Discrete Mathematics

TREES

Department of Mathematics

Quynhon, 2023

Outline of Chapter

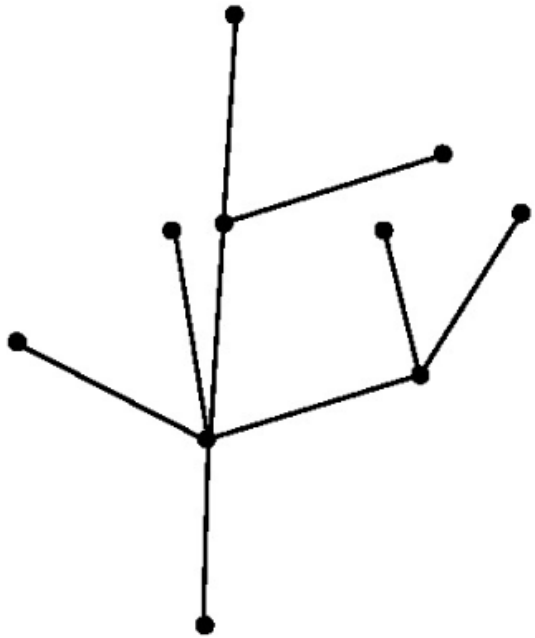
Textbook: Kenneth Rosen, **Discrete Mathematics and its applications.** Ed.6 (or 7), Mc.Graw Hill, IE, 2007.

FPT UNIVERSITY

Table of Contents

FPT UNIVERSITY

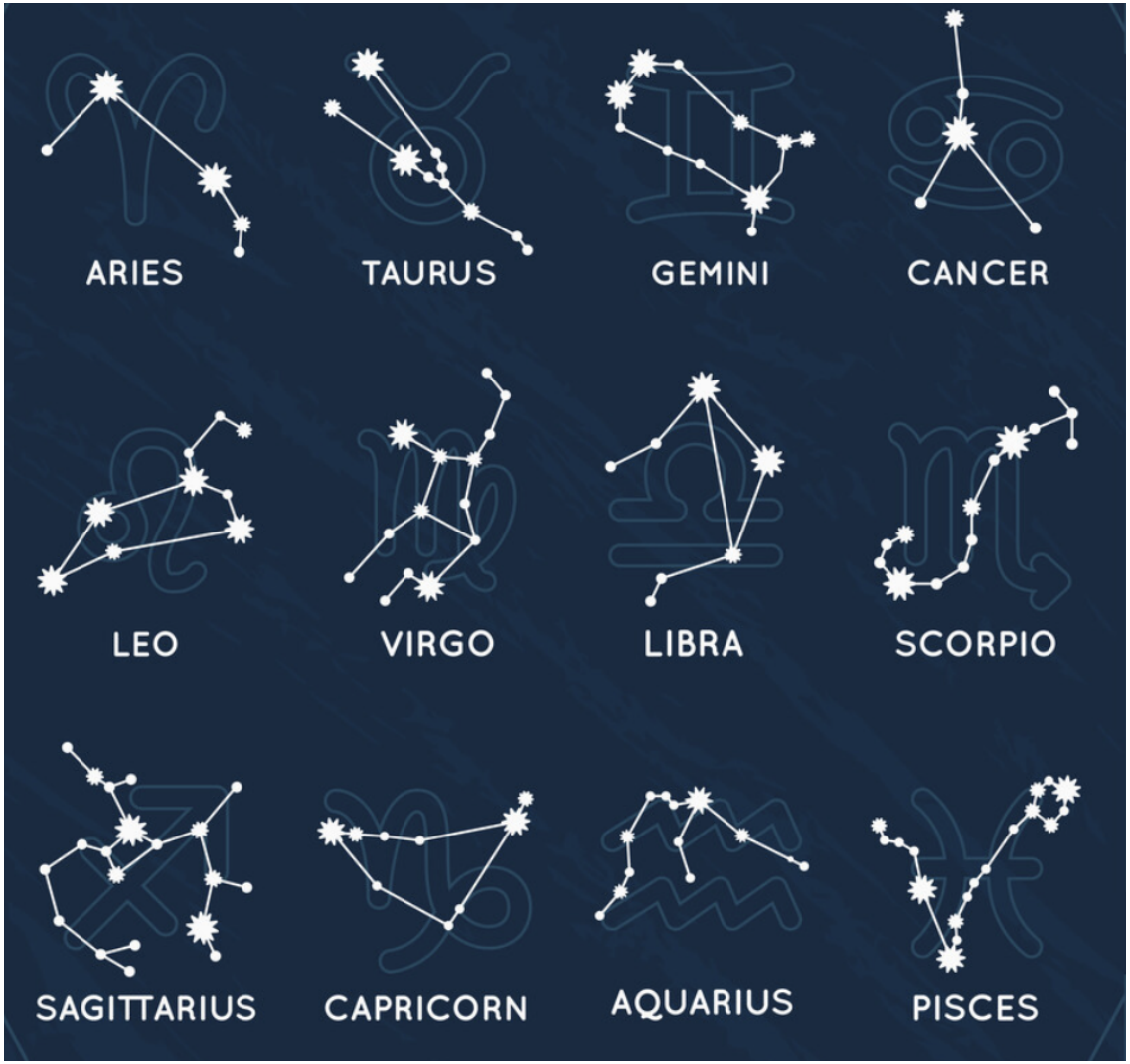
A tree is a connected undirected simple graph with no simple circuits.



Theorem

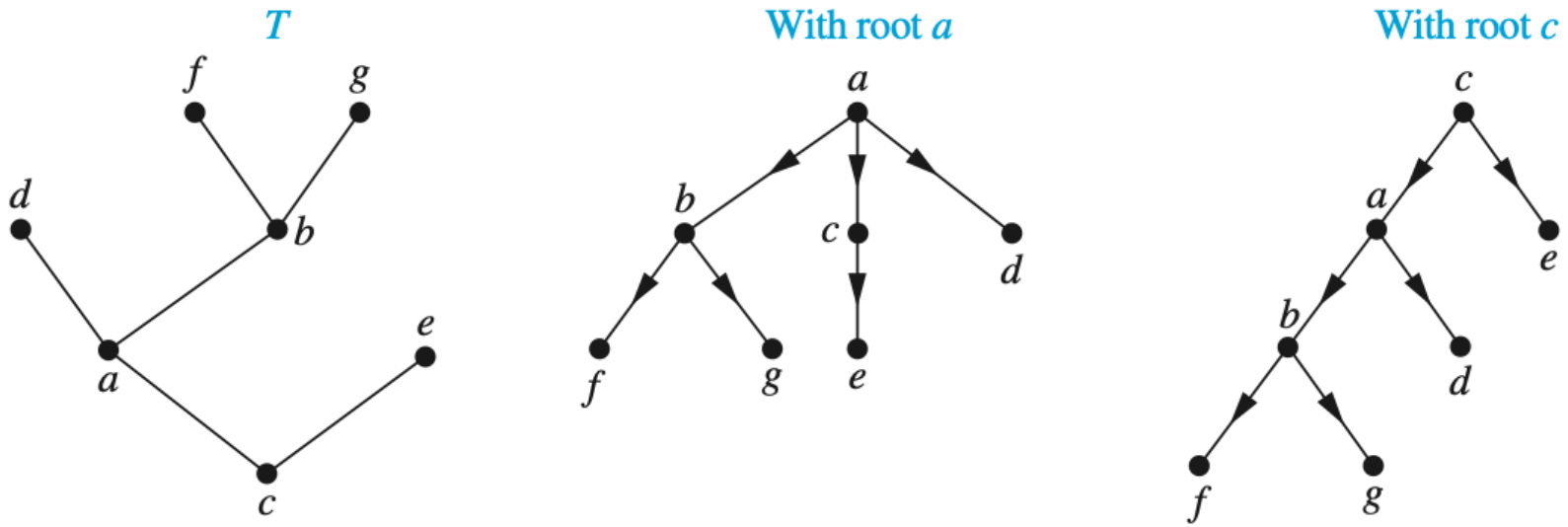
- An undirected graph is a tree if and only if between any two vertices there is a unique simple path.
- A tree with n vertices has $n - 1$ edges.

Which constellations of the zodiac
whose charts look like trees in Graph theory?



FPT UNIVERSITY

A rooted tree is a tree in which one vertex is designed as a root and every edge is directed away from the root.



Remark

The choice of root determines the directions of the edges.

Terminologies

- If there is an edge directed from u to v then u is called the parent of v and v is called a child of u.
- Vertices having the same parent are called siblings.
- Ancestors of a vertex are the vertices on the path from the root to that vertex, excluding the vertex itself.
- Descendants of a vertex v are the vertices that have v as an ancestor.
- Leaves of a tree are the vertices that do not have children.
- Internal vertices are the vertices that have children.
- A subtree with root a is the subgraph consisting of a and all its descendants and all edges incident to these descendants.

