES

Students will gain no credit for using the incorrect algorithm when finding the minimum spanning tree.

Matrix representation for Prim's algorithm is expected. Drawing a network from a given matrix and writing down the matrix associated with a network may be required.

2b. Dijkstra's algorithm (2.2)

Teaching time
4 hours

OBJECTIVES

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

- **be able to apply Dijkstra's algorithm to find the shortest path between two vertices in a network;**
- **be able to trace back through a network to be able to find the route corresponding to the shortest path;**
- **be able to consider modifications to an original shortest path problem, for example by dealing with multiple start points or a different end point.**

TEACHING POINTS

Dijkstra is pronounced Dike-stra.

The boxes will be drawn for the students to complete in the exam. Students' method should be clear from their working and they need to be aware that examiners check the order in which the numbers appear in the list of working values.

Include networks with directed arcs in your examples.

Some students find this algorithm difficult when they can already see the correct answer; they must continue to follow the algorithm as missing out one path will lose multiple marks in exam questions.

They must be able to *explain* how they found their quickest route by back tracking. They can do this using a series of label calculations.

OPPORTUNITIES FOR REASONING/PROBLEM SOLVING

Exam questions regularly ask students to find solutions to similar problems where new information is given which would update the network. This could include dealing with multiple starting points.

They should be exposed to problems where the shortest route is not unique.

COMMON MISCONCEPTIONS/ EXAMINER REPORT QUOTES

As highlighted above, an examiner reports: "Candidates had to apply Dijkstra's algorithm very carefully to obtain full marks in part and many were able to do this. There are still a few candidates who treat this as a sort of 'minimising critical path' forward pass however. The examiners use the working values and the order in which they occur in the box as the main confirmation that Dijkstra's algorithm has been correctly applied, thus it is important that they are legible and that candidates do write them in order."

When describing how they found their shortest route, students that give a numerical demonstration gain full marks most easily, with those who give a general explanation often missing out part of the process.

NOTES

Some students confuse Dijkstra and activity networks.