Q1

Process Queue/Stack visited

The strategy is, we try to follow the statements to traverse. If we get a different result, the statement traversal is not valid.

1. False. Starting from A, A

Pop A and push B, C B, C A

Pop B and push D, E C, D, E A, B

Pop C D, E A, B, C

Pop D and push G E, G A, B, C, D

Pop E and push F G, F A, B, C, D, E

Pop G and push H F, H A, B, C, D, E, G

Pop F H A, B, C, D, E, G, F

Pop H empty list A, B, C, D, E, G, F, H

Done. Cannot get a

1. False. Starting from B, B

Pop B and push A, C, E, D A, C, E, D B

Pop A C, E, D B, A

Pop C E, D B, A, C

Pop E and push H, F D, H, F B, A, C, E

Pop D and push G H, F, G B, A, C, E, D

Pop H F, G B, A, C, E, D, H

Pop F G B, A, C, E, D, H, F

Pop G empty list B, A, C, E, D, H, F, G

Done. Cannot get b

1. Starting from B, B

Pop B and push A, C, E, D A, C, E, D B

Pop A C, E, D B, A

Pop C E, D B, A, C

Pop E and push H, F H, F, D B, A, C, E

Pop H and push G G, F, D B, A, C, E, H

Pop G F, D B, A, C, E, H, G

Pop F D B, A, C, E, H, G, F

Pop D empty list B, A, C, E, H, G, F, D

Done. Cannot get c

1. Starting from E, E

Pop E and push C, B, D, F, H C, B, D, F, H E

Pop C and push A A, B, D, F, H E, C

Pop A B, D, F, H E, C, A

Pop B D, F, H E, C, A, B

Pop D and push G G, F, H E, C, A, B, D

Pop G F, H E, C, A, B, D, G

Pop F H E, C, A, B, D, G, F

Pop H empty list E, C, A, B, D, G, F, H

Done. Cannot get d

Using Kruskal’s algorithm:

The shortest path has length 1

Add edge (C, E)

图示

描述已自动生成

No edge with length 1 remains. We look at path with length 2.

Add edge (A, C).

图示

中度可信度描述已自动生成

Add edge (B, E).

图示

描述已自动生成

Add edge (D, E).

图示

描述已自动生成

Add edge (G, H).

图片包含 物体, 游戏机, 钟表, 桌子

描述已自动生成

No edge with length 2 remains. We look at edge with length 3.

Add edge (F, G).

图片包含 图示

描述已自动生成

Though edge (B, C) and edge (A, B) have length 3, adding them would form cycles. So ignore them. No edge with length 3. We look at edge with length 4.

Add edge (E, H).

图片包含 物体, 照片, 桌子, 挂

描述已自动生成

Now every node is reachable. We find the minimum spanning tree.

We visit D.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | F | Inf | None |
| B | F | Inf | None |
| C | F | Inf | None |
| D | T | 0 | None |
| E | F | Inf | None |
| F | F | Inf | None |
| G | F | Inf | None |
| H | F | Inf | None |

D has three neighbors: B, E, G.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | F | Inf | None |
| B | F | 5 | D |
| C | F | Inf | None |
| D | T | 0 | None |
| E | F | 2 | D |
| F | F | Inf | None |
| G | F | 8 | D |
| H | F | Inf | None |

The shortest path is (D, E) of length 2. We visit E. E has four unvisited neighbors: B, C, F, H.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | F | Inf | None |
| B | F | 4 | E |
| C | F | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | F | 6 | E |

The shortest path is (E, C) of length 3. We visit C. C has two unvisited neighbors: A, B.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | F | 5 | C |
| B | F | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | F | 6 | E |

The shortest path is (B, E) of length 4. We visit B. B has one unvisited neighbor: A.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | F | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | F | 6 | E |

The shortest path is (C, A) of length 5. We visit A. A has no unvisited neighbors.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | T | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | F | 6 | E |

The shortest path is (C, A) of length 5. We visit A. A has no unvisited neighbors.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | T | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | F | 6 | E |

The shortest path is (E, H) of length 6. We visit H. H has one unvisited neighbor: G.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | T | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | F | 7 | E |
| G | F | 8 | D |
| H | T | 6 | E |

The shortest path is (E, F) of length 7. We visit F. F has one unvisited neighbor: G.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | T | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | T | 7 | E |
| G | F | 8 | D |
| H | T | 6 | E |

The shortest path is (D, G) of length 8. We visit G. G has no unvisited neighbor.

|  |  |  |  |
| --- | --- | --- | --- |
| Vertex | Visited | Distance | Previous |
| A | T | 5 | C |
| B | T | 4 | E |
| C | T | 3 | E |
| D | T | 0 | None |
| E | T | 2 | D |
| F | T | 7 | E |
| G | T | 8 | D |
| H | T | 6 | E |

No node hasn’t been visited. Done.