An illustration of family tree

Describe the Bernoulli's family tree by using the terminologies: **parent, child, sibling, ancestor, descendant, leaf, internal vertex** and **subtree**.

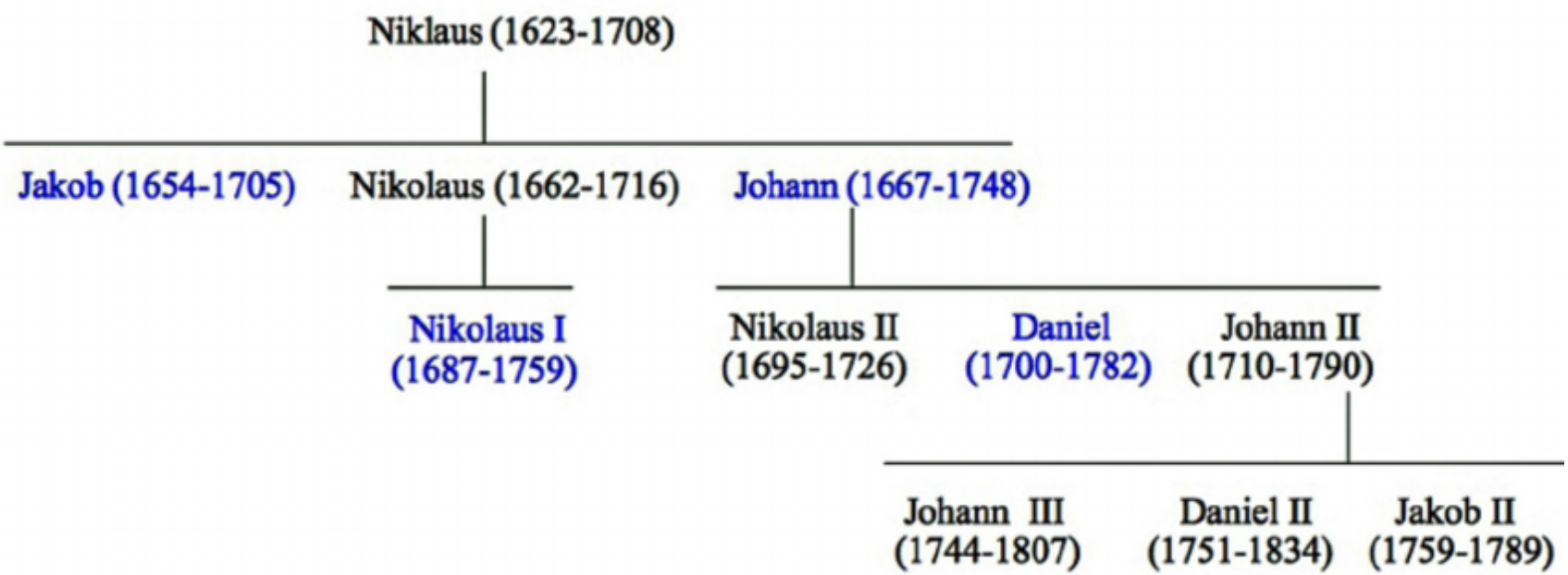
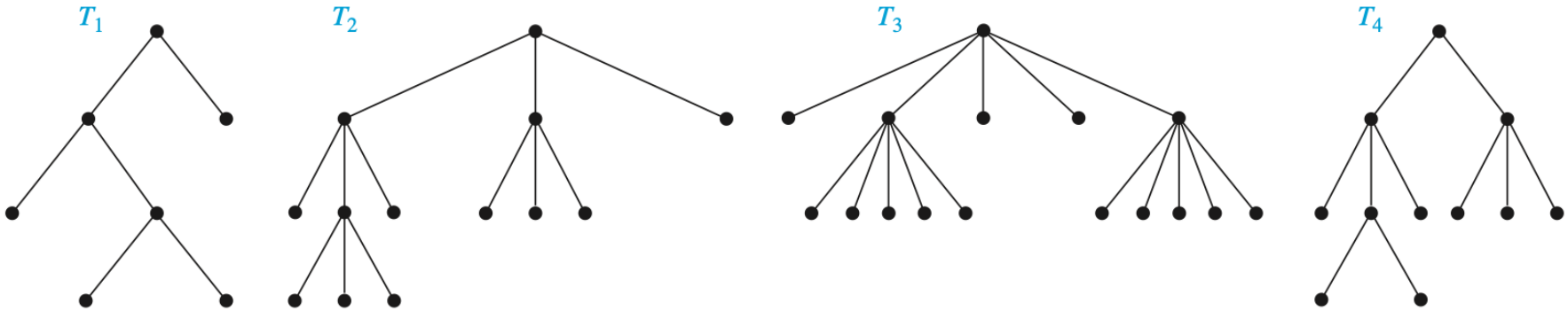


Figure: The Bernoulli Family of Mathematicians

- A rooted tree is called an *m*-ary tree if each vertex has at most *m* children.
- A rooted tree is called a full *m*-ary tree if each internal vertex has exactly *m* children.
- A 2-ary tree is called a binary tree.

Quiz

Are the rooted trees shown in the following figure full m-ary trees for some positive integer *m*?



Theorem

A full m-ary tree with

- n vertices has $i = (n-1)/m$ internal vertices & $\ell = [(m-1)n+1]/m$ leaves,
- i internal vertices has $n = mi+1$ vertices & $\ell = (m-1)i+1$ leaves,
- ℓ leaves has $n = (m\ell-1)/(m-1)$ vertices & $i = (\ell-1)/(m-1)$ internal vertices.

Examples

1. Does there exist a full 4-ary tree with 80 leaves?
2. Some one starts a chain letter game. Each person receiving the letter either sends it to exactly four other persons, or to no one. Assume that no one receives more than one letter. The game ends when there have been exactly 100 persons receiving the letter and not sending it out.
   How many persons received the letter (including the one who starts the game)?

Examples: Hints and Solutions

Example 1. There does not exist a full 4-ary tree with 80 leaves since the number of internal vertices is

$$i = (\ell - 1)/(m - 1) = (80 - 1)/(4 - 1) \notin \mathbb{N}.$$

Example 2. The chain letter can be represented using a 4-ary tree. The internal vertices correspond to people who sent out the letter, and the leaves correspond to people who did not send it out. Because 100 people did not send out the letter, the number of leaves in this rooted tree is $\ell = 100$. Hence, Theorem shows that the number of people who have seen the letter is $n = (4.100 - 1)/(4 - 1) = 133$. Also, the number of internal vertices is $133 - 100 = 33$, so 33 people sent out the letter.

FPT UNIVERSITY

- The level of a vertex in a rooted tree is the length of the unique path from the root to that vertex.
- The height of a rooted tree is the maximum of the levels of the vertices.
- An m-ary tree of height h is called a balanced tree if each leave has level h or $h-1$.

Theorem

- For any m-ary tree with height h and ℓ leaves, we have $\ell \leq m^h$. Hence,
$$h \geq \lceil \log_m(\ell) \rceil.$$
- In a full and balanced m-ary tree, we have $h = \lceil \log_m(\ell) \rceil$.

Quiz

Does there exist a full and balanced m-ary tree with height 4 and 100 leaves?

Table of Contents

FPT UNIVERSITY

Binary Search Trees

- Form a binary search tree for the words: *mathematics*, *physics*, *geography*, *zoology*, *meteorology*, *geology*, *psychology*, and *chemistry*.

How many comparisons needed to determine whether the words *meteorology* or *oceanography* are already in the list?

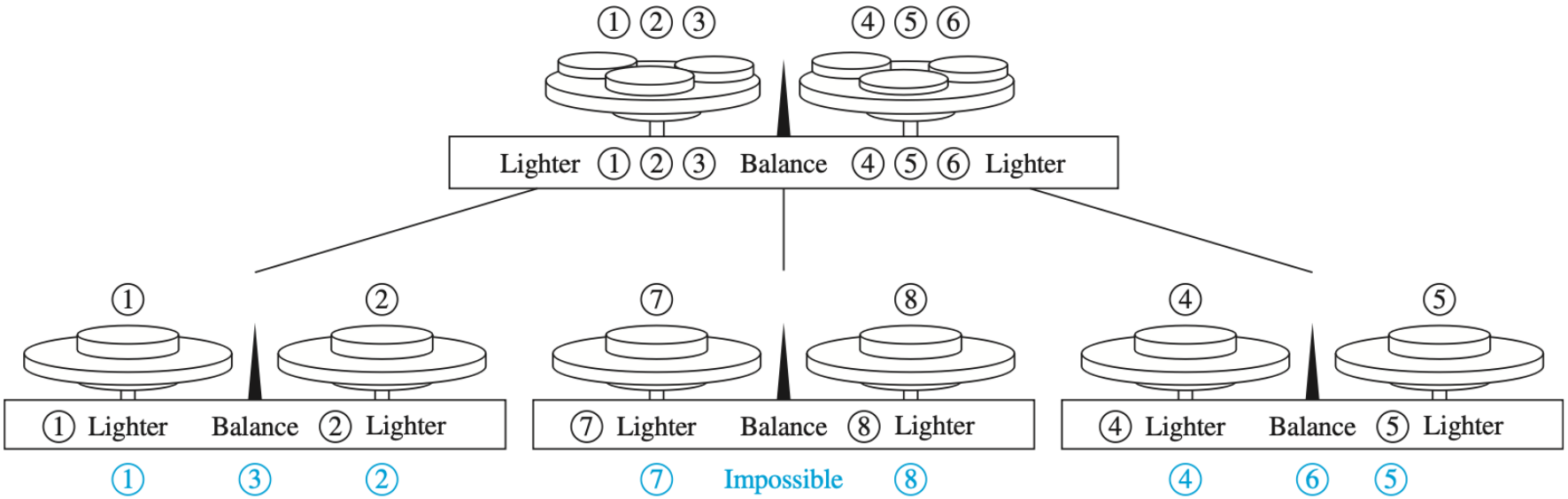
Theorem

The least number of comparisons to locate an element in a list of length n is $\lceil \log(n+1) \rceil$.

