InkMaths

ANSWER BOOK

Minimum spanning trees: Prim and Kruskal

Decision Mathematics 1 · Year FM

6 questions · 22 marks

NAVY EDITION

inkphysics.co.uk/maths

SECTION A · Warm-up

A1 A subset of the edges that connects every node with no cycle, chosen to have the least possible total weight. It always has exactly n - 1 edges: any fewer leaves the network disconnected and any more creates a cycle. B1 for the definition, B1 for n - 1 edges. [2]

A2 **Prim**: start at any node and repeatedly add the cheapest edge joining the growing tree to a node not yet in it. **Kruskal**: sort the edges by weight and take them in order, rejecting any that would close a cycle. Both always produce a minimum spanning tree, so the totals always agree. B1 for Prim, B1 for Kruskal, B1 for what they share. [3]

SECTION B · Standard

B1 From {A} the choices are AB 3 and AC 5, so take **AB**.
From {A, B}: AC 5, BC 4, BD 8, BE 13, so take **BC** (4).
From {A, B, C}: BD 8, CD 9, CE 11, BE 13, so take **BD** (8). AC has dropped out, because C is already in the tree.
From {A, B, C, D}: DE 7, CE 11, DF 12, BE 13, so take **DE** (7).
From {A, B, C, D, E}: EF 6 beats DF 12, so take **EF**.
Order **AB,** BC, BD, DE, EF, weight 3 + 4 + 8 + 7 + 6 = **28**, five arcs for six nodes. M1 for a correct Prim scan, A1 for the order of arcs, A1 for the complete tree, A1 for the weight 28. Give the arcs in the order chosen rather than sorted; the order is what the method mark is for. [4]

B2 Sorted: AB 3, BC 4, AC 5, EF 6, DE 7, BD 8, CD 9, CE 11, DF 12, BE 13.
Take **AB** (3), then **BC** (4).
Reject AC (5): A and C are already joined through B, so it would close the cycle ABC.
Take **EF** (6), then **DE** (7), then **BD** (8), which joins the two halves.
Five arcs, so stop. Weight 3 + 4 + 6 + 7 + 8 = **28**. M1 for sorting the arcs, A1 for the arcs accepted in order, B1 for rejecting AC with a reason, A1 for the weight 28. Naming the rejected arc and saying why costs one line and earns a mark; a list of accepted arcs alone does not show the algorithm. This is the same tree Prim builds, reached in a different order, which is what always happens when no two weights are equal. [4]

B3 Prim reads straight off the matrix. Delete the row of each node as it joins, then scan the columns of the joined nodes for the smallest remaining entry. Kruskal would need every entry sorted into a single list first, and then a cycle check on each arc in turn. B1 for what Prim does with the matrix, B1 for the extra work Kruskal needs. [2]

SECTION C · Stretch

C1 [7]

Delete row A and scan column A: the smallest entry is 15, so **AD** brings in D.
Scan columns A and D: 17, 23, 30, 25, 28, 21, 24, 20. The smallest is 17, so **AB** brings in B.
Scan A, D and B: the smallest is 16, so **BE** brings in E.
Scan A, D, B and E: the smallest is 18, so **EF** brings in F.
Only C remains, and its cheapest link is CF 14, so **FC** brings in C.
Sites join in the order **A,** D, B, E, F, C, using AD, AB, BE, EF, FC. Least total cost 15 + 17 + 16 + 18 + 14 = **80**.
Now rerun from A with the BE entry struck out. AD 15 and AB 17 are unchanged. From {A, B, D} the cheapest available link is now BC 19, then CF 14, then FE 18. The tree is AD, AB, BC, CF, FE, costing 15 + 17 + 19 + 14 + 18 = **83**, so losing BE adds **3** to the bill.
M1 for the Prim scan, A1 for the joining order, A1 for the arcs used, A1 for the cost 80, M1 for rerunning without BE, A1 for the new tree, A1 for the cost 83.
The tempting slip is to keep the first tree and reconnect E by its own next cheapest link, DE 24, giving 80 - 16 + 24 = 88. Deleting BE splits the tree into {A, B, D} and {C, E, F}, and the arc to look for is the cheapest one across that split, not the cheapest one at E. Removing an arc can rearrange the whole tree, and here E ends up reached from F rather than from B.