10.2 Minimum spanning trees and shortest paths

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS minimum spanning tree • Dijkstra's algorithm

Minimum spanning trees and shortest paths

Objective

Find minimum spanning trees by Kruskal and Prim (node and matrix forms) and run Dijkstra with labels in order of permanence, recovering routes (objectives 2.1–2.2).

1 Key knowledge and worked examples

MST. Kruskal: sort all edges, add the smallest that does not form a cycle (reject others); Prim: from a start vertex repeatedly add the cheapest edge joining the tree to a new vertex; the matrix form crosses out rows and rings cheapest column entries. Dijkstra: permanent labels smallest-first; update neighbours (new = permanent + weight if smaller); recover the route by tracing back.

Worked example (original). Edges AB 4, AC 2, BC 1, BD 5, CD 8. Kruskal: BC 1 ✓, AC 2 ✓ (the tree holds A, B, C), AB 4 forms a cycle (rejected), BD 5 adds D ✓, CD 8 rejected (cycle): MST = BC + AC + BD, total 8. Prim from A: AC 2, CB 1, BD 5 – the same tree. Dijkstra A to D: A 0, C 2, B 3, D 8; route A–C–B–D.

2 Practice

2.1 Edges AB 3, AC 1, BC 2, CD 4, AD 5: find the MST by Kruskal, listing edges in order of choice, and give its total weight. [3]

2.2 For the same network, run Prim from vertex B and show the MST is the same. [3]

2.3 Edges AB 2, BC 3, CD 1, AD 4, AC 2: use Dijkstra from A to find the shortest distance to D, showing each label as it becomes permanent, and state the route. [4]

3 Exam-style questions

3.1 *(Original.)* A network connects four villages P, Q, R, S with edge weights PQ 7, PR 3, PS 8, QR 2, RS 6, QS 5 (km). (a) Use Kruskal's algorithm to find the minimum spanning tree, listing the edges in the order chosen and rejecting cycles explicitly. (b) Use Prim's algorithm from P to find the same tree. (c) State the total weight of the MST and explain why both algorithms must agree. [8]

3.2 *(Original.)* With the same network, use Dijkstra's algorithm from P to find the shortest route from P to S, showing the order in which labels become permanent and the final route with its length. [6]

Solutions

2.1 Sorted: AC 1, BC 2, AB 3, CD 4, AD 5. Take AC(1) ✓; BC(2) ✓ (B joins); AB(3) – cycle (A–C–B) reject; CD(4) ✓ (D joins); AD(5) – cycle reject. MST = AC + BC + CD, total 7.

2.2 Prim from B: cheapest edge from {B} is BC(2); from {B,C} cheapest to a new vertex is CA(1); from {B,C,A} cheapest to D is CD(4). Same tree, total 7 ✓.

2.3 A = 0 permanent. Neighbours: B 2, C 2. B permanent (2) – D has no edge from B, so no update. C permanent (2; the tie with B may be broken in either order); update D: $2 + 1 = 3$, smaller than the direct AD 4. D permanent (3): shortest A to D = 3, route A–C–D ($2 + 1 = 3 < 4$ ✓).

3.1 (a) Sorted: QR 2, PR 3, QS 5, RS 6, PQ 7, PS 8. Take QR(2) ✓; PR(3) ✓ (P joins); QS(5) ✓ (S joins); RS(6) – cycle reject; PQ(7) – cycle reject; PS(8) – cycle reject. MST = QR + PR + QS, total 10. (b) Prim from P: PR(3); from {P,R} cheapest new edge RQ(2); from {P,R,Q} cheapest to S is QS(5). Same tree, total 10 ✓. (c) Both construct a spanning tree of least total weight; any MST of a network has the same minimum total, so the totals agree (the edges can differ only when weights tie).

3.2 P = 0 permanent. Update: R 3, Q 7, S 8. R permanent (3); update Q: $3 + 2 = 5$ (smaller than 7), S: $3 + 6 = 9$ (8 stays). Q permanent (5); update S via Q: $5 + 5 = 10 > 8$, so S keeps 8. S permanent (8). Shortest P to S = 8, route P–S (the alternatives P–R–S = $3 + 6 = 9$ and P–R–Q–S = $3 + 2 + 5 = 10$ are both longer ✓).

Local review guidance: edges chosen in weight order with cycle rejections shown; Prim grows by cheapest new-vertex edge, not cheapest free edge; Dijkstra labels in permanence order with updates shown. This is authored classroom guidance.