Decision Trees

Suppose there are seven coins, all with the same weight, and a counterfeit coin that weighs less than the others. By using a balance scale, how many weighing are necessary to determine the counterfeit coin?

Hint.



Complexity of Sorting Algorithms

How many comparisons are necessary to sort a list of 3 numbers using binary comparisons?

Theorem

A sorting algorithm based on binary comparisons requires at least $\lceil \log(n!) \rceil$ comparisons.

Note.

$$\log(n!) = \log 1 + \log 2 + \cdots + \log n = \Theta(n \log n).$$

Prefix code

To encode the message good morning by a bit string we can represent each letter by a bit string of length 3. In this case the total number of bits used is 33.

Problem: Is this possible to encode this message using fewer bits?

- We cannot use bit strings of length 2 to represent the letters of this message. We need to use bit strings of different lengths.

- However, there might be troubles when decoding the encoded message.

Example

If encode the message tea using representations e: 0, a: 1, t: 01 then the bit string 0101 can be decoded as eat, eaea or tt.

Prefix code

- Prefix codes are bit strings used to represent letters in such a way that no string is the first part of another string.
- Binary trees can be used to create prefix codes.

Example

Use binary trees to create prefix codes to encode the message good morning.

Quiz

How many bits are used? Is this possible to construct a binary tree for which fewer bits are used?

FPT UNIVERSITY

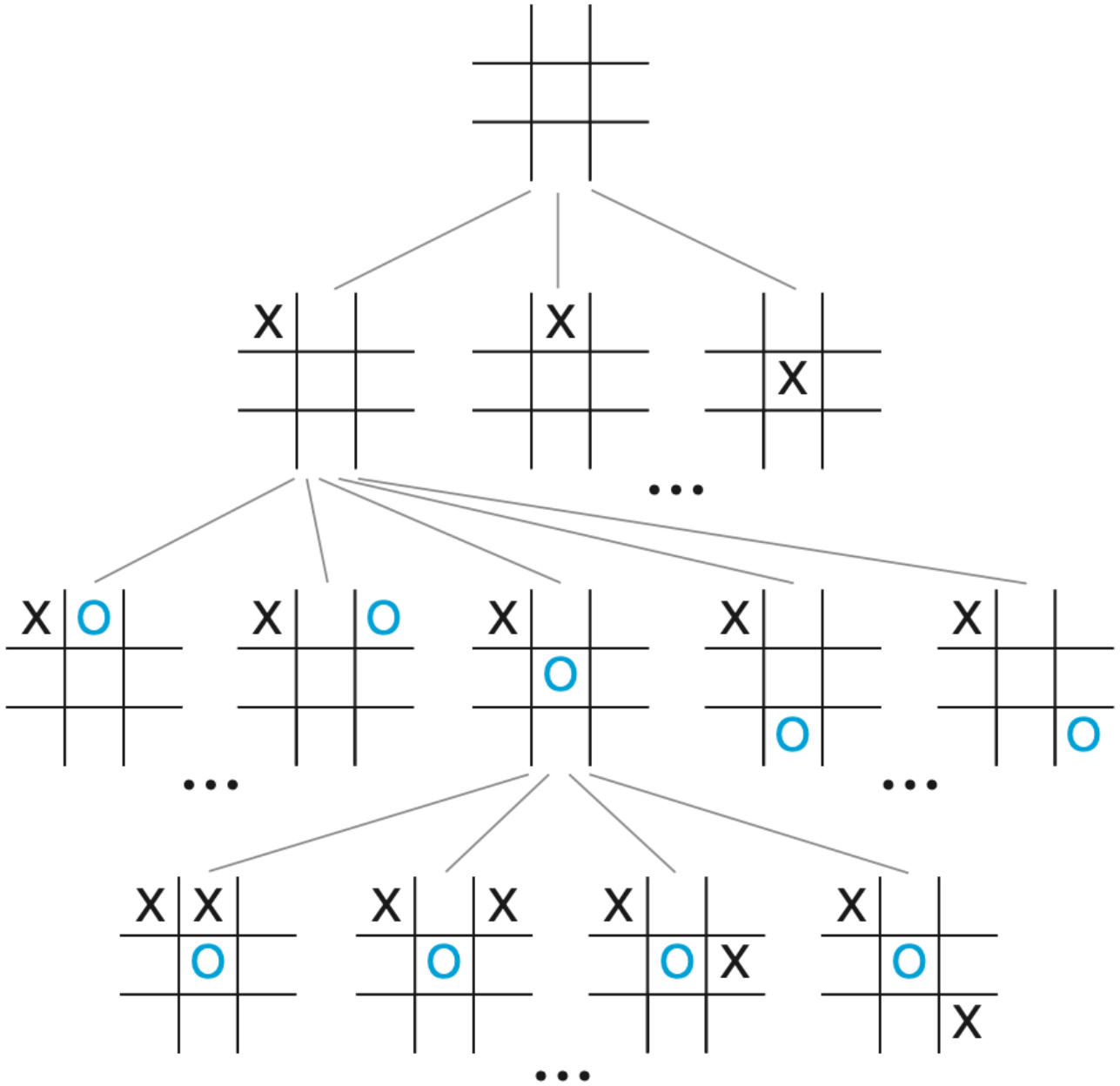
Huffman Coding

- Write out all distinct letters of the message together with their frequencies. Consider each letter as a binary tree with only one vertex. Call the frequency corresponding to each tree its weight.
- At each step, combine two trees whose sum of weights is a minimum into a single tree by adding a new root, and assign this sum of weights as the weight of the new tree.
- The algorithm is finished if a single tree is constructed.

Examples

1. Construct a Huffman coding to encode the message good morning. How many bits are used, and what is the average number of bits used for each letter?
2. Construct a Huffman coding to encode the letters with their frequencies, and find the average number of bits used to encode a letter: A: 0.08, B: 0.1, C: 0.12, D: 0.15, E: 0.2, F: 0.35.

Game trees



Game trees (2)

