

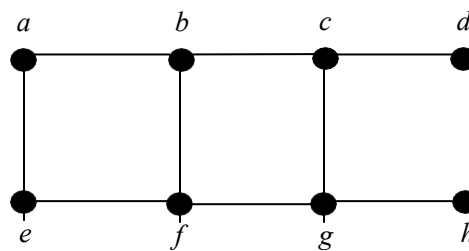
INT102 Algorithmic Foundations
Problem Session 3, Week 6

Location: SC176

Suggested Solutions

Question 1

Consider the following graph G .



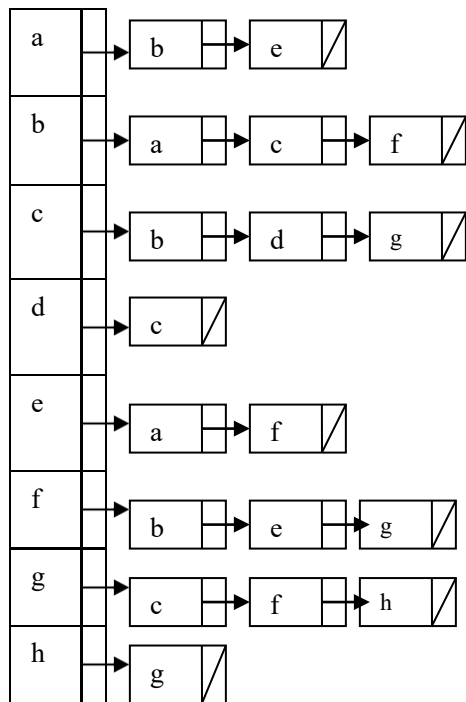
1. Give the adjacency matrix and adjacency list of the graph G . (4 marks)

Answer:

adjacency matrix (2 marks)

	a	b	c	d	e	f	g	h
a	0	1	0	0	1	0	0	0
b	1	0	1	0	0	1	0	0
c	0	1	0	1	0	0	1	0
d	0	0	1	0	0	0	0	0
e	1	0	0	0	0	1	0	0
f	0	1	0	0	1	0	1	0
g	0	0	1	0	0	1	0	1
h	0	0	0	0	0	0	1	0

adjacency list (2 marks)



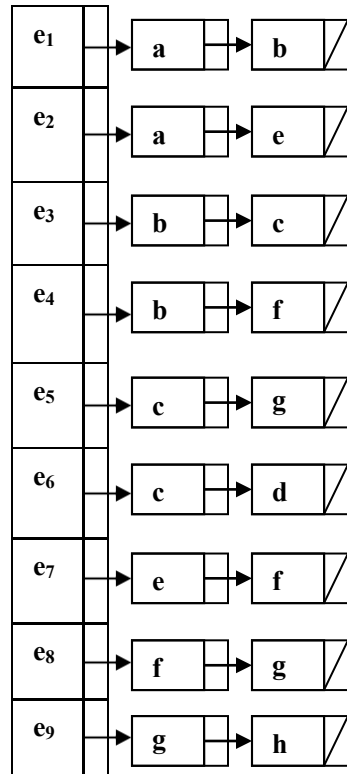
2. Give the incidence matrix and incidence list of the graph G . (4 marks)

Answer: Let $e_1=(a, b)$, $e_2=(a, e)$, $e_3=(b, c)$, $e_4=(b, f)$, $e_5=(c, g)$, $e_6=(c, d)$, $e_7=(e, f)$, $e_8=(f, g)$, $e_9=(g, h)$

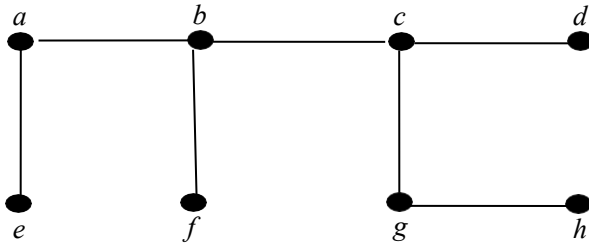
incidence matrix (2 marks)

	a	b	c	d	e	f	g	h
e_1	1	1	0	0	0	0	0	0
e_2	1	0	0	0	1	0	0	0
e_3	0	1	1	0	0	0	0	0
e_4	0	1	0	0	0	1	0	0
e_5	0	0	1	0	0	0	1	0
e_6	0	0	1	1	0	0	0	0
e_7	0	0	0	0	1	1	0	0
e_8	0	0	0	0	0	1	1	0
e_9	0	0	0	0	0	0	1	1

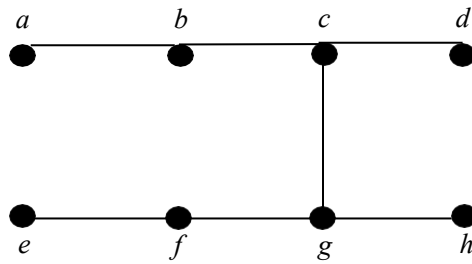
incidence list (2 marks)



3. Starting at the vertex *a* and resolving ties by the vertex alphabetical order traverse the graph by breadth-first-search (BFS) and construct the corresponding BFS tree. (5 marks)



4. Starting at the vertex *a* and resolving ties by the vertex alphabetical order traverse the graph by depth-first-search (DFS) and construct the corresponding BFS tree. (5 marks)



Question 2

Consider the following recursive function

$$f(n) = \begin{cases} 1 & \text{if } 0 \leq n \leq 3 \\ f(n-1) + f(n-2) + f(n-3) + f(n-4) & \text{if } n > 4 \end{cases}$$

1. Write a recursive (top-down) algorithm to compute it. (5 marks)

Answer:

Procedure F(n)

if $n=0$ or $n=1$ or $n=2$ or $n=3$ then

return 1

else

return $F(n-1) + F(n-2) + F(n-3) + F(n-4)$

2. What is the complexity of your algorithm (in big-O notation)? (5 marks)

Answer:

The time complexity of this algorithm is exponential, specifically $O(4^n)$, where n is the input value. This is because for each value of n greater than 4, the function makes four recursive calls to values of $n-1$, $n-2$, $n-3$, and $n-4$, and each of those calls makes four more recursive calls, and so on. As a result, the number of function calls grows exponentially with the input size, leading to an exponential time complexity.

