Chapter 8
Trees

MAD101

Vo Tran Duy

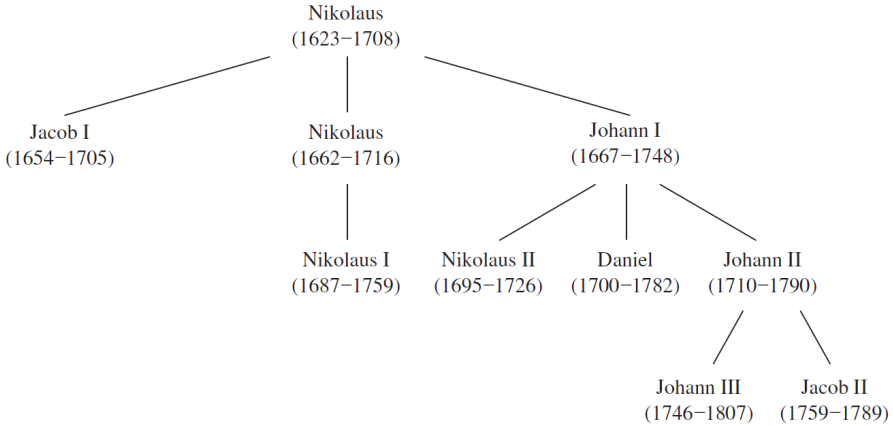
duyvt15@fe.edu.vn

Table of Contents

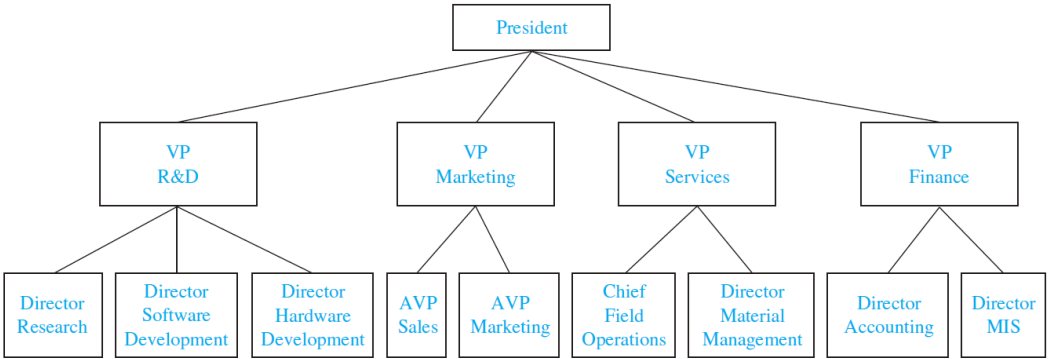
1 Introduction and Definitions

The Bernoulli family of mathematicians

1 Introduction and Definitions



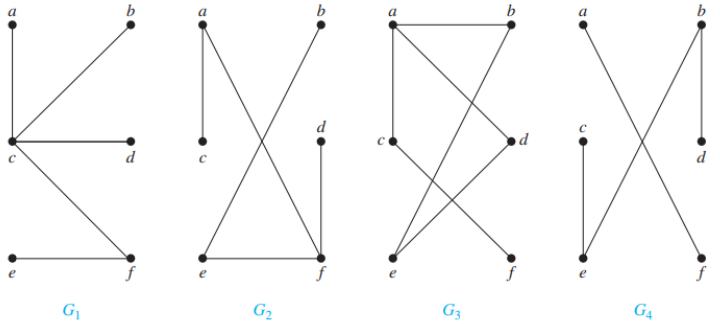
An organizational tree for a computer company



Definition

1 Introduction and Definitions

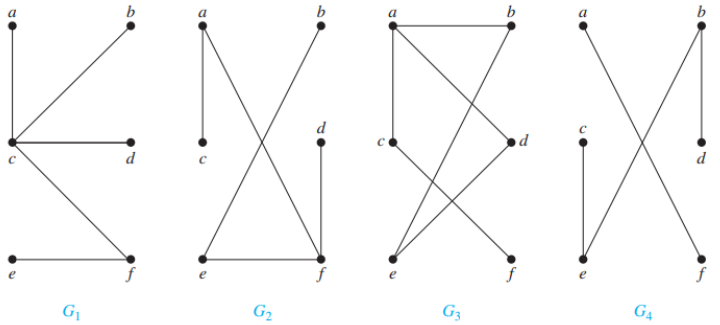
A **tree** is a connected undirected graph with no simple circuits.



Definition

1 Introduction and Definitions

A **tree** is a connected undirected graph with no simple circuits.



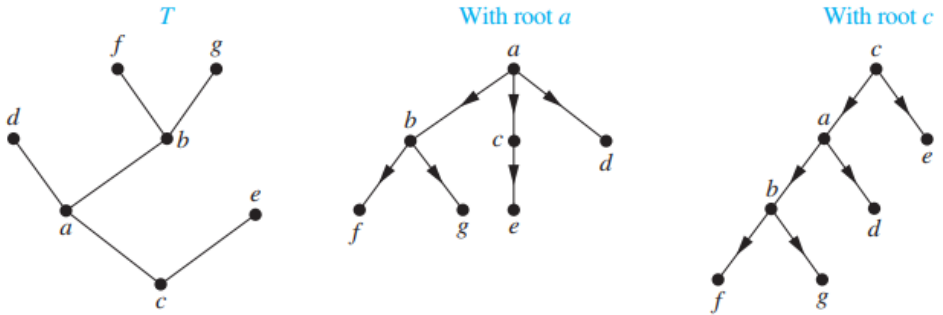
G_1, G_2 are trees and G_3, G_4 are not.

Theorem

An **undirected** graph is a tree if and only if there is a unique simple path between any two of its vertices.

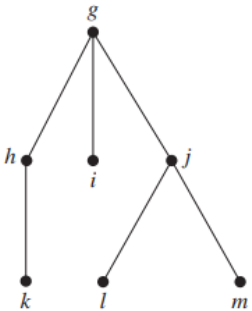
Definition

A **rooted tree** is a tree in which one vertex has been designated as the root and every edge is directed away from the root.



Some common concepts

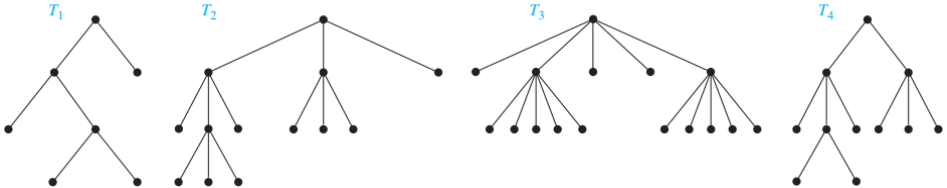
1 Introduction and Definitions



- g is **root** of tree
- j is **parent** of m and l
- m or l is a **child** of j
- m, l are **siblings**
- g is **ancestor** of m and m is **descendant** of g
- i, k, l, m are leaves: nodes without children.
- g, h, j are Internal nodes: nodes have children.

Definition

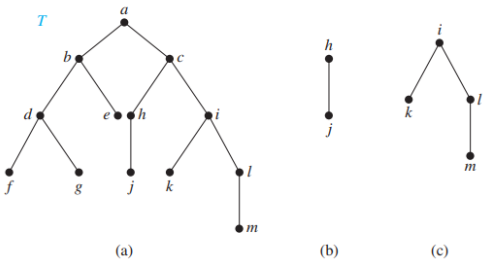
- A rooted tree is called an **m-ary** tree if every internal vertex has no more than m children.
- The tree is called a **full m-ary** tree if every internal vertex has exactly m children.
- An m-ary tree with $m = 2$ is called a **binary tree**.



Definition

1 Introduction and Definitions

An **ordered rooted tree** is a rooted tree where the children of each internal vertex are ordered (which is usually represented as in the graph).



f is the left child of d and g is the right child of d
(b): the left subtree of c and (c): the right subtree of c.

Properties of trees

1. A tree with n vertices has $n - 1$ edges.
2. For any **m-ary** tree
$$n = i + \ell$$
3. For any **full m-ary** tree
$$n = mi + 1 \quad \text{and} \quad \ell = (m - 1)i + 1$$

- n is the number of nodes,
- i is the number of internal nodes,
- ℓ is the number of leaves.

Properties of trees

1. A tree with n vertices has $n - 1$ edges.
2. For any **m-ary** tree
$$n = i + \ell$$
3. For any **full m-ary** tree
$$n = mi + 1 \quad \text{and} \quad \ell = (m - 1)i + 1$$

- n is the number of nodes,
- i is the number of internal nodes,
- ℓ is the number of leaves.

Example.

a. How many leaves are there in a full 5-ary tree with 56 nodes?

b. How many vertices are there in a full ternary (3-ary) tree with 27 leaves?

Properties of trees

1. A tree with n vertices has $n - 1$ edges.
2. For any **m-ary** tree
$$n = i + \ell$$
3. For any **full m-ary** tree
$$n = mi + 1 \quad \text{and} \quad \ell = (m - 1)i + 1$$

- n is the number of nodes,
- i is the number of internal nodes,
- ℓ is the number of leaves.

