**CS 48600-01 Analysis of Algorithms (3 cr.)**

**Assignment As 07\_(Final Exam)**

**Due on 12:00pm, Tuesday December 17, 2019 [in my office ET 125L]**

**This assignment is 70 points. Each of the problems is 10 points. Problem 8 and 9 are 30 points bonuses. Answer must be in a neat and legible handwriting or print out. No late turn-in is accepted. Do not send your solutions to me by email unless you are permitted.**

**Try your best!**

1. Given the following maze on 20x20 grids. An agent A search for goal G.

**Construct an equivalent graph for the given maze.** In terms of reachability, T can be reached by U or vice versa by a distance 1. B can reach A and vice versa by a distance 10. L can be reached and vice versa by J by a distance 16, etc. Thus, thus this equivalent graph is a weighted undirected graph.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **g** |
| **Y** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **Z** |
| **C** |  |  | **D**  | **X** |  |  |  |  |  |  |  | **H** |  |  |  | **O** |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | **Z** |  |  |  | **h** |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **B** |  |  |  |  |  |  |  | **E** |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **F** |  |  |  |  |  |  |  |  |  | **T** |  | **V** |
|  |  |  |  |  |  |  |  |  | **Q** |  |  |  |  |  |  |  | **U** | **t** |  |
|  |  |  |  |  |  |  |  | **I** |  |  |  | **N** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  | **J** | **n** |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **R** |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | **M** |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | **P** |  |  |  |  |  |  |
| **A** |  |  | **K** |  | **L** |  |  | **Y** |  |  |  |  |  |  |  | **S** |  | **W** | **G** |

19

2

20

20

1

20

25

1

10

22

3

10

36

171

3

3

16

18

10

2

4

5

5

21

16

16

9

4

4

13

1111

5

4

4

44

9

1

30

8

40

16

9

10

1. From the obtained graph for problem 1 for representing the given maze, construct

**(a) a weighted adjacency list and**

**(b) a weighted adjacency matrix**

for the obtained graph of the given maze.

Solution:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | $$\rightarrow $$ | $$\rightarrow $$ | $$\rightarrow $$ | $$\rightarrow $$ | $$\rightarrow $$ | $$\rightarrow $$ | $$\rightarrow $$ |
| A | B, 10 | K, 3 |  |  |  |  |  |
| B | A, 10 | C, 36 |  |  |  |  |  |
| C | B, 36 | D, 3 | Z, 20 |  |  |  |  |
| D | C, 3 | K, 16  | d, 1 |  |  |  |  |
| E | H, 10 | d, 22 | n, 3 |  |  |  |  |
| F | d, 10 |  |  |  |  |  |  |
| G | U, 10 |  |  |  |  |  |  |
| H | E, 10 | O, 4 | e, 13 |  |  |  |  |
| J | K, 18 | L, 16 | M, 5 | n, 2 |  |  |  |
| K | A, 3 | D, 16 | J, 18 | d, 17 |  |  |  |
| L | J, 16 | S, 11 | m, 4 |  |  |  |  |
| M | J, 5 | N, 21 | n, 5 |  |  |  |  |
| N | M, 21 | e, 16 | n, 4 |  |  |  |  |
| O | H, 4 | P, 30 | R, 40 | T, 8 |  |  |  |
| P | O, 30 | R, 44 | S, 4 | m, 5 |  |  |  |
| R | P, 44 | O, 40 |  |  |  |  |  |
| S | L, 11 | P, 4 | U, 9 |  |  |  |  |
| T | O, 8 | U, 1 | V, 16 |  |  |  |  |
| U | G, 10 | S, 9 | T, 1 | W, 9 |  |  |  |
| V | T, 16 |  |  |  |  |  |  |
| W | U, 9 |  |  |  |  |  |  |
| X | Y, 1 | g, 19 |  |  |  |  |  |
| Y | X, 1 | Z, 20 | g, 20 |  |  |  |  |
| Z | C, 20 | Y, 20 | g, 2 |  |  |  |  |
| d | D, 1 | E, 22 | F 10 | K, 17 |  |  |  |
| e | H, 13 | N, 16 | h, 4 |  |  |  |  |
| g | X, 19 | Y, 20 | Z, 2 |  |  |  |  |
| h | e, 4 |  |  |  |  |  |  |
| m | L, 4 | P, 5 |  |  |  |  |  |
| n | E, 3 | M, 5 | N, 4 | J, 2 |  |  |  |
|  |  |  |  |  |  |  |  |