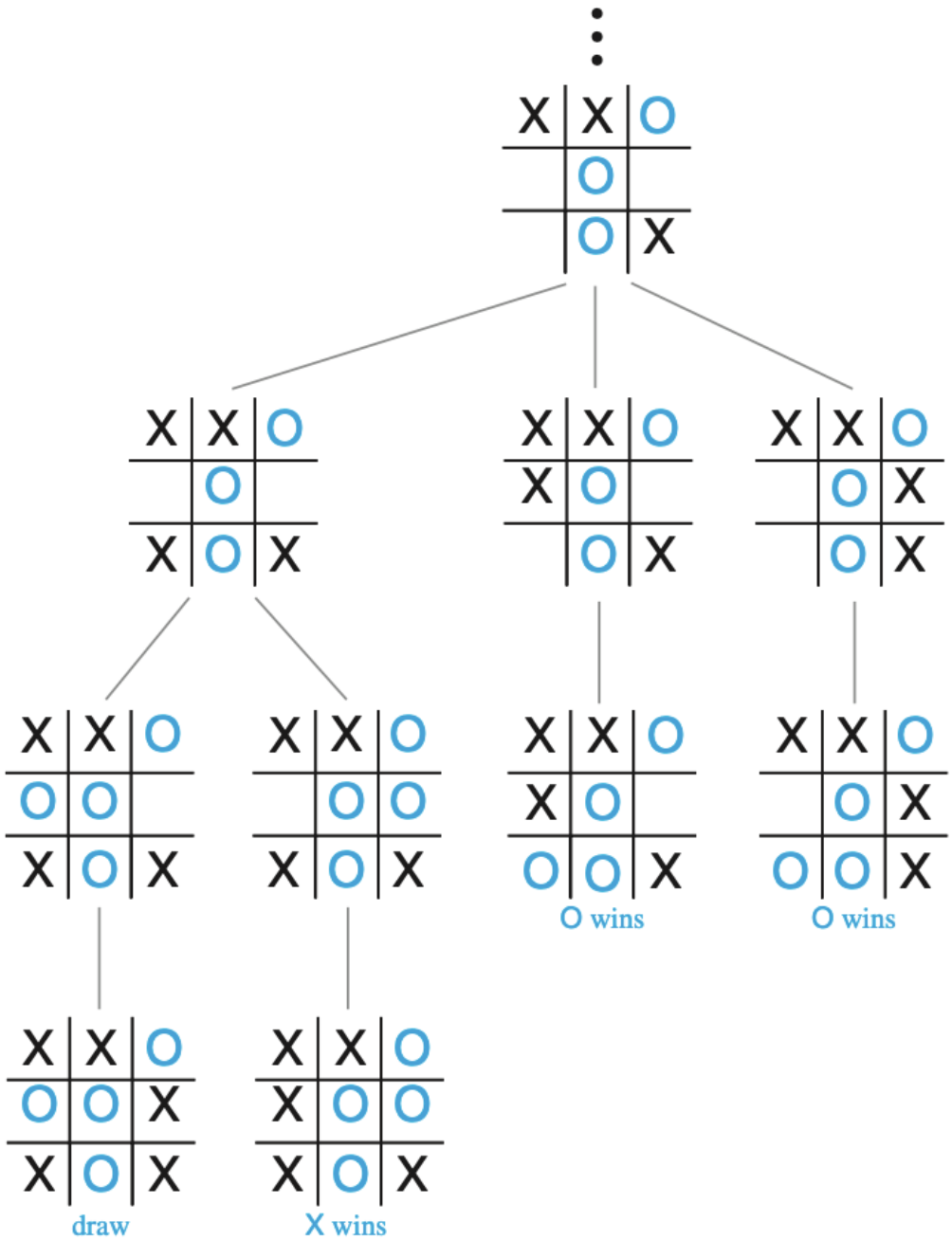
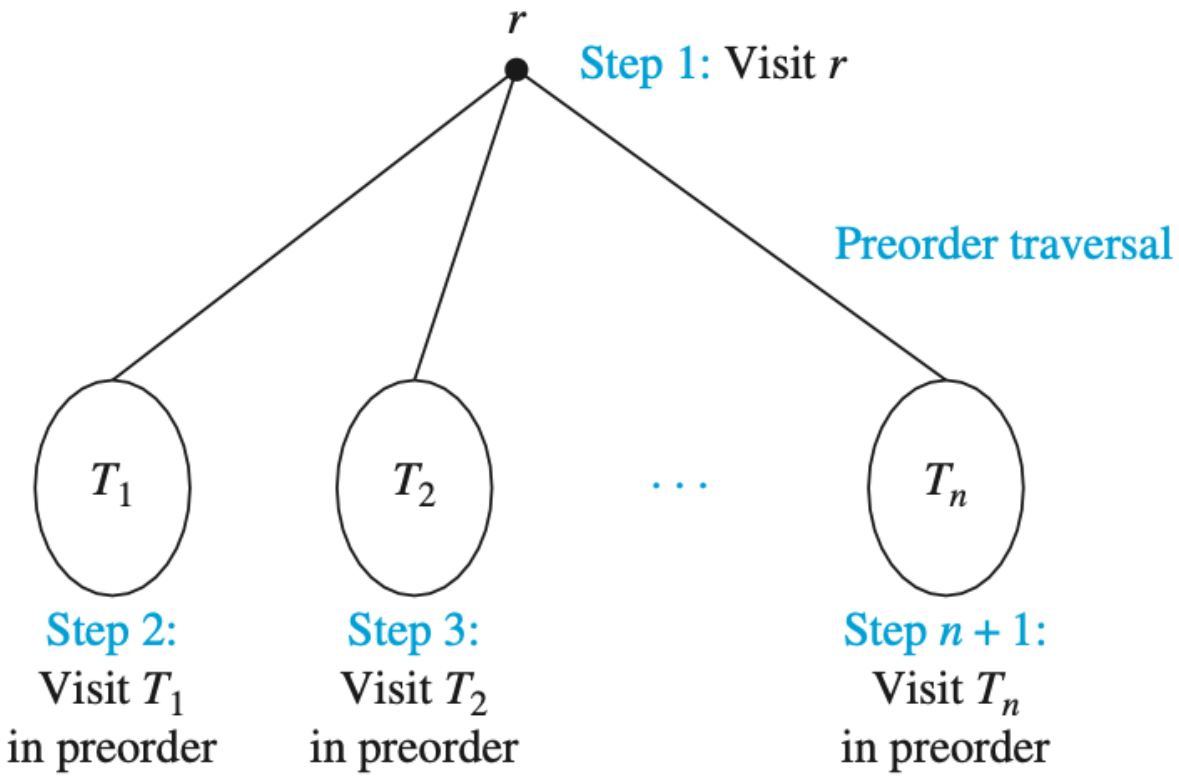