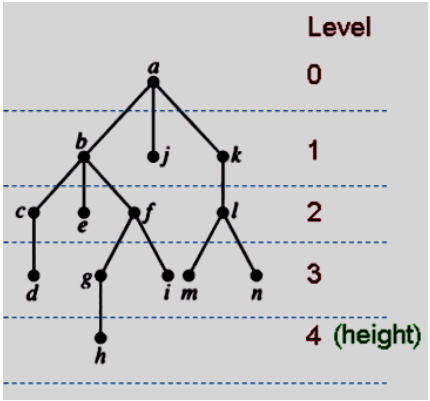
Example.

a. How many leaves are there in a full 5-ary tree with 56 nodes?

b. How many vertices are there in a full ternary (3-ary) tree with 27 leaves?

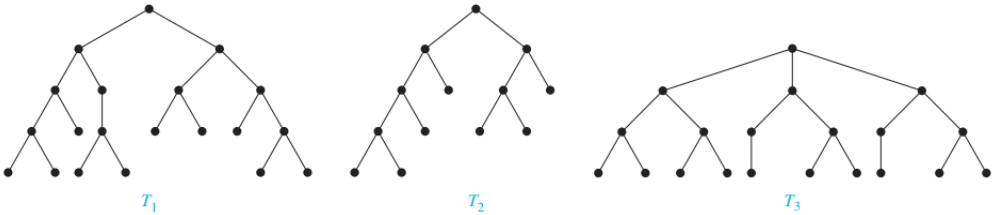
Solution. *a.* $\ell = 45$; *b.* $n = 40$.

Definition



- The **level** of a vertex v in a rooted tree is the length of the unique path from the root to this vertex.
- The **level of the root** is defined to be zero.
- The **height** of a rooted tree is the maximum of the levels of vertices.

Example.



1. There are 10 nodes leaves in T_1 and their level are 3&4.(h=4)
2. There are 7 nodes leaves in T_2 and their level are 2,3 & 4.(h=4)
3. There are 10 nodes leaves in T_3 and their level are 3.(h=3)

$\hookrightarrow$ A rooted m-ary tree of height h is **balanced** if all leaves are at levels h or $h-1$.
- In example above, T_1, T_3 are balanced and T_2 is not.

Theorem

There are at most m^h leaves in an m-ary tree of height h.

Corollary

If an m-ary tree of height h has ℓ leaves, then $h \geq \lceil \log_m \ell \rceil$. If the m-ary tree is full and balanced, then $h = \lceil \log_m \ell \rceil$.

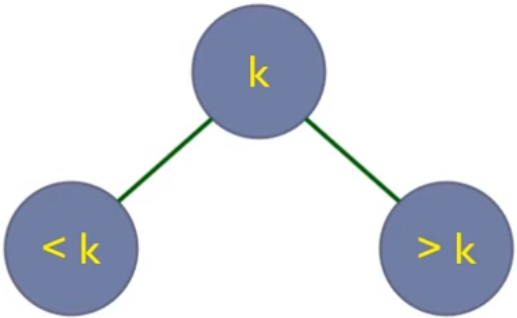
Table of Contents

2 Binary Search Tree

Definition

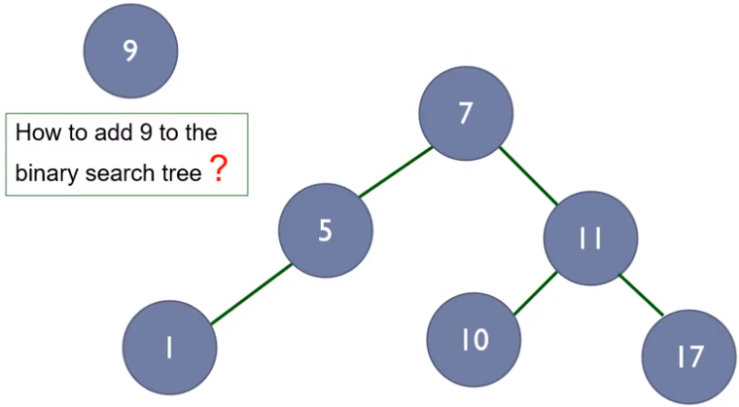
2 Binary Search Tree

- A binary search tree where each vertex is labeled with a key.
- The key of a vertex is both larger than the keys of all vertices in its left subtree and smaller than the keys of all vertices in its right subtree.



Example.

2 Binary Search Tree



How about $6, 13, 4$?

Example.

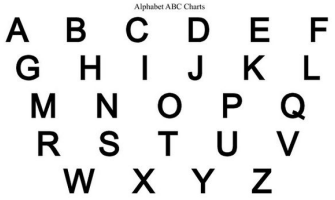
Construct the Binary Search Trees for $8, 3, 11, 15, 2, 4, 14$. How many comparisons are required to locate number 6 in the search tree?

Example.

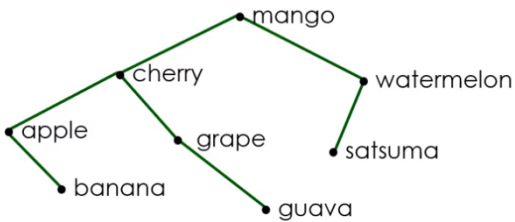
Construct the Binary Search Trees for $8, 3, 11, 15, 2, 4, 14$. How many comparisons are required to locate number 6 in the search tree? $\Longrightarrow$ 4 comparisons are required.

Alphabetical order

2 Binary Search Tree



Binary search tree for words: mango, cherry, grape, apple, guava, watermelon, satsuma, banana.



$a < b < c ...; ab < ac < ae ...; abc < abd < abe ...$

Algorithm: Locating an Item in or Adding an Item to a Binary Search Tree

2 Binary Search Tree

procedure *insertion*(T: binary search tree,
x: item)
v := root of T
{a vertex not present in T has the value *null* }
while $v \neq$ *null* and *label*(v) $\neq x$
  if $x <$ *label*(v) **then**
    if left child of $v \neq$ *null* **then** v := left child of v
    else add *new vertex* as a left child of v and set v := *null*
  else
    if right child of $v \neq$ *null* **then** v := right child of v
    else add *new vertex* as a right child of v and set v := *null*
if root of T = *null* **then** add a vertex v to the tree and label it with x
else if v is null or *label*(v) $\neq x$ **then** label *new vertex* with x and let v be this new vertex
return v {v = location of x}

Note: The algorithm returns the location of x or adds a new vertex with label x into binary search tree. Complexity $O(logn)$.

Table of Contents

3 Prefix code

Introduction

3 Prefix code

- Standard ASCII (*American Standard Code for Information Interchange*) character encoding: use 8 bits (1 byte) to store each character.
- For example, using the standard ASCII encoding, the 12-characters string "no one knows" requires $12 * 8 = 96$ bits total.

char	ASCII	bit pattern
n	110	01101110
o	111	01101111
e	101	01100101
k	107	01101011
w	119	01110111
s	115	01110011
space	32	00100000

Introduction

3 Prefix code

- A better code scheme

char	number	bit pattern
n	0	000
o	1	001
e	2	010
k	3	011
w	4	100
s	5	101
space	6	110

$12 * 3 = 36$ bits total.

- A variable-length encoding

char	bit pattern
n	01
o	11
e	000
k	001
w	1000
s	1001
space	101

Use 32 bits total.

Consider the codes

1. Use the following code

$a:10$	$e:01$	$t:0110$	$n:1$

Decode the message 01101

01101 mean "*ean*"

01101 mean "*tn*"

Hence, it is not a prefix code.

2. Use the following code

$a:10$	$e:01$	$t:001$	$n:11$

Decode the message 1101001

1101001 mean "net" only.

It is a prefix code.

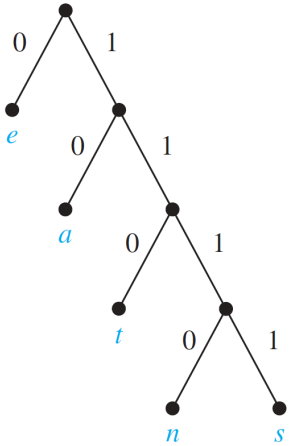
Definition

3 Prefix code

Prefix code is a type of codes which encodes the letters so that the bit string for a letter never occurs as the first part of the bit string for another letter.

A binary tree with a prefix code

3 Prefix code



s : 1111

a : 10

n : 1110

e : 0

So, "*sane*" will be store as 11111011100 $\rightarrow$ 11 bits.

Hence, the compression factor: $32/11 \sim 3$ compared to ASCII. Is it best? How to construct a prefix code that is using the fewest bits?

Huffman coding algorithm

3 Prefix code

- Find frequencies (probabilities) of each character in a text,
- Constructing a rooted binary tree representing prefix codes of character.
  Notes:
  - Sort by frequencies (probabilities) from small to large after each step,
  - The larger weight sub-tree is on the **left** of the binary tree.