1. Weighted adjacency list (30 vertices) of the graph corresponding to the given maze.
2. Weighted adjacency matrix (30 vertices) of the graph corresponding to the given maze.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | J | K | L | M | N | O | P | R | S | T | U | V | W | X | Y | Z | d | e | g | h | m | n |
| A |  | 10 |  |  |  |  |  |  |  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| B | 10 |  | 36 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| C |  | 36 |  | 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 20 |  |  |  |  |  |  |
| D |  |  | 3 |  |  |  |  |  |  | 16 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |
| E |  |  |  |  |  |  |  | 10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 22 |  |  |  |  | 3 |
| F |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 10 |  |  |  |  |  |
| G |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 10 |  |  |  |  |  |  |  |  |  |  |  |
| H |  |  |  |  | 10 |  |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  |  |  |  |  |  | 13 |  |  |  |  |
| J |  |  |  |  |  |  |  |  |  | 18 | 16 | 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 2 |
| K | 3 |  |  | 16 |  |  |  |  | 18 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 17 |  |  |  |  |  |
| L |  |  |  |  |  |  |  |  | 16 |  |  |  |  |  |  |  | 11 |  |  |  |  |  |  |  |  |  |  |  | 4 |  |
| M |  |  |  |  |  |  |  |  | 5 |  |  |  | 21 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5 |
| N |  |  |  |  |  |  |  |  |  |  |  | 21 |  |  |  |  |  |  |  |  |  |  |  |  |  | 16 |  |  |  | 4 |
| O |  |  |  |  |  |  |  | 4 |  |  |  |  |  |  | 30 | 40 |  | 8 |  |  |  |  |  |  |  |  |  |  |  |  |
| P |  |  |  |  |  |  |  |  |  |  |  |  |  | 30 |  | 44 | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| R |  |  |  |  |  |  |  |  |  |  |  |  |  | 40 | 44 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S |  |  |  |  |  |  |  |  |  |  | 11 |  |  |  | 4 |  |  |  | 9 |  |  |  |  |  |  |  |  |  |  |  |
| T |  |  |  |  |  |  |  |  |  |  |  |  |  | 8 |  |  |  |  | 1 | 16 |  |  |  |  |  |  |  |  |  |  |
| U |  |  |  |  |  |  | 10 |  |  |  |  |  |  |  |  |  | 9 | 1 |  |  | 9 |  |  |  |  |  |  |  |  |  |
| V |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 16 |  |  |  |  |  |  |  |  |  |  |  |  |
| W |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 9 |  |  |  |  |  |  |  |  |  |  |  |
| X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  |  |  | 19 |  |  |  |
| Y |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |  | 20 |  |  | 20 |  |  |  |
| Z |  |  | 20 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 20 |  |  |  | 2 |  |  |  |
| d |  |  |  | 1 | 22 | 10 |  |  |  | 17 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| e |  |  |  |  |  |  |  | 13 |  |  |  |  | 16 |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |
| g |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 19 | 20 | 2 |  |  |  |  |  |  |
| h |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  |  |
| m |  |  |  |  |  |  |  |  |  |  | 4 |  |  |  | 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| n |  |  |  |  | 3 |  |  |  | 2 |  |  | 5 | 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | A | B | C | D | E | F | G | H | J | K | L | M | N | O | P | R | S | T | U | V | W | X | Y | Z | d | e | g | h | m | n |

1. Traversing the graph obtained in problem 1, based on its **weighted adjacency list** representation obtained in problem 2(a), construct its **depth-first search tree** forest **starting from the vertex A.** In your obtained DFS tree forest, show the **tree edges** (indicated as solid line) and **back edges** (indicated as dotted line) for your trees. **Traversal’s stack** contains symbols (such as Vi, j , the first subscript number indicates the order in which a vertex V was first visited, say at i, (pushed onto the stack, V), where 0 < i $\leq $ n; the second one indicates the order in which it became a dead-end, say at j (popped off the stack V), where 0 < j < n. n is the total number of vertices for the given graph. For simplicity sake, please **use two time stamps**: one is 0 < i $\leq $ n, the order for **pushing a vertex onto the stack** counting from 1 through n. The other one is 0 < i $\leq $ n, the order for **popping off a vertex from the stack** counting from 1 through n. For this problem, you need to give your:

**(a) traversal’s stack with time-stamp**, and

**(b) the corresponding depth-first search (DFS) tree forest, with indications of tree edges and back edges.**

**(c) The DFS yields orderings of vertices; what are they?**

**(d)** **Is this graph connected?**

**(e) Is this graph acyclic?**

**(f) Does the graph have articulation points?** and

**(g) How do you call this graph?**

**(h) What is the topological sort ordering for the graph?**

**(i) What are the time efficiency and space efficiency of the DFS?**

**Solutions**

**(a) Based less weight first, stack with time-stamp for DFS traversal on the graph.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | **L21,4** |  |  |  |  |  |  |
|  |  |  | **m20,5** | **R22,6** |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | **P19, 7** |  |  |  |  |  |  |
|  |  | **W17, 3** | **S18, 8** |  | **G23,9** |  |  |  |  |
|  |  | **U16, 10** |  |  |  | **V24,11** |  |  |  |
|  |  | **T15, 12** |  |  |  |  |  |  |  |
|  |  | **O14,13** |  |  |  |  |  |  |  |
|  | **h12,2** | **H13,14** |  |  |  |  |  |  |  |
|  | **e11, 15** |  |  |  |  |  |  |  |  |
|  | **N10, 16** |  |  |  |  |  |  |  |  |
|  | **M9, 17** |  |  |  |  |  | **Y29,22** |  |  |
|  | **J8, 18** |  |  |  |  |  | **X28,23** |  |  |
|  | **n7, 19** |  |  |  |  |  | **g27,24** |  |  |
| **F5, 1**  | **E6, 20** |  |  |  |  |  | **Z26,25** | **B30,26** |  |
| **d4, 21**  |  |  |  |  |  |  | **C25,27** |  |  |
| **D3, 28**  |  |  |  |  |  |  |  |  |  |
| **K2, 29** |  |  |  |  |  |  |  |  |  |
| **A1, 30** |  |  |  |  |  |  |  |  |  |

1. **The DFS yields two orderings of vertices: The order in which the vertices are reached for the first time *(pushed onto the stack*) [when it is discovered] … listing them out. (See page 8)**

**The order in which the vertices become dead ends *(popped off the stack)* [when it is finished]. (See page 8)**

**The DFS yields orderings of vertices**

|  |
| --- |
| The below two lines are the popping-off ordering. |
| F5, 1 | h12, 2 | W17, 3 | L21, 4 | m20, 5 | R22, 6 | P19, 7 | S18, 8 | G23, 9 | U16, 10 | V24, 11 | T15, 12 | O14, 13 | H13, 14 | e11, 15 |
| N10, 16 | M9, 17 | J8, 18 | n7, 19 | E6, 20 | d4, 21 | Y29, 22 | X28, 23 | g27, 24 | Z26, 25 | B30, 26 | C25, 27 | D3, 28  | K2, 29 | A1, 30 |
| The below is the push-onto ordering. |
| A1, 30 | K2, 29 | D3, 2 | d4, 21 | F5, 1 | E6, 20 | n7, 19 | J8, 18 | M9, 17 | N10, 16 | e11, 15 | h12, 2 | H13, 14 | O14, 13 | T15, 12 |
| U16, 10 | W17, 3 | S18, 8 | P19, 7 | m20, 5 | L21, 4 | R22, 6 | G23, 9 | V24, 11 | C25, 27 | Z26, 25 | g27, 24 | X28, 23 | Y29, 22 | B30, 26 |

**(b) As shown above, the corresponding depth-first search (DFS) tree forest, with indications of tree edges (solid lines) and back edges (dotted lines).**

3

10

16

3

1

17

36

20

22

10

2

3 **F5, 1**

18

19

10

2

 5

20

1

4

5

20

21

16

13

4

4

16

8

16

1

10

40

9**F5, 1**

30

9**F5, 1**

4

5

11**5, 1**

44

4

**(b) As shown above, the corresponding depth-first search (DFS) tree forest, with indications of tree edges (solid lines) and back edges (dotted lines).**

**(d) This graph is connected.**

**(e) The graph is not acyclic. The DFS tree forest has back edge, which means there is cycle.**

**(f) An articulation point of the graph is a vertex whose removal disconnects the graph. So as a bridge is an edge of the graph whose removal disconnects the graph.**

**The articulation points are: d, e, U, T, C and Z.**

**The bridges are: (d F), (e h), (U W), (U G), (T, V) and (C Z)**

**(g) This graph is called weighted, connected, undirected graph.**

**(h) Since the graph is not acyclic, there is no topological sort ordering for the graph; namely it does not have the linearization form. (A topological sort of a dag G = (V, E) is a linear ordering of all its vertices such that if G contains an edge (u, v), then u appears before v in the ordering. (If the graph contains a cycle, then no linear ordering is possible.)**