Q2

1. Transitive closure:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| C | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| D | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| E | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| F | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| G | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

(A, F) = 1, true. (C, G) = 1, true. (E, F) = 0, false.

(1). H: 1, I: 2, G: 3, E: 4, F: 5, D: 6, C: 7, B: 8, A: 9

(2). H: [1, 1], I: [2, 2], G: [1, 3], E: [1, 4], F: [5, 5], D: [1, 6], C: [1, 7], B: [1, 8], A: [1, 9]

(1). C: 1, E: 2, I: 3, H: 4, G: 5, F: 6, D: 7, B: 8, A: 9

(2). C: [1, 1], E: [2, 2], I: [3, 3], H: [4, 4], G: [4, 5], F: [3, 6], D: [3, 7], B: [1, 8], A: [1, 9]

1. The optimal spanning tree is the better index. When querying reachability for (C, G), the optimal spanning tree returns true since 1 <= 3 <= 7. However, the random spanning tree would return false since 1 < 4. Thus, the random spanning tree don’t contain this information.

Starting from node D, we add in(D), in(E), in(F), in(G), out(B), out(C), out(D).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  |  |  | {D} | {D} | {D} |  |  |  |
| out |  | {D} | {D} | {D} |  |  |  |  |  |
| Processed | F | F | F | T | F | F | F | F | F |

Then we choose node G, we add in(G), in(H), in(I), out(D), out(E), out(F), out(G).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  |  |  | {D} | {D} | {D} | {G} | {G} | {G} |
| out |  | {D} | {D} | {D, G} | {G} | {G} | {G} |  |  |
| Processed | F | F | F | T | F | F | T | F | F |

Then we choose node C, we add in(C), in(D), in(E), out(A), out(B), out(C). (C, D) is covered by D. (C, E) is covered by D.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  |  | {C} | {D} | {D} | {D} | {G} | {G} | {G} |
| out | {C} | {C, D} | {C, D} | {D, G} | {G} | {G} | {G} |  |  |
| Processed | F | F | T | T | F | F | T | F | F |

Then we choose node B, we add in(B), in(C), in(D), out(A), out(B). (B, C) is covered by C. (B, D) is covered by D.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  | {B} | {C} | {D} | {D} | {D} | {G} | {G} | {G} |
| out | {B, C} | {B, C, D} | {C, D} | {D, G} | {G} | {G} | {G} |  |  |
| Processed | F | T | T | T | F | F | T | F | F |

Then we choose node E, we add in(E), in(G), out(C), out(D), out(E). (E, G) is covered by G. (C, E) is covered by D. (D, E) is covered by D.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  | {B} | {C} | {D} | {E, D} | {D} | {G} | {G} | {G} |
| out | {B, C} | {B, C, D} | {C, D} | {D, G} | {E, G} | {G} | {G} |  |  |
| Processed | F | T | T | T | T | F | T | F | F |

Then we choose node F, we add in(F), in(G), in(I), out(D), out(F). (E, G) is covered by G. (F, G) is covered by G. (F, I) is covered by G. (D, F) is covered by D.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in |  | {B} | {C} | {D} | {E, D} | {D, F} | {G} | {G} | {G} |
| out | {B, C} | {B, C, D} | {C, D} | {D, G} | {E, G} | {G, F} | {G} |  |  |
| Processed | F | T | T | T | T | T | T | F | F |

Then we choose node A, we add in(A), in(B), in(C), out(A). (A, B) is covered by B. (A, C) is covered by C.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in | {A} | {B} | {C} | {D} | {E, D} | {D, F} | {G} | {G} | {G} |
| out | {A, B, C} | {B, C, D} | {C, D} | {D, G} | {E, G} | {G, F} | {G} |  |  |
| Processed | T | T | T | T | T | T | T | F | F |

Then we choose node I, we add in(I), out(F), out(G), out(I). (A, B) is covered by B. (F, I) is covered by G. (G, I) is covered by G.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in | {A} | {B} | {C} | {D} | {E, D} | {D, F} | {G} | {G} | {G, I} |
| out | {A, B, C} | {B, C, D} | {C, D} | {D, G} | {E, G} | {G, F} | {G} |  | {I} |
| Processed | T | T | T | T | T | T | T | F | T |

Then we choose node H, we add in(H), out(G), out(H). (G, H) is covered by G.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I |
| in | {A} | {B} | {C} | {D} | {E, D} | {D, F} | {G} | {H, G} | {G, I} |
| out | {A, B, C} | {B, C, D} | {C, D} | {D, G} | {E, G} | {G, F} | {G} | {H} | {I} |
| Processed | T | T | T | T | T | T | T | T | T |

All nodes are processed. Done.