Table of Contents

FPT UNIVERSITY

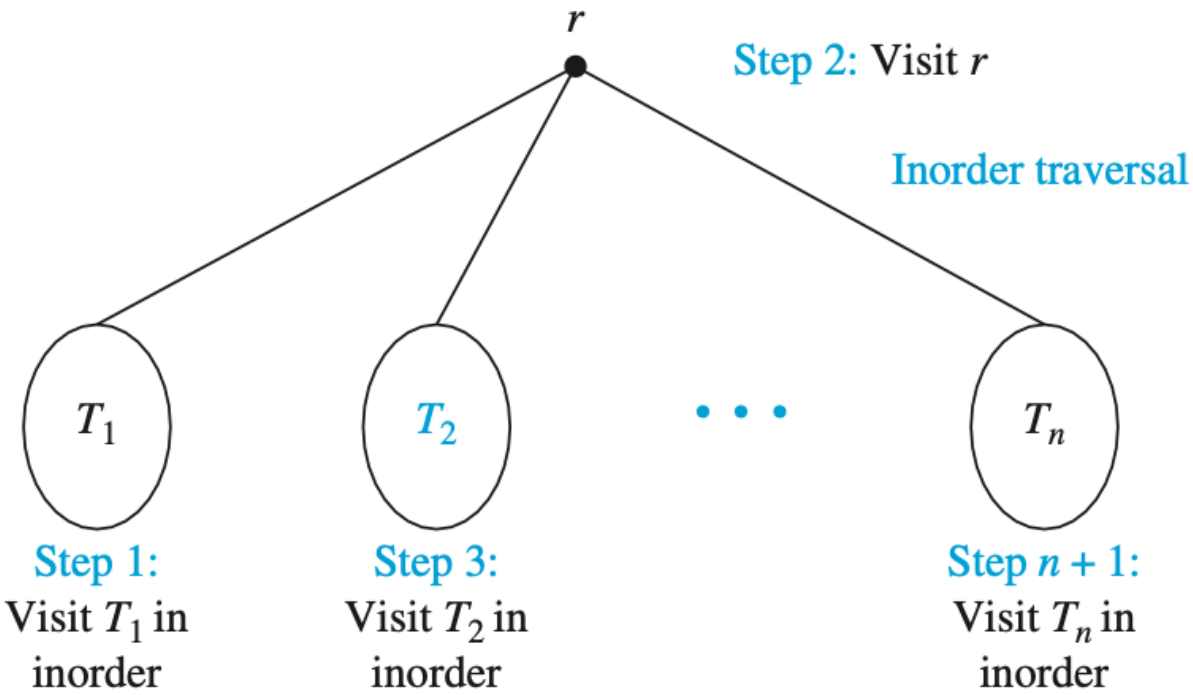
Preorder Traversal

Preorder Traversal: Visit root, visit subtrees left to right



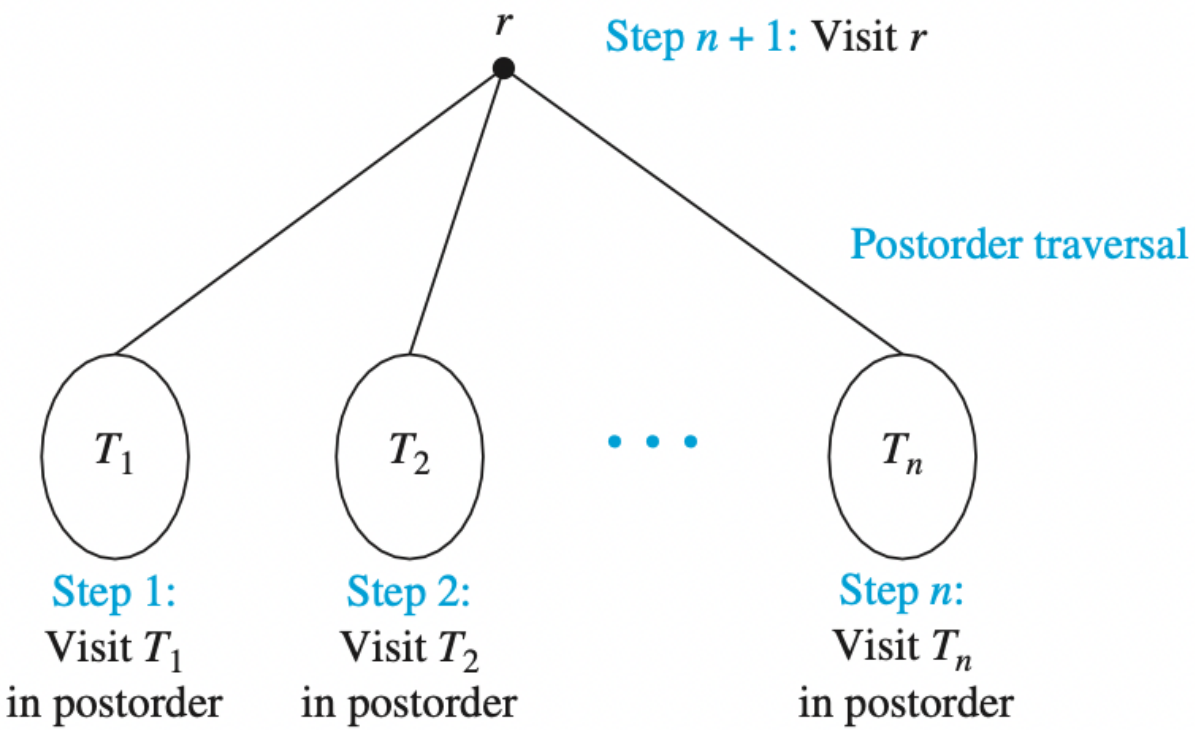
Inorder traversal

Inorder traversal: Visit leftmost subtree, visit root, visit other subtrees left to right

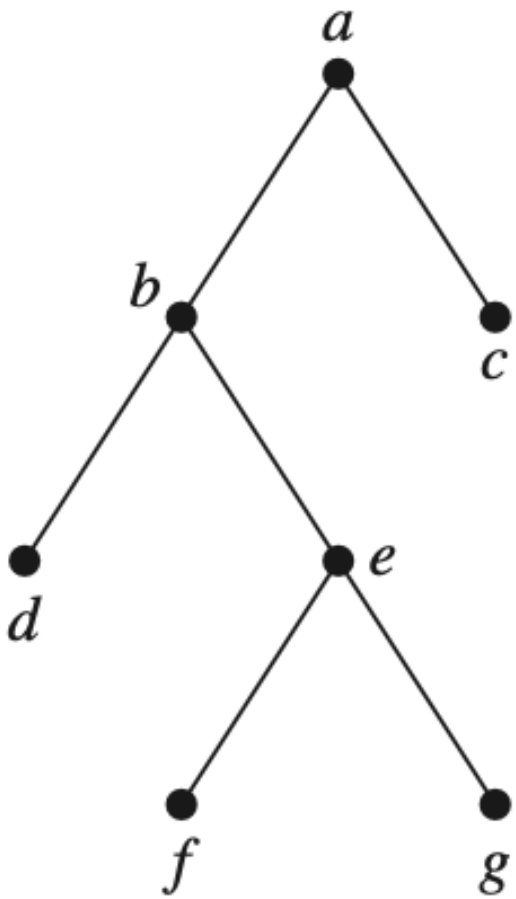


Postorder traversal

Postorder traversal: Visit subtrees left to right; visit root



Match the order in which a traversal visits the vertices of the given ordered rooted tree.



A. Preorder traversal

B. Inorder traversal

C. Postorder traversal

1. d f g e b c a

2. a b d e f g c

3. d b f e g a c

Infix, Prefix, and Postfix Notation

- Complicated expressions, such as compound propositions, combinations of sets, and arithmetic expressions can be represented by using ordered rooted trees.

- We can use a binary tree to represent an expression, in which the internal vertices indicate operations, and the leaves represent numbers or variables.

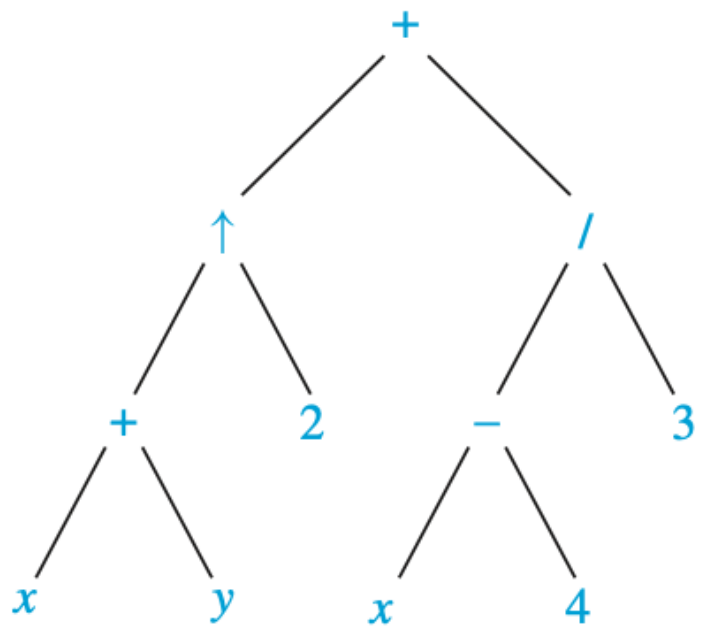


Figure: A Binary Tree Represents $(x + y)^2 + (x - 4)/3$.

Infix, Prefix, and Postfix Notation (2)

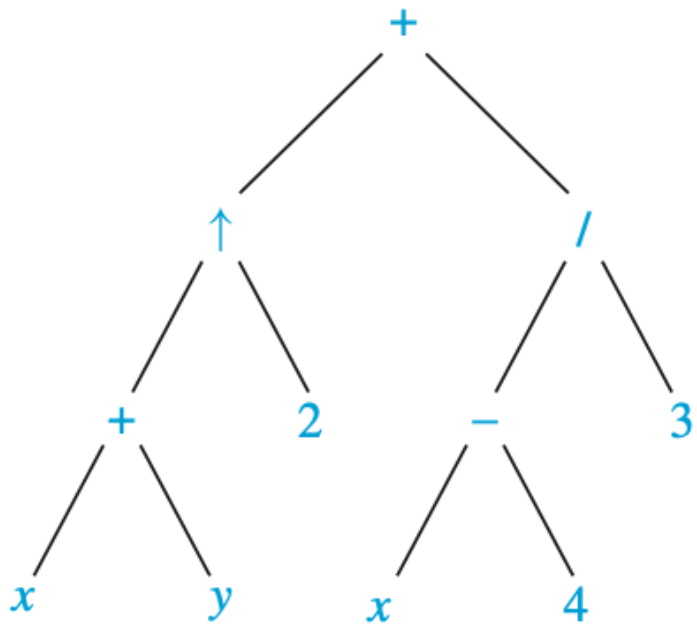
We obtain

- the prefix form of an expression when we traverse its rooted tree in preorder. Expressions written in prefix form are said to be in **Polish notation**.
- the postfix form of an expression by traversing its binary tree in postorder. Expressions written in postfix form are said to be in **reverse Polish notation**.
- infix notation: the form of an expression (including a full set of parentheses) obtained from an inorder traversal of the binary tree representing this expression.

FPT UNIVERSITY

Infix, Prefix, and Postfix Notation (3)

For example, the following binary tree has:



- Prefix notation: $+ \uparrow + x\, y\, 2 / - x\, 4\, 3$
- Postfix notation: $x\, y + 2 \uparrow x\, 4 - 3 / +$
- Infix notation: $(x + y)^2 + (x - 4)/3$

Note

To evaluate the value of a form:

- For **postfix form**, **work from left to right**, carrying out operations whenever an operator follows two operands. After each operation is carried out, the result of this operation is a new operand.
- For **prefix form**, follows the same procedure, but **work from right to left** instead.

Example.

- Find value of the prefix expression $+ - * 2\,3\,5 \,/\, \uparrow 2\,3\,4$.
- Find value of the postfix expression $7\,2\,3 * - 4 \uparrow 9\,3 \,/\, +$.