**(i) The time (or space) efficiency of the DFS is Θ(|V| + |E|) if the adjacency list representation is used, comparing with Θ(|V|2 ) for the adjacency matrix representation.**

1. Traversing the graph constructed in problem 1, based on its **weighted adjacency list** representation obtained in problem 2(a), construct its **breath-first search (BFS) tree** forest **starting from the vertex A**. For this, you need to use a **queue** (note the *different from DFS*) to trace the operation of breadth-first search, indicating the order in which the vertices {…, V’, …, V”, … } were visited, i.e. **added to** (or **removed from) the queue** {…, $V\_{i}^{"}$, $V\_{i+1}^{'}$, … }. The order in which vertices are added to the queue (i.e., enqueue operation) is the same order in which they are removed from it (i.e., dequeue operation). Indicate the **tree edges** (indicated as solid line) and **cross-edges** (indicated as dotted line) for your trees. For this problem, you need to give your:

**(a) traversal’s queue with time-stamp indicating the order in which the vertices were visited**, and

**(b) the corresponding breadth-first search (BFS) tree forest, with indication of tree edges and cross edges.** (Back edges and Forward edges)

**(c) The BFS yields an ordering of vertices; what is it?**

**(d)** **Is this graph connected?**

**(e) Is this graph acyclic?**

**(f) Computes the *shortest* *distance (smallest number of edges) from* A to each reachable vertex.**

**(g) What is the topological sort ordering for the graph?**

**(h) What are the time efficiency and space efficiency of the BFS?**

Solution

19

2

20

20

1

20

25

1

10

22

3

10

36

171

3

3

16

18

10

2

4

5

5

21

16

16

9

4

4

13

11

5

4

4

44

9

1

30

8

40

16

9

10

1. **Traversal’s queue with time-stamp indicating the order in which the vertices were visited**.

Based on light weight first, the historical priority queue Q will be as follows:

A

K B

B D d J

D d J C

d J C

J C F E

C F E n M L

F E n M L Z

E n M L Z

n M L Z H

M L Z H N

L Z H N

Z H N m S

H N m S g Y

N m S g Y O e

m S g Y O e

S g Y O e P

g Y O e P U

Y O e P U X

O e P U X

e P U X T R

P U X T R h

U X T R h

X T R h W G

T R h W G

R h W G V

h W G V

W G V

G V

V

NIL

The ordering of removal from Q is:

A K B D d J C F E n M L Z H N m S g Y O e P U X T R h W G V

**(b) the corresponding breadth-first search (BFS) tree forest, with indication of tree edges and cross edges.** (Back edges and Forward edges)

3

3

18

16

36

17

1

3

20

22

11

5

2

16

5

3

11

20

2

4

21

10

4

20

16

13

5

9

1

19

4

4

30

4

40

44

8

1

9

10

16

**(c)The BFS yields an ordering of vertices: The order in which the vertices are enqueued for the first time *(added to the queue*) which is the same as the order in which the vertices are dequeued (removed from the queue). ….**

Based on light weight first, the historical priority queue Q will be as follows:

The ordering of removal from Q is:

A K B D d J C F E n M L Z H N m S g Y O e P U X T R h W G V

**(d) This graph is connected**

**(e) The graph is not acyclic. The BFS tree forest has cross edge, which means there is cycle.**

**(f) It computes the *distance (smallest number of edges) from* A to each reachable vertex v, such as G.**

* + **For any vertex v reachable from A, *the simple path in the breadth-first tree from* A *to*  v *corresponds to a shortest path from* A *to* v *in the graph* [ that is, a path containing the smallest number of edges].**

**(A K J L S U G) which has six edges**

* + **If the graph is weighted undirected graph, then the shortest path from A to v could be defined in terms of minimum sum the weights of the edges joining from A to v, such as G.**

**(A 3 K 18 J 16 L 11 S 9 U 10 G) which has six edges with total weight = 67.**

**(g) This graph is called weighted, connected, undirected graph.**

**(h) Since the graph is not acyclic, there is no topological sort ordering for the graph; namely it does not have the linearization form. (A topological sort of a dag G = (V, E) is a linear ordering of all its vertices such that if G contains an edge (u, v), then u appears before v in the ordering. (If the graph contains a cycle, then no linear ordering is possible.)**

**(i) The time (or space) efficiency of the BFS is Θ(|V| + |E|) if the adjacency list representation is used, comparing with Θ(|V|2 ) for the adjacency matrix representation.**