Example. Use Huffman coding algorithm to encode the word “google”.

Solution.

3 Prefix code

- Counting frequencies of letter

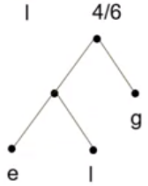
e	l	g	o
1/6	1/6	2/6	2/6

- $P(l) = P(e) = 1/6$

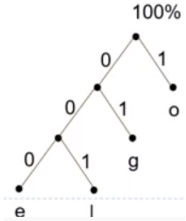
Example

3 Prefix code

- $P(l) + P(e) = 2/6 = P(g)$

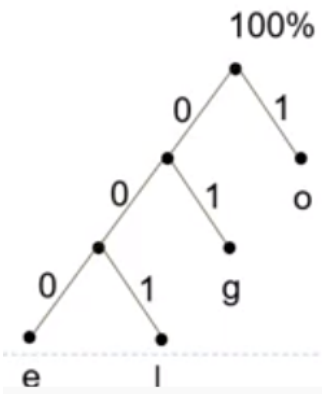


- Final character



Example.

3 Prefix code



- o : 1
- g : 01
- l : 001
- e : 000

Hence, "*google*" will be store as 011101001000 $\rightarrow$ 12 bits. Compare with ASCII, the copression factor is $48/12 \sim 4$.

Example.

3 Prefix code

Use Huffman coding algorithm to encode the text "*maximum*". What is the average number of bits?

Example.

3 Prefix code

Use Huffman coding algorithm to encode the text "*maximum*". What is the average number of bits?

Ans: We get:

$$a : 000 \qquad i : 001 \qquad u : 010$$
$$x : 011 \qquad m : 1$$

The average number of bits used to encode a symbol using this encoding is $\frac{15}{7} \approx 2.14$

Example

3 Prefix code

Use Huffman coding to encode the following symbols with the frequencies listed: A: 0.08, B: 0.10, C: 0.12, D: 0.15, E: 0.20, F: 0.35. What is the average number of bits used to encode a character?

Example

3 Prefix code

Use Huffman coding to encode the following symbols with the frequencies listed: A: 0.08, B: 0.10, C: 0.12, D: 0.15, E: 0.20, F: 0.35. What is the average number of bits used to encode a character?

Ans: The average number of bits used to encode a symbol using this encoding is

$$3 \times 0.08 + 3 \times 0.10 + 3 \times 0.12 + 3 \times 0.15 + 2 \times 0.20 + 2 \times 0.35 = 2.45$$

Huffman Coding Algorithm

3 Prefix code

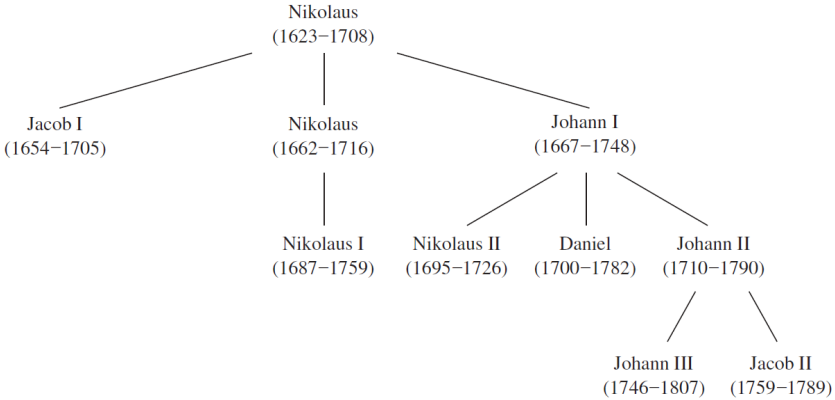
procedure *Huffman*(C: symbols a_i with frequencies w_i, $i = 1, \ldots, n$)
$F :=$ forest of n rooted trees, each consisting of the single vertex a_i and assigned weight w_i
while F is not a tree
  Replace the rooted trees T and T' of least weights from F with $w(T) \geq w(T')$ with a tree having a new root that has T as its left subtree and T' as its right subtree. Label the new edge to T with 0 and the new edge to T' with 1.
  Assign $w(T) + w(T')$ as the weight of the new tree.
{the Huffman coding for the symbol a_i is the concatenation of the labels of the edges in the unique path from the root to the vertex a_i}

Table of Contents

4 Tree Traversal

Introduction

4 Tree Traversal



Tree Traversal

4 Tree Traversal

Traversal a tree: A way to visit all vertices of the rooted tree.

Traversal Algorithms

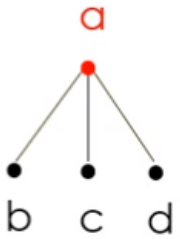
4 Tree Traversal

- At a time, a vertex is visited,
- Recursive algorithm,
- Bases on orders of tasks, traversals are classified into:
  1. Pre-order traversal. N L R
  2. In-order traversal. L N R
  3. Post-order traversal. L R N

  where,
  - N: root node
  - L: left subtree
  - R: right subtree

Example.

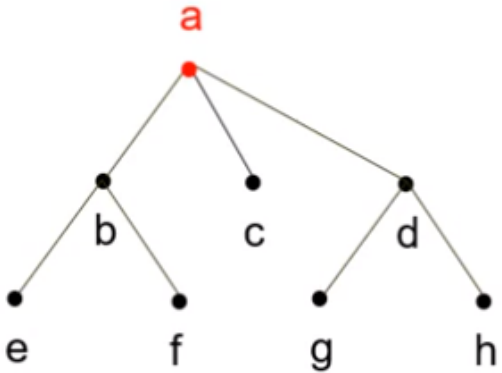
4 Tree Traversal



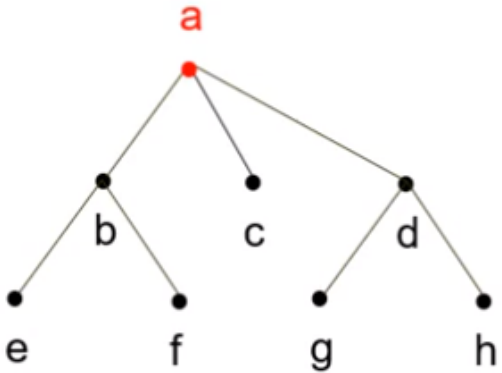
Pre-order	In-order	Post-order
N L R	L N R	L R N
a b c d	b a c d	b c d a

Pre-order traversal example.

4 Tree Traversal



Pre-order traversal example.



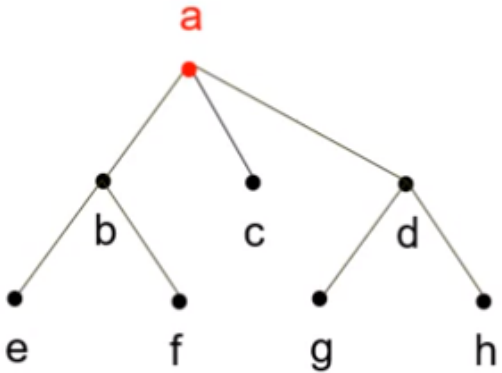
Pre-order traversal: a b e f c d g h

Pre-order traversal algorithm

procedure *preorder*(T: ordered rooted tree)
r := root of T
list r
for each child c of r from left to right
    $T(c)$:= subtree with c as its root
    preorder($T(c)$)

In-order traversal example

4 Tree Traversal



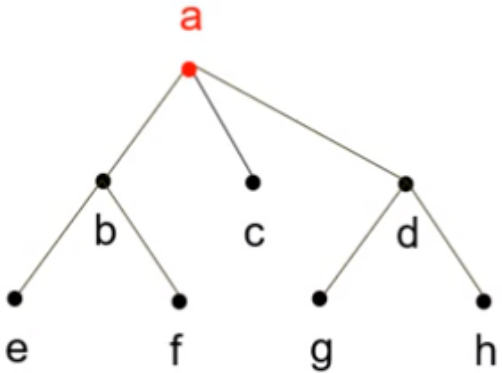
In-order traversal: e b f a c g d h

In-order traversal algorithm

procedure *inorder*(T: ordered rooted tree)
r := root of T
if r is a leaf **then** list r
else
  l := first child of r from left to right
  $T(l)$:= subtree with l as its root
  inorder($T(l)$)
  list r
  for each child c of r except for l from left to right
    $T(c)$:= subtree with c as its root
    inorder($T(c)$)

Post-order traversal example.

4 Tree Traversal



Post-order traversal: e f b c g h d a

Post-order traversal algorithm

procedure *postorder*(T: ordered rooted tree)
r := root of T
for each child c of r from left to right
    $T(c)$:= subtree with c as its root
    postorder($T(c)$)
list r

Exercise

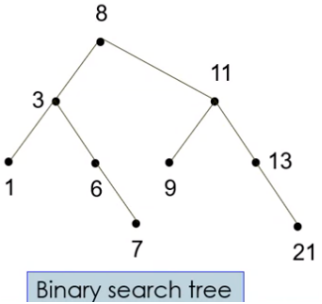
4 Tree Traversal

Construct a **binary search tree** for the numbers:
8, 11, 3, 6, 9, 1, 13, 7, 21
What are the order of numbers after applying:

- preorder traversal
- inorder traversal
- postorder traversal?