3. Design and write the pseudo code of a faster nonrecursive (bottom-up) algorithm using the concept of dynamic programming. (5marks)

Procedure F(n)

Set $A[0] = A[1] = A[2] = A[3] = 1$

for $i = 4$ to n do

$A[i] = A[i-1] + A[i-2] + A[i-3] + A[i-4]$

return $A[n]$

4. What is the time complexity of the faster algorithm (in big-O notation)? (5 marks)

$O(n)$

Question 3

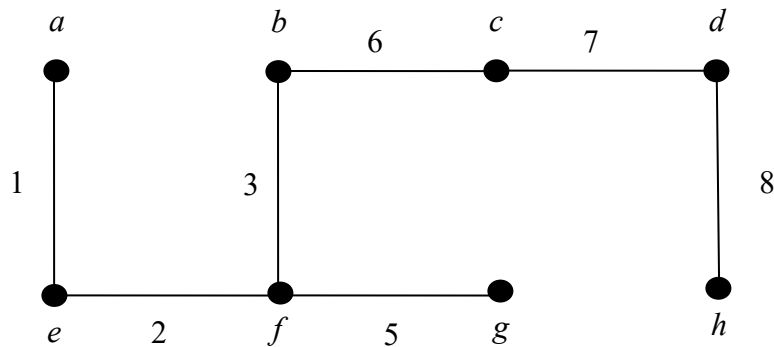
1. By Prim's algorithm, the order in which the vertices are selected is as follows:

a, e, f, b, g, c, d, h

(3 marks for the priority queue)

Order Selected	a(0,-)	b(-,∞)	c(-,∞)	d(-,∞)	e(-,∞)	f(-,∞)	g(-,∞)	h(-,∞)
a(0,-)		b(a,4)	c(-,∞)	d(-,∞)	e(a,1)	f(-,∞)	g(-,∞)	h(-,∞)
e(a,1)		b(a,4)	c(-,∞)	d(-,∞)		f(-,2)	g(-,∞)	h(-,∞)
f(e,2)		b(f,3)	c(f,10)	d(-,∞)			g(f,5)	h(-,∞)
b(f,3)			c(b,6)	d(-,∞)			g(f,5)	h(-,∞)
g(f,5)			c(b,6)	d(g,14)				h(g,8)
c(b,6)				d(c,7)				h(g,8)
d(c,7)								h(g,8)
h(g,8)								

The MST is as follows: (3 marks)

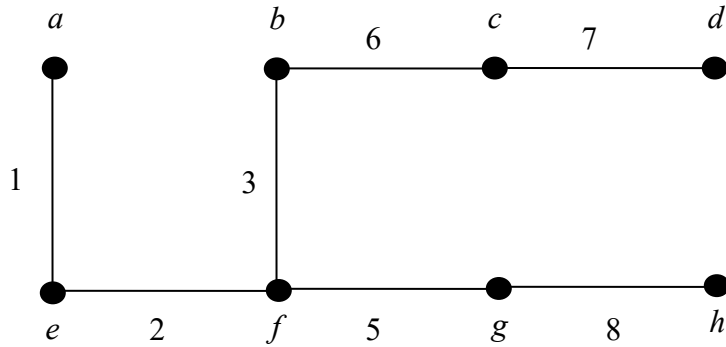


The MST drawn is not unique (1 mark)

2. Sorting edges according to their weights in increasing order:

(3 marks for the order list, 3 marks for the MST)

(a, e)	1
(e, f)	2
(f, b)	3
(a, b)	4
(f, g)	5
(b, c)	6
(c, d)	7
(g, h)	8
(d, h)	8
(c, f)	10
(c, g)	12
(d, g)	14



The MST is not unique. (1 mark)

3.

Selected order of edges: (a, e), (e, f), (a, b), (f, g), (b, c), (g, h), (c, d)

(3 mark for the priority queue)

Order Selected	a(0,-)	b(-,∞)	c(-,∞)	d(-,∞)	e(-,∞)	f(-,∞)	g(-,∞)	h(-,∞)
a(0,-)		b(a,4)	c(-,∞)	d(-,∞)	e(a,1)	f(-,∞)	g(-,∞)	h(-,∞)
e(a,1)		b(a,4)	c(-,∞)	d(-,∞)		f(e,3)	g(-,∞)	h(-,∞)
f(e,3)		b(a,4)	c(f,13)	d(-,∞)			g(f,8)	h(-,∞)
b(a,4)			c(b,10)	d(-,∞)			g(f,8)	h(-,∞)
g(f,8)			c(b,10)	d(g,22)				h(g,16)
c(b,10)				d(c,17)				h(g,16)
h(g,16)				d(c,17)				
d(c,17)								

(3 mark for the shortest paths)

