2.6 Advanced Algorithms: Searching, Sorting and Path-Finding: 30-minute Pack

OCR H446 · Calculators not allowed · about 30 minutes

Total Marks

Name: _____ Date: ____ / ____ / _____

Answer ALL questions. Show all your working.

1 An admissible heuristic h(v) for a search from vertex v to a goal G never overestimates the true shortest remaining distance from v to G, i.e. h(v) ≤ actualDistance(v, G) always. A consistent (monotone) heuristic additionally satisfies h(u) ≤ cost(u, v) + h(v) for every edge (u, v). A* is only guaranteed to find the shortest path when its heuristic is admissible.

(a) In the courier network used earlier in this paper, the true shortest remaining journey time from D to F is 5 minutes, and from E to F is 3 minutes. State, with a reason, whether the heuristic values h(D) = 4 and h(E) = 2 used earlier are each admissible. **(2)**

(b) Show that the heuristic values h(D) = 4 and h(E) = 2 satisfy the consistency condition for the edge D-E (weight 2). **(2)**

(c) A much simpler graph has three vertices: S (start), X, and G (goal), plus a fourth vertex Y. Edges: S-X weight 1, X-G weight 5, S-Y weight 1, Y-G weight 1. The true shortest distance from S to G is therefore 2 (via Y: S-Y-G), not 6 (via X: S-X-G). Suppose A* is run using the heuristic values h(X) = 0 and h(Y) = 10 (note: h(Y) = 10 is NOT admissible, since the true remaining distance from Y to G is only 1). Complete the trace below to show which vertex A* expands at each step, and state the path and total distance A* returns. **(6)**

step	vertex expanded	g, f of vertices reached
1	S	X: g=1, f=1 ; Y: g=1, f=11
2		
3		

(d) State what this example demonstrates about the importance of the admissibility condition for A* to guarantee an optimal (shortest) result. **(1)**

(Total for Question 1 is 11 marks)

© Revision Library | revisionlibrary.co.uk

2 A warehouse robot navigates a grid-shaped map with R rows and C columns of cells. Each cell is connected to its (up to 4) directly adjacent cells (up, down, left, right), forming a graph where each cell is a vertex.

(a) State, in terms of R and C, the number of vertices V in this graph. **(1)**

(b) Show that the number of edges E in this grid graph is given by E = R(C - 1) + C(R - 1). **(2)**

(c) The warehouse grid has R = 100 and C = 100 (a 100 x 100 grid). Calculate V and E for this grid. **(3)**

(d) Using the adjacency-list-with-min-heap implementation of Dijkstra/A*, with time complexity O((V + E) log V), calculate an estimate for the number of operations needed to find a route across this 100 x 100 grid. Give your answer to 3 significant figures. **(3)**

(e) In practice, A* with a well-chosen heuristic (such as Manhattan distance, |x1 - x2| + |y1 - y2|, between grid coordinates) typically explores far fewer than V vertices when finding a route between two specific cells, even though the algorithm's worst-case time complexity calculated in part (d) is the same as Dijkstra's. Explain why this is the case. **(2)**

(Total for Question 2 is 11 marks)

© Revision Library | revisionlibrary.co.uk

Mark scheme · 2.6 Advanced Algorithms: Searching, Sorting and Path-Finding: 30-minute Pack

Question 1

(a) h(D) = 4 is admissible because 4 ≤ 5 (the true shortest distance from D to F) **B1**

(a) h(E) = 2 is admissible because 2 ≤ 3 (the true shortest distance from E to F) **B1**

(a) Answer: Both are admissible: h(D)=4 ≤ 5, and h(E)=2 ≤ 3.

(b) checks h(D) ≤ cost(D, E) + h(E): 4 ≤ 2 + 2 **M1**

(b) 4 ≤ 4 is true (holds, with equality), so the heuristic is consistent for this edge cao **A1**

(b) Answer: h(D) = 4 ≤ cost(D,E) + h(E) = 2 + 2 = 4, so the condition holds (with equality).

(c) step 2: expands X (lowest f = 1, since f(Y) = 11) **B1**

(c) step 2: G is reached via X with g = 1 + 5 = 6, f = 6 + 0 = 6 **B1**

(c) step 3: expands G (f = 6, now the lowest in the open list, since f(Y) = 11) **B1**

(c) G is the goal, so A* terminates, returning the path S-X-G with total distance 6 **B1**

(c) recognises this is NOT the true shortest path, which is S-Y-G with total distance 2 **B1**

(c) recognises Y was never expanded before the search terminated **B1**

(c) Answer: Step2: expand X, G reached with g=6, f=6. Step3: expand G (lowest f=6, since f(Y)=11), goal reached, A* returns path S-X-G, distance 6. This is not the true shortest path (S-Y-G, distance 2); Y was never expanded.

(d) if a heuristic overestimates the true remaining distance for even one vertex, A* can terminate having found a suboptimal path, because that vertex is wrongly deprioritised (given a high f value) and so is never explored before the algorithm stops; an admissible heuristic is therefore essential for A*'s optimality guarantee **B1**

(d) Answer: It shows that an inadmissible (overestimating) heuristic can cause A* to terminate with a suboptimal path, because the vertex leading to the true shortest path is wrongly given a low priority and never explored before the search stops.

Question 2

(a) V = R x C **B1**

(a) Answer: V = R x C

(b) each of the R rows contains (C - 1) adjacent horizontal pairs of cells, giving R(C - 1) horizontal edges in total **M1**

(b) each of the C columns contains (R - 1) adjacent vertical pairs of cells, giving C(R - 1) vertical edges in total; adding these gives E = R(C - 1) + C(R - 1) **A1**

(b) Answer: E = R(C - 1) + C(R - 1)

(c) V = 100 x 100 **M1**

(c) E = 100 x 99 + 100 x 99 **M1**

(c) V = 10 000, E = 19 800 cao **A1**

(c) Answer: V = 10,000; E = 19,800

(d) V + E = 10 000 + 19 800 = 29 800 **M1**

(d) log2(10 000) = awrt 13.3 **M1**

(d) 29 800 x 13.3 = awrt 396 000 (accept 390 000 to 400 000) cao **A1**

(d) Answer: Approximately 396,000 operations.

(e) the heuristic gives the algorithm a strong sense of direction towards the specific goal cell, so it prioritises expanding cells that lie roughly between the start and the goal, rather than exploring in every direction **B1**

(e) as a result, in typical (non-worst-case) situations A* settles only a small fraction of the total V cells before reaching the goal, even though the stated worst-case bound O((V + E) log V) - which assumes the heuristic gives little useful guidance - remains identical to Dijkstra's (oe) **B1**

(e) Answer: The Manhattan-distance heuristic directs A* towards cells lying between the start and goal rather than in every direction, so in typical cases it expands far fewer than V cells before reaching the goal, even though its worst-case bound is the same as Dijkstra's.

© Revision Library | revisionlibrary.co.uk