Solution.

4 Tree Traversal



1. Pre-order: 8 3 1 6 7 11 9 13 21
2. In-order: 1 3 6 7 8 9 11 13 21
3. Post-order: 1 7 6 3 9 21 13 11 8

In in-order, the sequences of numbers are sorted.

Table of Contents

5 I.,P.&P.fix No.

Representing the expression

5 I.,P.&P.fix No.

- $+$: addition
- $-$: subtraction
- $*$: multiplication
- $/$: division
- $\uparrow$: exponentiation

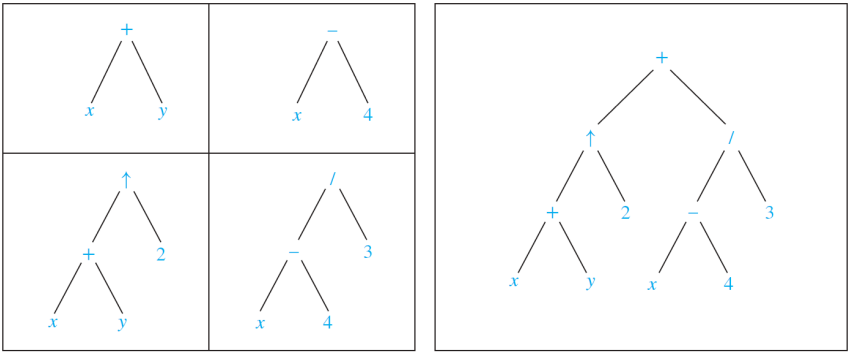
The expression can be represent by the binary trees with the rules:

- The internal vertices represent operations,
- The leaves represent the variables or numbers.

Example

5 I.,P.&P.fix No.

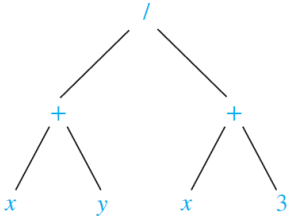
What is the ordered rooted tree that represents the expression $((x + y) \uparrow 2) + ((x - 4)/3)$?

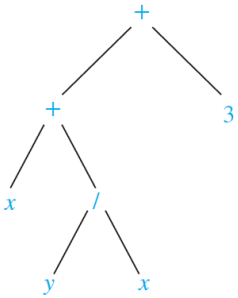


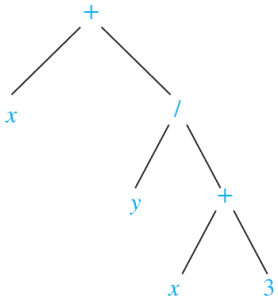
Example

5 I.,P.&P.fix No.

What is the ordered rooted tree that represents the expression $(x + y)/(x + 3)$, $(x + (y/x)) + 3$, and $x + (y/(x + 3))$?







Definition

5 I.,P.&P.fix No.

- We obtain **prefix form** of an expression when we traverse its rooted tree in preorder + x y
- We obtain **infix form** of an expression when we traverse its rooted tree in inorder x + y
- We obtain **postfix form** of an expression when we traverse its rooted tree in postorder x y +

Note: Prefix form is called **Polish notation** and Postfix form is called **reverse Polish notation**.

How to find prefix and postfix form from infix form?

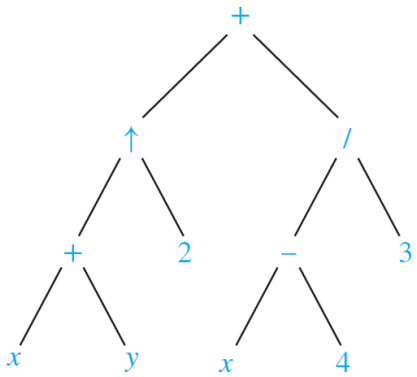
5 I.,P.&P.fix No.

1. Draw expression tree.
2. Using Preorder traverse → Prefix form. Using Postorder traverse → Postfix form.

Example

5 I.,P.&P.fix No.

What is the prefix (infix, postfix) form of the following tree?



- Infix form $((x + y) \uparrow 2) + ((x - 4)/3)$
- Prefix form $+ \uparrow + xy2/ - x43$
- Postfix form $xy + 2 \uparrow x4 - 3/+$

Evaluate a Prefix and Postfix form

5 I.,P.&P.fix No.

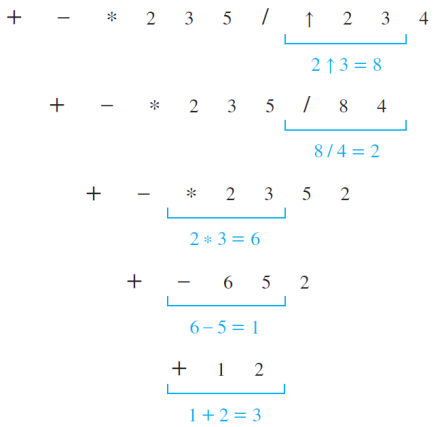
1. Postfix form, work from left to right.
2. Prefix form, work from right to left.

Examples

1. $+ - *235/\uparrow 234$
2. $723 * -4 \uparrow 93/+$

Solution.

5 I.,P.&P.fix No.



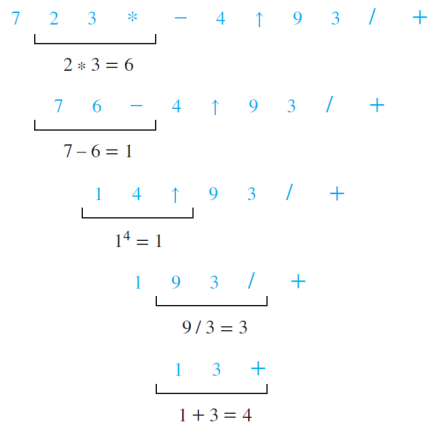
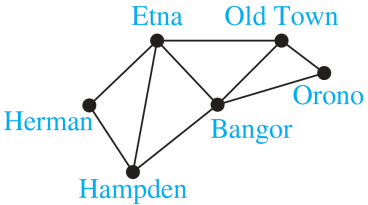


Table of Contents

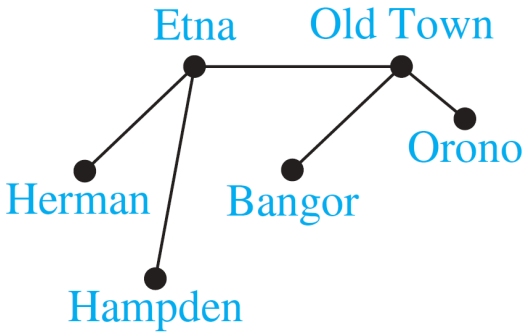
6 Spa.Trees

The system of roads in Maine (USA)



- The only way the roads can be kept open in the winter is by frequently plowing them,
- The highway department wants to plow the fewest roads so that there will always be cleared roads connecting any two towns,
- How can this be done?

At least five roads must be plowed to ensure that there is a path between any two towns



Definition

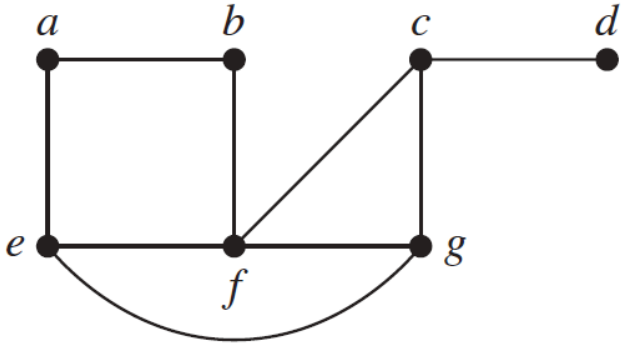
6 Spa.Trees

Let G be a simple graph. A **spanning tree** of G is a subgraph of G that is a tree containing every vertex of G.

Example

6 Spa.Trees

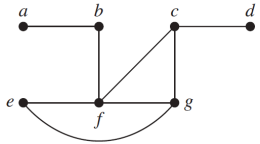
Find a spanning tree of this simple graph:



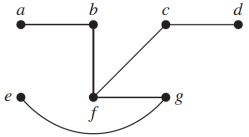
Solution.

6 Spa.Trees

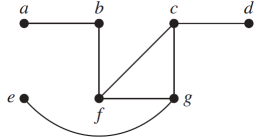
- Removed $\{a, e\}$



- Removed $\{c, g\}$



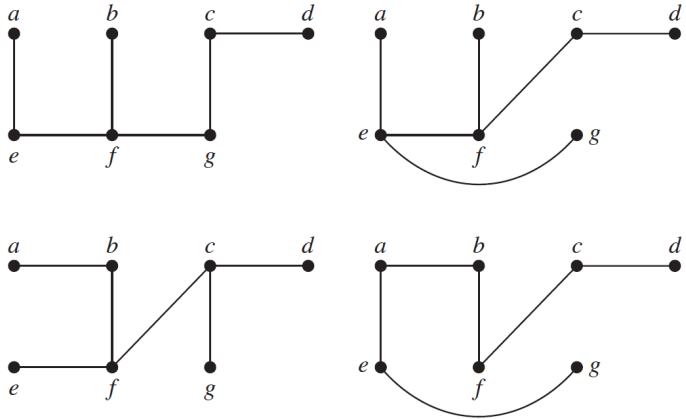
- Removed $\{e, f\}$



Solution.