Evaluating a Prefix Expression

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

$$+ \quad - \quad * \quad 2 \quad 3 \quad 5 \quad / \quad \underbrace{\uparrow \quad 2 \quad 3} \quad 4$$

$$2 \uparrow 3 = 8$$

$$+ \quad - \quad * \quad 2 \quad 3 \quad 5 \quad \underbrace{/ \quad 8 \quad 4}$$

$$8 / 4 = 2$$

$$+ \quad - \quad \underbrace{* \quad 2 \quad 3} \quad 5 \quad 2$$

$$2 * 3 = 6$$

$$+ \quad \underbrace{- \quad 6 \quad 5} \quad 2$$

$$6 - 5 = 1$$

$$\underbrace{+ \quad 1 \quad 2}$$

$$1 + 2 = 3$$

Evaluating a Postfix Expression

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

$$7\;\underbrace{2\;3\;*}_{2*3=6}\;-\;4\;\uparrow\;9\;3\;/\;+$$

$$\underbrace{7\;6\;-}_{7-6=1}\;4\;\uparrow\;9\;3\;/\;+$$

$$\underbrace{1\;4\;\uparrow}_{1^4=1}\;9\;3\;/\;+$$

$$1\;\underbrace{9\;3\;/}_{9/3=3}\;+$$

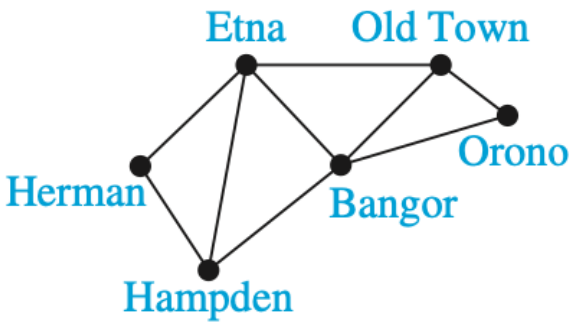
$$\underbrace{1\;3\;+}_{1+3=4}$$

Table of Contents

FPT UNIVERSITY

Spanning tree

Many situations coming from real life lead us to **find a connected subgraph with the minimum number of edges containing all vertices of the original simple graph.**

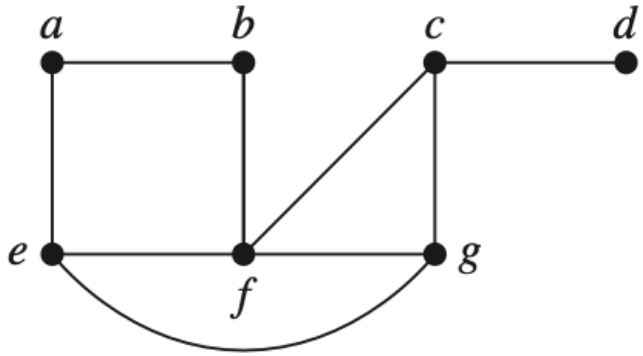


For example, the system of roads in Maine is usually covered by snow in the winter. The highway department wants to **plow the fewest roads so that there will always be cleared roads connecting any two towns.** How can this be done?

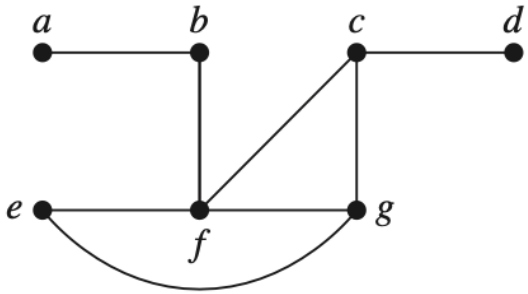
Let G be a simple graph. A spanning tree of G is a subgraph of G that is a tree containing every vertex of G.

Example

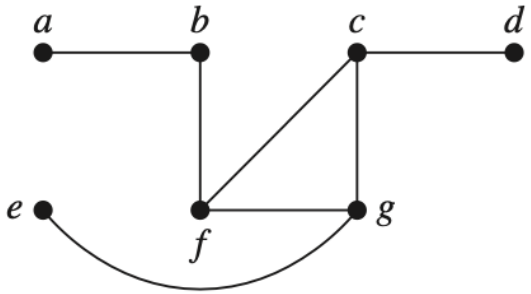
Find a spanning tree of G.



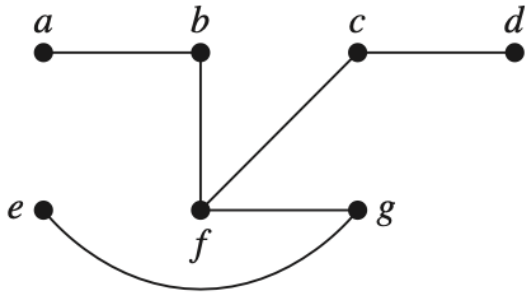
Solution. Removing edges that form simple circuits until no circuits exists.



Edge removed: $\{a, e\}$



$\{e, f\}$



$\{c, g\}$

Example (2)

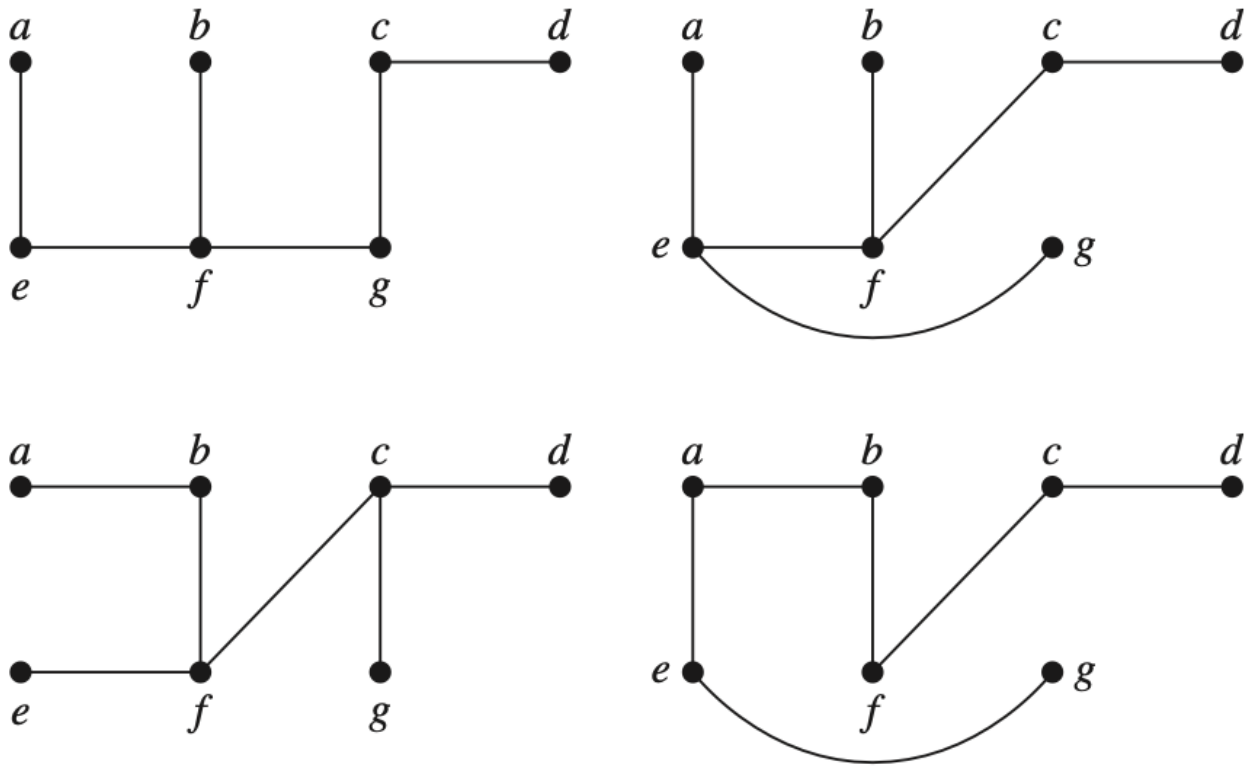


Figure: Spanning Trees of G

Producing a spanning tree

Theorem

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

- Theorem gives an algorithm for finding spanning trees by removing edges from simple circuits.
- This algorithm is inefficient, because it requires that simple circuits be identified.

Spanning trees can be built up by successively adding edges.
Two algorithms based on this principle:

- Depth-First Search (DFS)
- Breadth-First Search (BFS)

FPT UNIVERSITY

Depth-First Search

- Pick any vertex of the graph as the root and form a path starting at this vertex that is as long as feasible by successively adding vertices and edges such that each new edge is incident with the last vertex in the path and a vertex not already in the path.

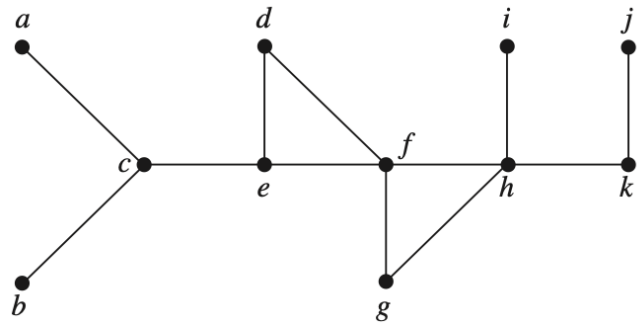
- If the path does not go through all vertices, move back to the next to last vertex in the path, and, if possible, form a new path starting at this vertex passing through vertices that were not already visited. If this cannot be done, move back another vertex in the path, that is, two vertices back in the path, and try again.

- Repeat this procedure, beginning at the last vertex visited, moving back up the path one vertex at a time, forming new paths that are as long as possible until no more edges can be added. This process ends with the production of a spanning tree.

FPT UNIVERSITY

Example: DFS

Use depth-first search to find a spanning tree for the graph G shown below.



Hint.