6 Spa.Trees

The spanning tree above is not unique, there are four different spanning trees



Theorem

6 Spa.Trees

A simple graph is connected if and only if it has a spanning tree.

Depth-First Search (backtracking)

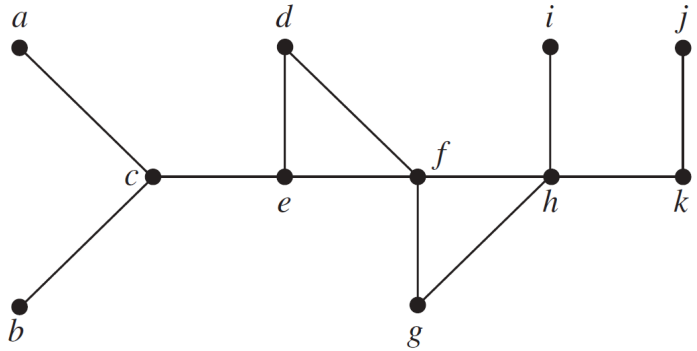
6 Spa.Trees

- Arbitrarily choose a vertex of the graph as the root. Form a path starting at this vertex by successively adding vertices and edges, where each new edge is incident with the last vertex in the path and a vertex not already in the path.
- Continue adding vertices and edges to this path as long as possible,
  1. If the path goes through all vertices of the graph, the tree consisting of this path is a spanning tree.
  2. If the path does not go through all vertices, move back to the next to last vertex in the path, forming new paths (adding new vertices and edges) that are as long as possible until no more edges can be added $\rightarrow$ spanning tree.

Example

6 Spa.Trees

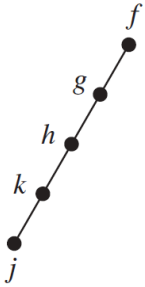
Find a spanning tree of this graph using Depth-First Search:



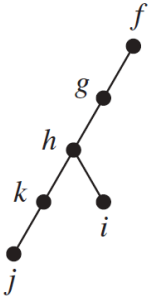
Solution.

6 Spa.Trees

(a)



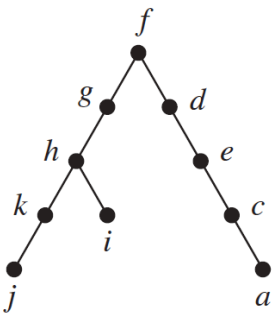
(b)



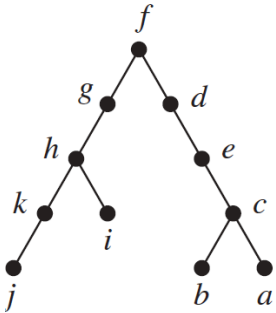
(c)

Solution.

6 Spa.Trees



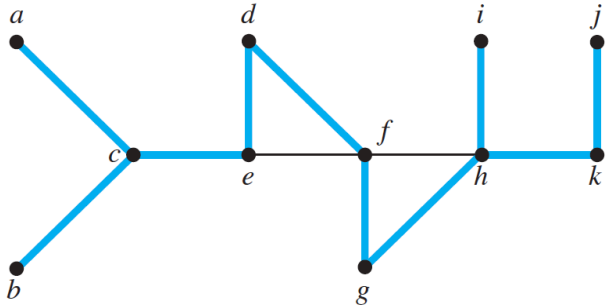
(d)



(e)

Tree edges and back edges

6 Spa.Trees

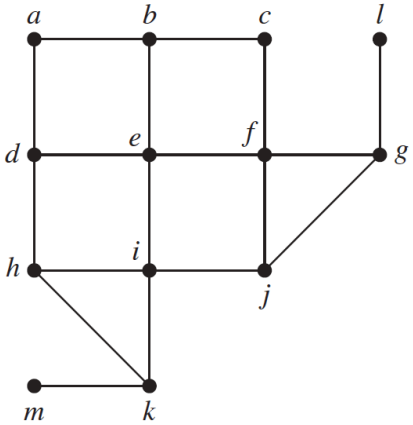


- Tree edges: highlight,
- Back edges: thinner black lines.

Example

6 Spa.Trees

Find a spanning tree of this graph using Depth-First Search:



Depth-First Search Algorithm

6 Spa.Trees

procedure *DFS*(G: connected graph with vertices $v_1, v_2, \ldots, v_n$)
$T :=$ tree consisting only of the vertex v_1
visit(v_1)

procedure *visit*(v: vertex of G)
for each vertex w adjacent to v and not yet in T
    add vertex w and edge $\{v, w\}$ to T
    visit(w)

Breadth-First Search

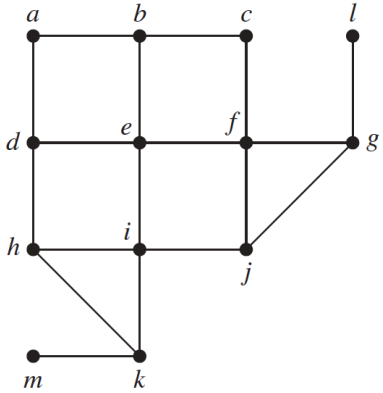
6 Spa.Trees

- Arbitrarily choose a root from the vertices of the graph, then add all edges incident to this vertex (level 1),
- For each vertex at level 1, add each edge incident to this vertex to the tree as long as it does not produce a simple circuit (level 2),
- The same procedure until all the vertices in the tree have been added.

Example

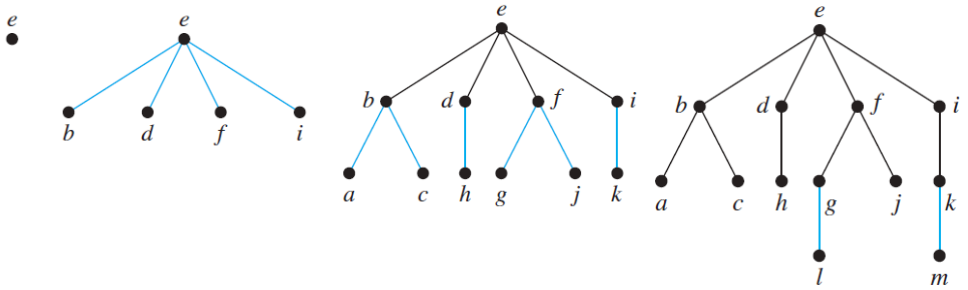
6 Spa.Trees

Find a spanning tree of this graph using Breadth-First Search:



Solution.

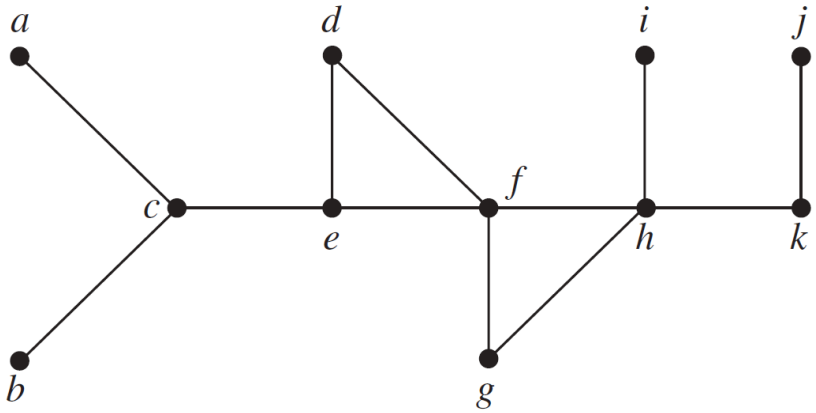
6 Spa.Trees



Example.

6 Spa.Trees

Find a spanning tree of this graph using Breadth-First Search:



Breadth-First Search Algorithm

procedure *BFS*(*G*: connected graph with vertices $v_1, v_2, \ldots, v_n$)
T := tree consisting only of vertex v_1
L := empty list
put v_1 in the list L of unprocessed vertices
while L is not empty
    remove the first vertex, v, from L
    for each neighbor w of v
        if w is not in L and not in T **then**
            add w to the end of the list L
            add w and edge $\{v, w\}$ to T

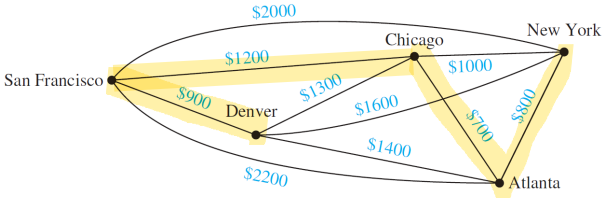
Table of Contents

7 Min.Span.Trees

Introduction

7 Min.Span.Trees

A company plans to build a communications network connecting its five computer centers.



- Which links should be made to ensure that there is a path between any two computer centers so that the total cost of the network is minimized?
- We can solve this problem by finding a spanning tree so that the sum of the weights of the edges of the tree is minimized. Such a spanning tree is called a minimum spanning tree.

Definition

7 Min.Span.Trees

A **minimum spanning tree** in a connected weighted graph is a spanning tree that has the smallest possible sum of weights of its edges.

Prim's Algorithm

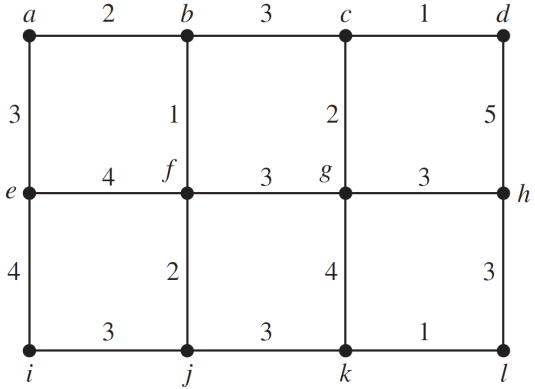
7 Min.Span.Trees

- Begin by choosing any edge with smallest weight, putting it into the spanning tree.
- Successively add to the tree edges of minimum weight that are incident to a vertex already in the tree, never forming a simple circuit with those edges already in the tree.
- Stop when $n - 1$ edges have been added.

Example

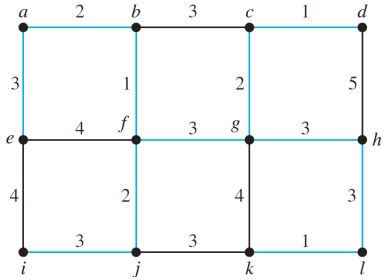
7 Min.Span.Trees

Use Prim's algorithm to find a minimum spanning tree in the graph



Solution.

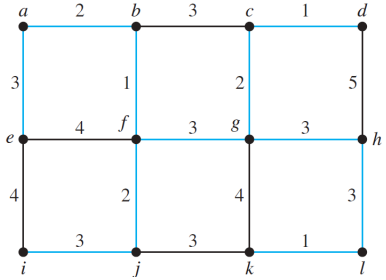
7 Min.Span.Trees



Choice	Edge	Weight
1	$\{b, f\}$	1
2	$\{a, b\}$	2
3	$\{f, j\}$	2
4	$\{a, e\}$	3
5	$\{i, j\}$	3

Solution.

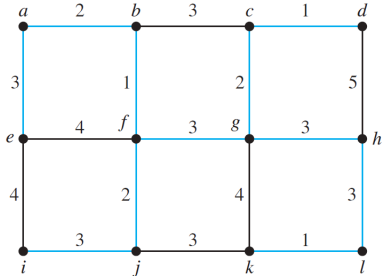
7 Min.Span.Trees



Choice	Edge	Weight
6	$\{f, g\}$	3
7	$\{c, g\}$	2
8	$\{c, d\}$	1
9	$\{g, h\}$	3
10	$\{h, l\}$	3

Solution.

7 Min.Span.Trees



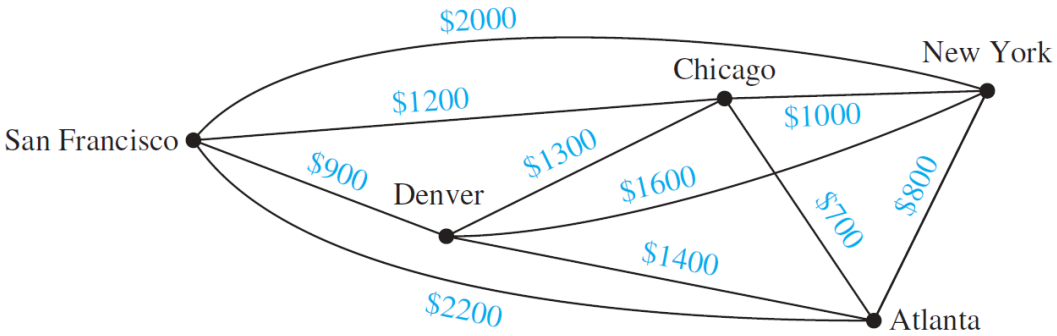
Choice	Edge	Weight
11	$\{k,\ l\}$	1

$\Longrightarrow$ Total 24

Example.

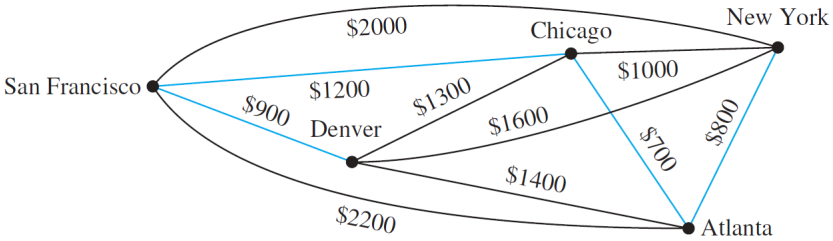
7 Min.Span.Trees

Use Prim's algorithm to find a minimum spanning tree in the graph



Solution.

7 Min.Span.Trees



Choice	Edge	Cost
1	{Chicago, Atlanta}	$ 700
2	{Atlanta, New York}	$ 800
3	{Chicago, San Francisco}	$1200
4	{San Francisco, Denver}	$ 900
	Total:	$3600

Prim's Algorithm

procedure *Prim*(G: weighted connected undirected graph with n vertices)
T := a minimum-weight edge
for $i := 1$ **to** $n - 2$
  e := an edge of minimum weight incident to a vertex in T and not forming a simple circuit in T if added to T
  $T := T$ with e added
return T {T is a minimum spanning tree of G}

Kruskal's Algorithm

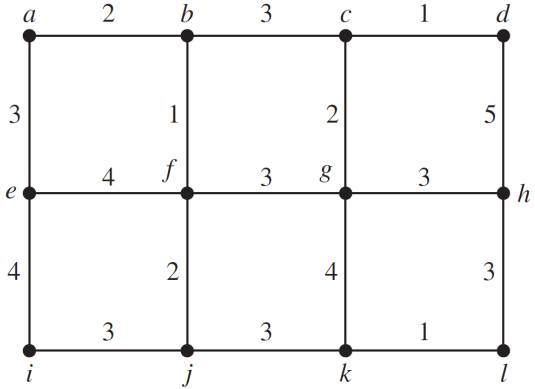
7 Min.Span.Trees

- Choose an edge in the graph with minimum weight,
- Successively add edges with minimum weight that do not form a simple circuit with those edges already chosen,
- Stop after $n - 1$ edges have been selected.

Example.

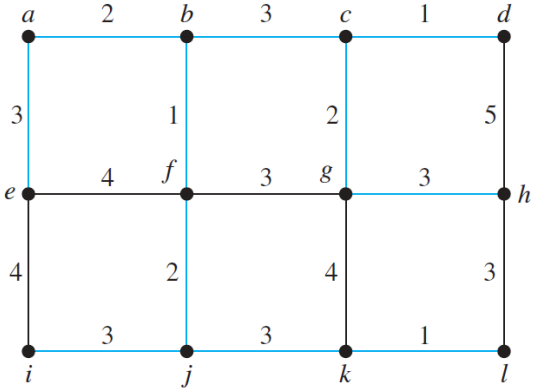
7 Min.Span.Trees

Use Kruskal's algorithm to find a minimum spanning tree in the weighted graph



Solution.

7 Min.Span.Trees



Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

Kruskal's Algorithm

7 Min.Span.Trees

procedure *Kruskal*(G: weighted connected undirected graph with n vertices)
T := empty graph
for $i := 1$ **to** $n - 1$
    e := any edge in G with smallest weight that does not form a simple circuit when added to T
    $T := T$ with e added
return T {T is a minimum spanning tree of G}

Note.

7 Min.Span.Trees

In Prim's algorithm, edges of minimum weight that are incident to a vertex already in the tree, and not forming a circuit, are chosen; whereas in Kruskal's algorithm edges of minimum weight that are not necessarily incident to a vertex already in the tree, and that do not form a circuit, are chosen.

Table of Contents

8 Problems

Quizzes

8 Problems

Use **Huffman coding algorithm** to encode the word "football".

What is the **average number** of bits required to encode a character?

Select one:

- a. 2.35
- b. 2.5
- c. 2.45
- d. 2.25

Quizzes

8 Problems

Use **Huffman coding algorithm** to encode the word "football".

What is the **average number** of bits required to encode a character?

Select one:

- a. 2.35
- b. 2.5
- c. 2.45
- d. 2.25

Ans: 2.5

Quizzes

8 Problems

Which codes are prefix codes?