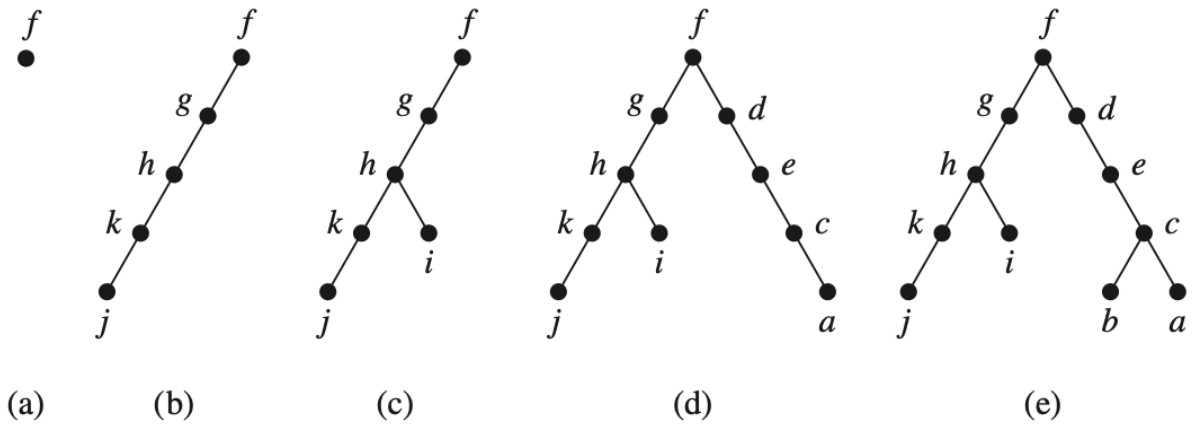
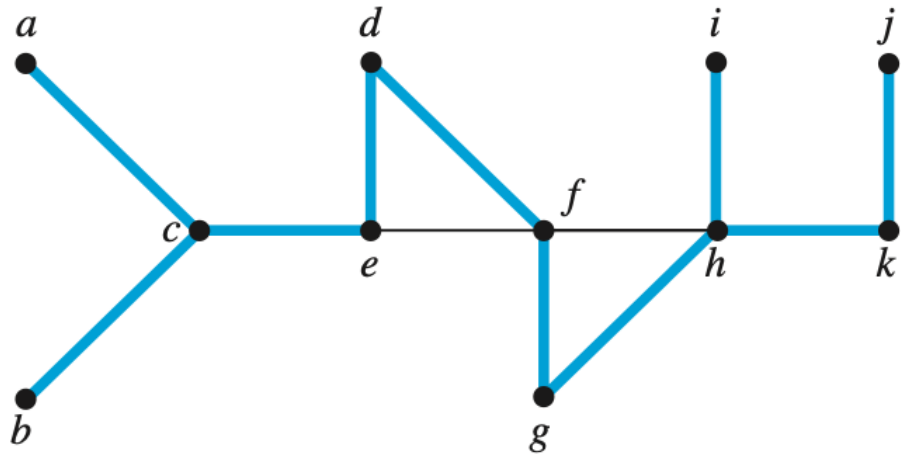


Figure: Depth-First Search of G

- The edges selected by depth-first search of a graph are called tree edges.
- All other edges of the graph must connect a vertex to an ancestor or descendant of this vertex in the tree. These edges are called back edges.

Example. The tree edges found by depth-first search starting at vertex f are highlighted with heavy colored lines. The back edges (e, f) and (f, h) are shown with thinner black lines.



Breadth-First Search

- Pick any vertex of the graph as the root and then add all edges incident to this vertex. The new vertices added at this stage become the vertices at level 1 in the spanning tree. Arbitrarily order them.

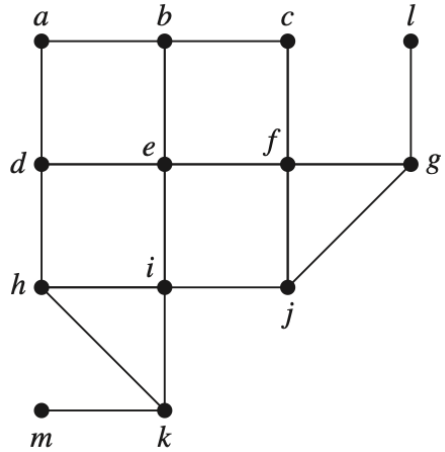
- For each vertex at level 1, visited in order, add each edge incident to this vertex to the tree as long as it does not produce a simple circuit. Arbitrarily order the children of each vertex at level 1. This produces the vertices at level 2 in the tree.

- Follow the same procedure until all the vertices in the tree have been added.

FPT UNIVERSITY

Example: BFS

Use breadth-first search to find a spanning tree for the graph shown below.



Hint.

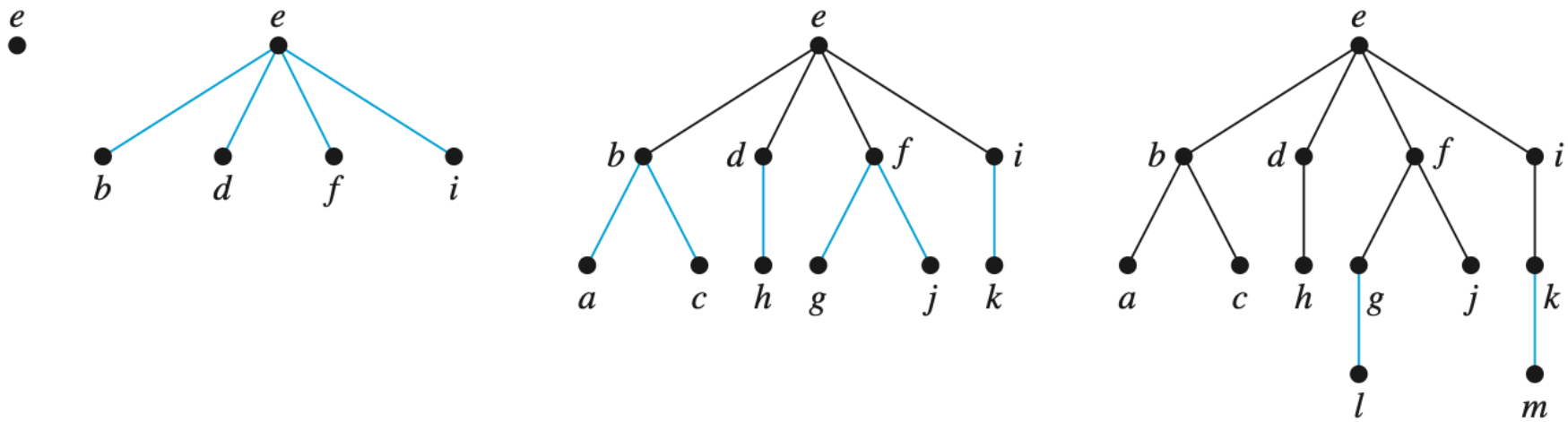


Figure: Breadth-First Search of G

Backtracking Applications

There are problems that can be solved only by performing an exhaustive search of all possible solutions. One way to search systematically for a solution is to use a decision tree, where each internal vertex represents a decision and each leaf a possible solution.

In the following, we will study some problems solved by backtracking.

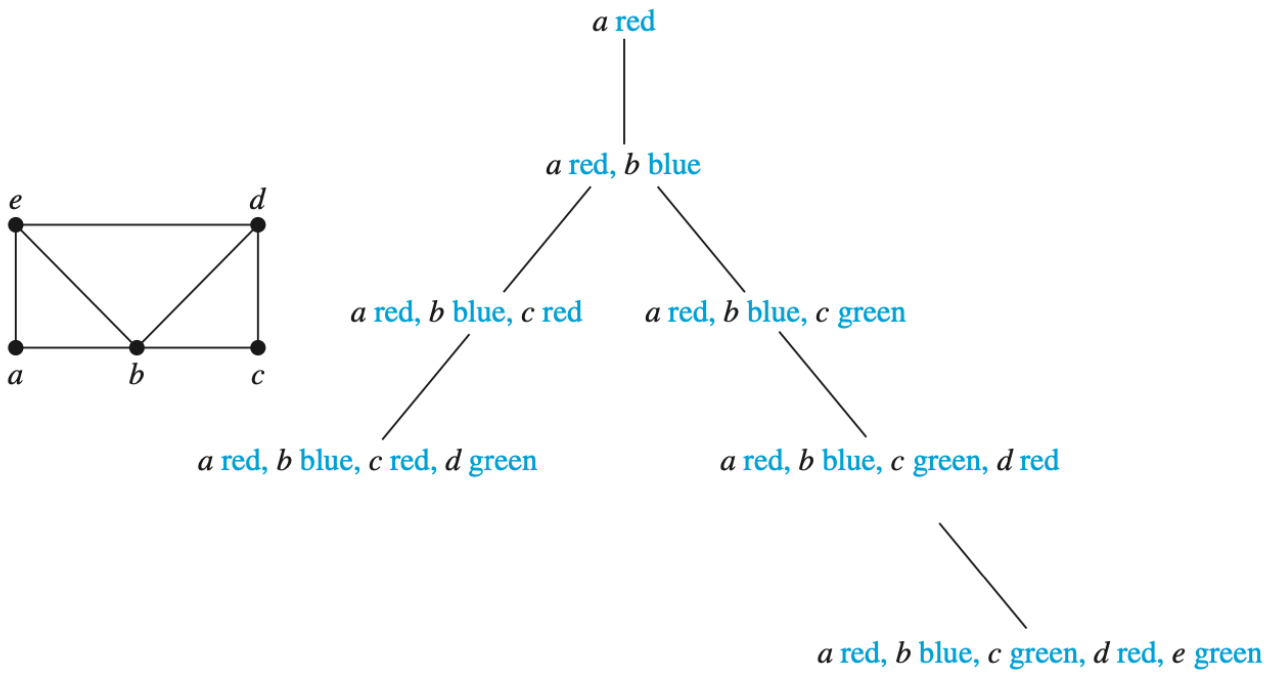
- Graph Colorings
- The n-Queens Problem
- Sums of Subsets

FPT UNIVERSITY

Graph Colorings

How can backtracking be used to decide whether a graph can be colored using n colors?