(i) a: 1010, b: 010, c: 1101, d: 100

(ii) a: 10, b: 0101, c: 1110, d: 1001

Select one:

- a. (i)
- b. None
- c. Both
- d. (ii)

Quizzes

8 Problems

Which codes are prefix codes?

(i) a: 1010, b: 010, c: 1101, d: 100

(ii) a: 10, b: 0101, c: 1110, d: 1001

Select one:

- a. (i)
- b. None
- c. Both
- d. (ii)

Ans: (i)

Quizzes

8 Problems

Set up a **binary search tree** for the following list, in the given order, using alphabetical ordering: 5, 3, 6, 2, 4, 7.
Write the **preorder traversal** of the tree.

Select one:

- a. 5, 2, 3, 4, 6, 7
- b. 2, 4, 3, 7, 6, 5
- c. 5, 3, 2, 4, 6, 7
- d. 2, 3, 4, 5, 6, 7

Quizzes

8 Problems

Set up a **binary search tree** for the following list, in the given order, using alphabetical ordering: 5, 3, 6, 2, 4, 7.
Write the **preorder traversal** of the tree.

Select one:

- a. 5, 2, 3, 4, 6, 7
- b. 2, 4, 3, 7, 6, 5
- c. 5, 3, 2, 4, 6, 7
- d. 2, 3, 4, 5, 6, 7

Ans: 5, 3, 2, 4, 6, 7

Quizzes

8 Problems

How many **edges** does a full binary tree with 100 internal vertices have?

Select one:

- a. 200
- b. 100
- c. 101
- d. 201

Quizzes

8 Problems

How many **edges** does a full binary tree with 100 internal vertices have?

Select one:

- a. 200
- b. 100
- c. 101
- d. 201

Ans: 200

Quizzes

8 Problems

Which of the following statements are true?

(i) The bubble sort has complexity O (n^2)

(ii) If T is a binary tree with 101 vertices, its minimum height is 5

(iii) If T is a binary tree with 101 vertices, its minimum height is 7

(iv) Every full binary tree with 61 vertices has 30 leaves

Select one:

- a. (iv)
- b. (ii)
- c. (i)
- d. (iii)

Quizzes

8 Problems

Which of the following statements are true?

(i) The bubble sort has complexity O (n^2)

(ii) If T is a binary tree with 101 vertices, its minimum height is 5

(iii) If T is a binary tree with 101 vertices, its minimum height is 7

(iv) Every full binary tree with 61 vertices has 30 leaves

Select one:

- a. (iv)
- b. (ii)
- c. (i)
- d. (iii)

Ans: (i)

Quizzes

8 Problems

Study the following prefix expression:

+ - * 2 3 5 / * 2 4 4

It will be evaluated to ….

Select one:

- a. 5
- b. 4
- c. None of the others
- d. 6

Quizzes

8 Problems

Study the following prefix expression:

+ - * 2 3 5 / * 2 4 4

It will be evaluated to ....

Select one:

- a. 5
- b. 4
- c. None of the others
- d. 6

Ans: None of the others

Quizzes

8 Problems

Given the coding scheme: a: 00, b: 01, c: 10, d: 110, e: 111.
Find the word represented by: 11000010110111

Select one:

- a. abcde
- b. abcdee
- c. daabce
- d. dabbce

Quizzes

8 Problems

Given the coding scheme: a: 00, b: 01, c: 10, d: 110, e: 111.
Find the word represented by: 11000010110111

Select one:

- a. abcde
- b. abcdee
- c. daabce
- d. dabbce

Ans: dabbce

Quizzes

8 Problems

The following prefix expression will be evaluated to .....

+ - * 2 3 4 / + 1 2 3

Select one:

- a. 4
- b. 2
- c. 1
- d. 3

Quizzes

8 Problems

The following prefix expression will be evaluated to .....

+ - * 2 3 4 / + 1 2 3

Select one:

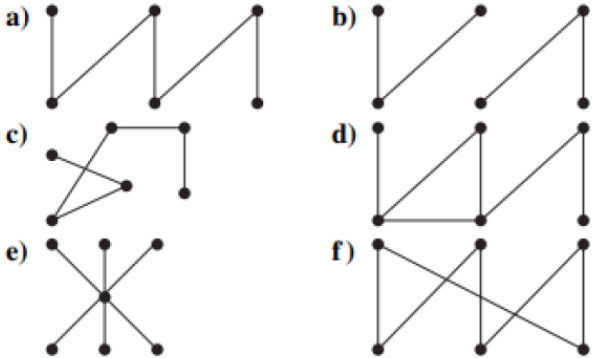
- a. 4
- b. 2
- c. 1
- d. 3

Ans: 3

Introduction to Trees

8 Problems

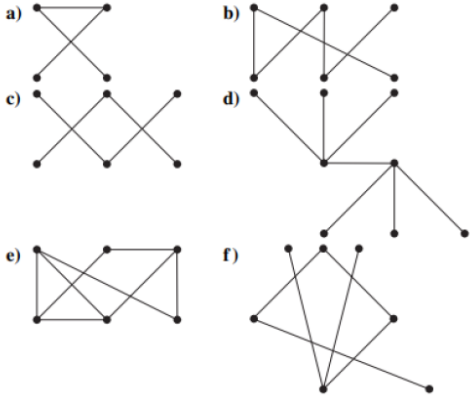
1. Which of these graphs are trees?



Introduction to Trees

8 Problems

2. Which of these graphs are trees?



Introduction to Trees

8 Problems

3. Construct a complete binary tree of height 4 and a complete 3-ary tree of height 3.
4. Which complete bipartite graphs $K_{m,n}$, where m and n are positive integers, are trees?
5. How many edges does a tree with 10,000 vertices have?
6. How many vertices does a full 5-ary tree with 100 internal vertices have?
7. How many edges does a full binary tree with 1000 internal vertices have?
8. How many leaves does a full 3-ary tree with 100 vertices have?

Applications of Trees

8 Problems

1. Form a binary search tree for the words *mathematics, physics, geography, zoology, meteorology, geology, psychology* and *chemistry* (using alphabetical order).

2. Use Huffman coding to encode the following symbols with the frequencies listed: A: 0.08, B: 0.10, C: 0.12, D: 0.15, E: 0.20, F: 0.35. What is the average number of bits used to encode a character?

3. Build a binary search tree for the words *banana, peach, apple, pear, coconut, mango* and *papaya* using alphabetical order.

4. Build a binary search tree for the words *oenology, phrenology, campanology, ornithology, ichthyology, limnology, alchemy* and *astrology* using alphabetical order.

5. How many comparisons are needed to locate or to add each of these words in the search tree for Exercise 3, starting fresh each time?

a) pear b) banana c) kumquat d) orange

Applications of Trees

8 Problems

6. How many comparisons are needed to locate or to add each of the words in the search tree for Exercise 4, starting fresh each time?

a) palmistry b) etymology c) paleontology d) glaciology

7. Which of these codes are prefix codes?

a) a: 11, e: 00, t: 10, s: 01

b) a: 0, e: 1, t: 01, s: 001

c) a: 101, e: 11, t: 001, s: 011, n: 010

d) a: 010, e: 11, t: 011, s: 1011, n: 1001, i: 10101

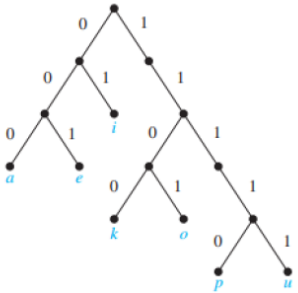
8. Construct the binary tree with prefix codes representing these coding schemes.

a) a: 11, e: 0, t: 101, s: 100

Applications of Trees

8 Problems

9. What are the codes for a, e, i, k, o, p, and u if the coding scheme is represented by this tree?



Applications of Trees

8 Problems

10. Given the coding scheme a: 001, b: 0001, e: 1, r: 0000, s: 0100, t: 011, x: 01010, find the word represented by

a) 01110100011 b) 0001110000

c) 0100101010 d) 01100101010

11. Use Huffman coding to encode these symbols with given frequencies: a: 0.20, b: 0.10, c: 0.15, d: 0.25, e: 0.30. What is the average number of bits required to encode a character?

12. Use Huffman coding to encode these symbols with given frequencies: A: 0.10, B: 0.25, C: 0.05, D: 0.15, E: 0.30, F: 0.07, G: 0.08. What is the average number of bits required to encode a symbol?

13. Construct two different Huffman codes for these symbols and frequencies: t: 0.2, u: 0.3, v: 0.2, w: 0.3.

Tree Traversal

8 Problems

1. What is the ordered rooted tree that represents the expression $((x + y)\uparrow 2) + ((x - 4)/3)$?

2. What is the prefix form for $((x + y) \uparrow 2) + ((x - 4)/3)$?

3. What is the value of the prefix expression $+ - * 2\ 3\ 5/\uparrow 2\ 3\ 4$?

4. What is the postfix form of the expression $((x + y) \uparrow 2) + ((x - 4)/3)$?

5. What is the value of the postfix expression $7\ 2\ 3\ * - 4 \uparrow 9\ 3/+$?

6. a) Represent the expression $((x + 2) \uparrow 3) * (y - (3 + x)) - 5$ using a binary tree.

Write this expression in

b) prefix notation c) postfix notation d) infix notation

Tree Traversal

8 Problems

7. a) Represent the expressions $(x + xy) + (x/y)$ and $x + ((xy + x)/y)$ using binary trees.

Write these expressions in

b) prefix notation c) postfix notation d) infix notation

8. a) Represent the compound propositions $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ and $(\neg p \wedge (q \leftrightarrow \neg p)) \vee \neg q$ using ordered rooted trees.

Write these expressions in

b) prefix notation c) postfix notation d) infix notation

9. a) Represent $(A \cap B) - (A \cup (B - A))$ using an ordered rooted tree.

Write this expression in

b) prefix notation c) postfix notation d) infix notation

Tree Traversal

8 Problems

10. Draw the ordered rooted tree corresponding to each of these arithmetic expressions written in prefix notation. Then write each expression using infix notation.

a) $+ * + - 5\ 3\ 2\ 1\ 4$ b) $\uparrow + 2\ 3 - 5\ 1$ c) $* / 9\ 3 + * 2\ 4 - 7\ 6$

11. What is the value of each of these prefix expressions?

a) $- * 2 / 8\ 4\ 3$ b) $\uparrow - * 3\ 3 * 4\ 2\ 5$

c) $+ - \uparrow 3\ 2 \uparrow 2\ 3 / 6 - 4\ 2$ d) $* + 3 + 3 \uparrow 3 + 3\ 3\ 3$

Tree Traversal

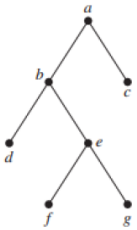
8 Problems

12. What is the value of each of these postfix expressions?

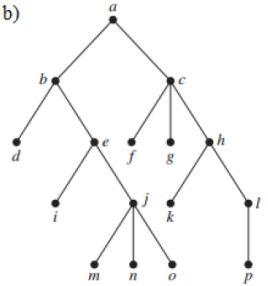
a) 5 2 1 − − 3 1 4 ++ ∗ b) 9 3 / 5 + 7 2 − ∗ c) 3 2 ∗ 2 ↑ 5 3 − 8 4 / ∗ −

13. Determine the order in which a preorder traversal visits the vertices of the given ordered rooted tree.

a)



b)



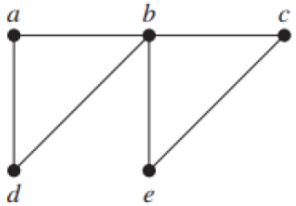
Spanning Trees

8 Problems

1. How many edges must be removed from a connected graph with n vertices and m edges to produce a spanning tree?

2. Find a spanning tree for the graph shown by removing edges in simple circuits.

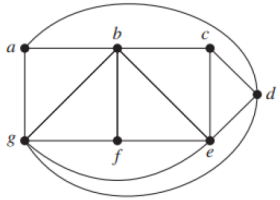
a.



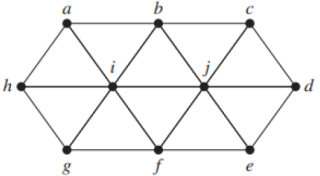
Spanning Trees

8 Problems

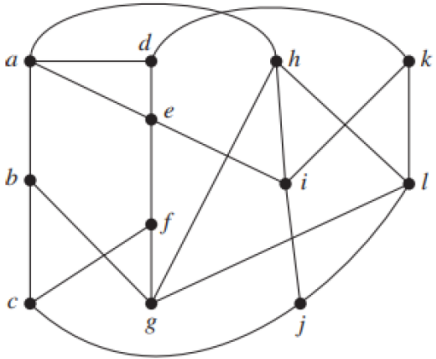
b.



c.



f.



Spanning Trees

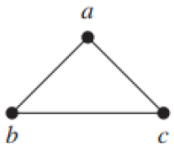
8 Problems

3. Find a spanning tree for each of these graphs.

a) K_5 **b)** $K_{4,4}$ **c)** $K_{1,6}$
d) Q_3 **e)** C_5 **f)** W_5

4. Draw all the spanning trees of the given simple graphs.

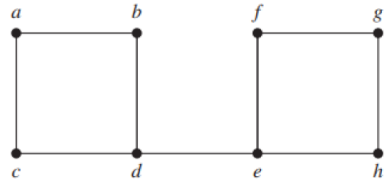
a.



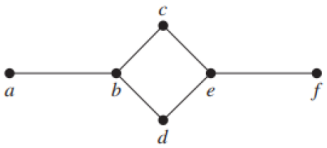
Spanning Trees

8 Problems

b.



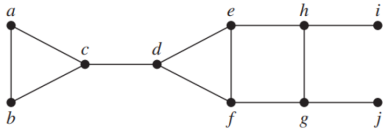
c.



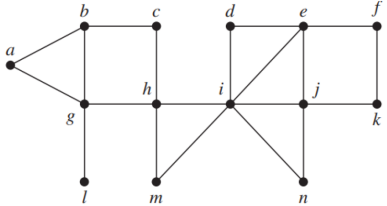
Spanning Trees

8 Problems

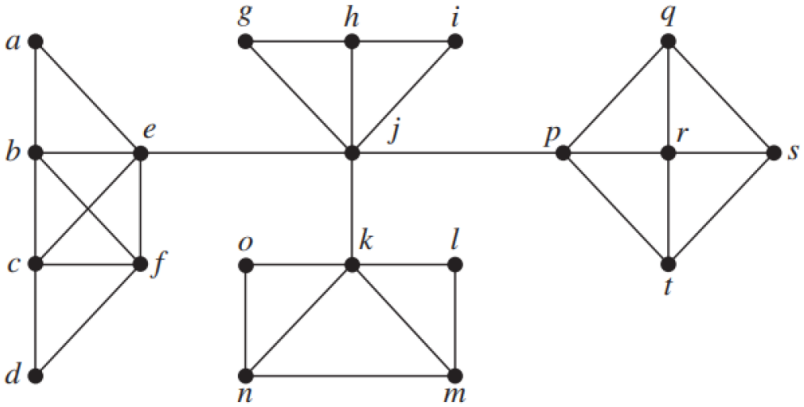
5. Use depth-first search and breadth-first search to produce a spanning tree for the given simple graph. Choose a as the root of this spanning tree and assume that the vertices are ordered alphabetically



b.



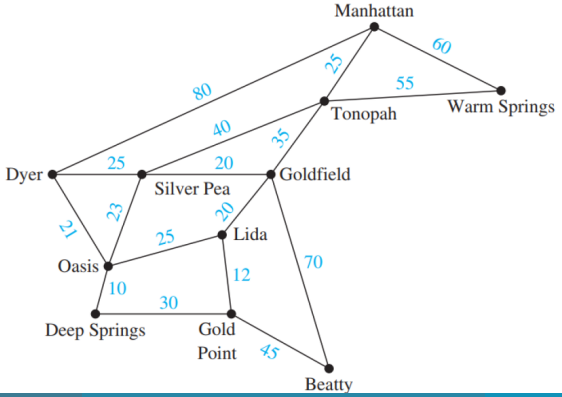
c.



Minimum Spanning Trees

8 Problems

1. The roads represented by this graph are all unpaved. The lengths of the roads between pairs of towns are represented by edge weights. Which roads should be paved so that there is a path of paved roads between each pair of towns so that a minimum road length is paved? (Note: These towns are in Nevada.)

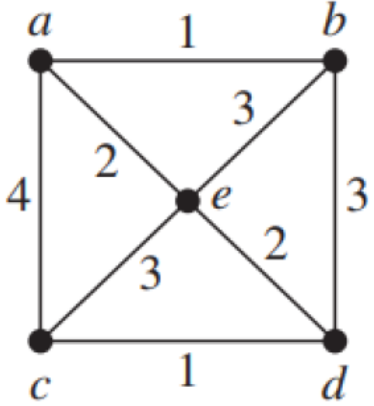


Minimum Spanning Trees

8 Problems

Minimum Spanning Trees

8 Problems

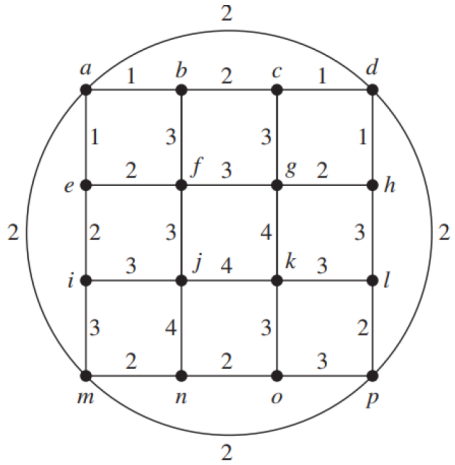


Minimum Spanning Trees

8 Problems

Minimum Spanning Trees

8 Problems



Minimum Spanning Trees

8 Problems

Q&A

Thank you for listening!