An example illustrates coloring the graph with 3 colors by using backtracking



The n-Queens Problem

The n-queens problem asks how n queens can be placed on an $n \times n$ chessboard so that no two queens can attack one another. How can backtracking be used to solve the n-queens problem?

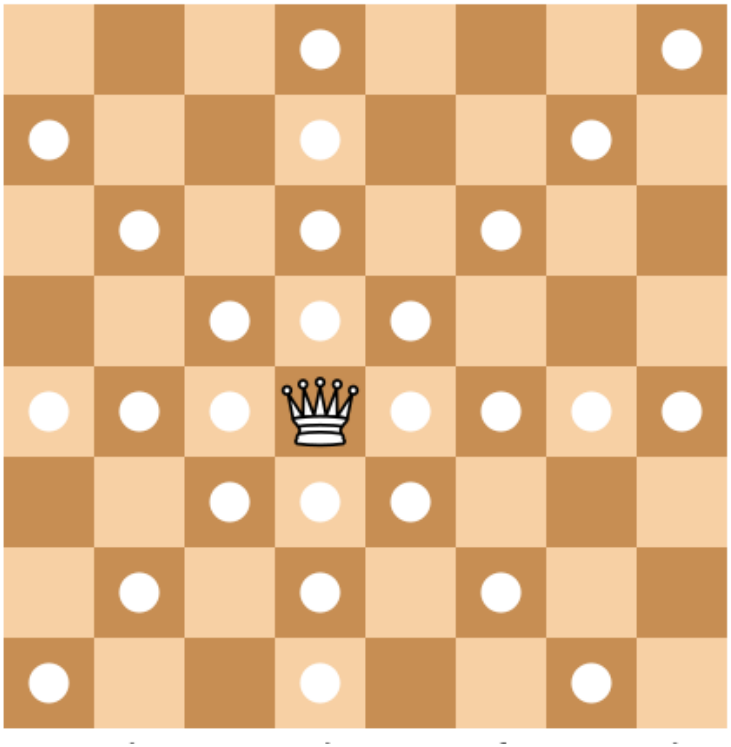
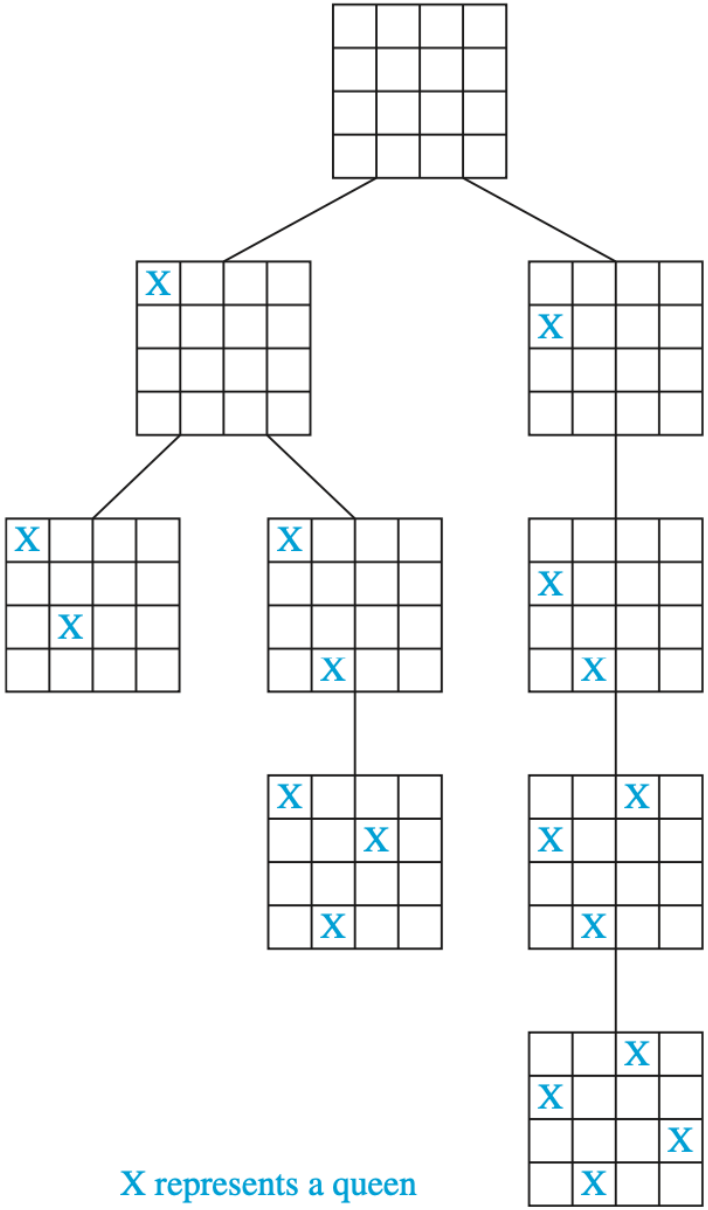


Figure: The squares controlled by a Queen

A backtracking solution to the four-queens problem



Sums of Subsets

Given a set of positive integers $x_1, x_2, \ldots, x_n$, find a subset of this set of integers that has M as its sum. How can backtracking be used to solve this problem?

Example. The following figure displays a backtracking solution to the problem of finding a subset of $\{31, 27, 15, 11, 7, 5\}$ with the sum equal to 39.

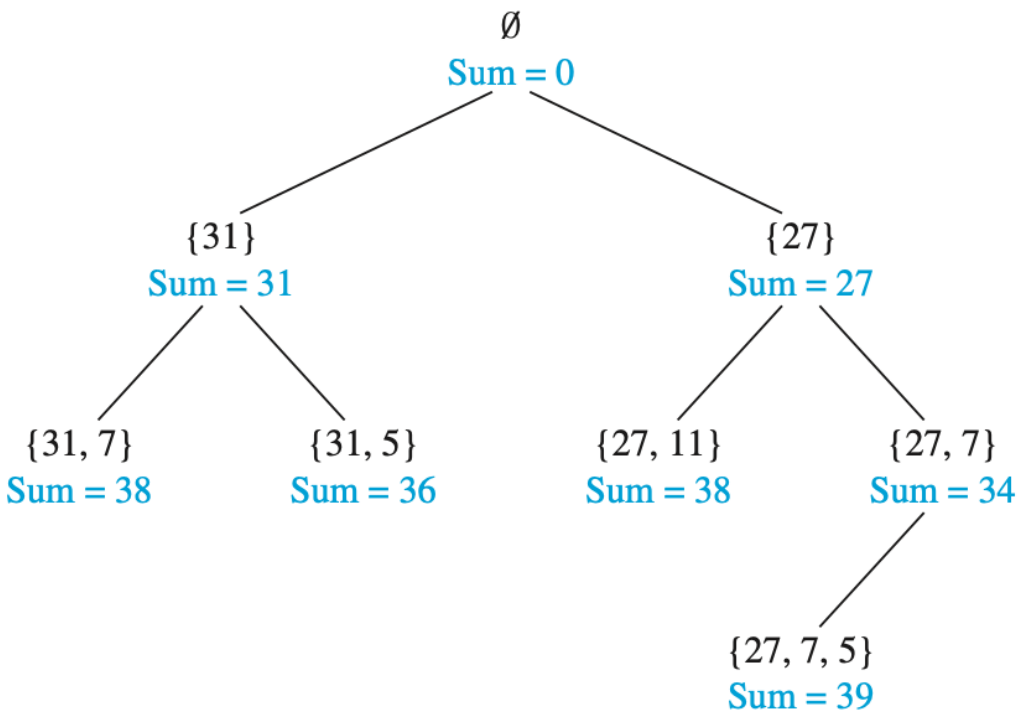
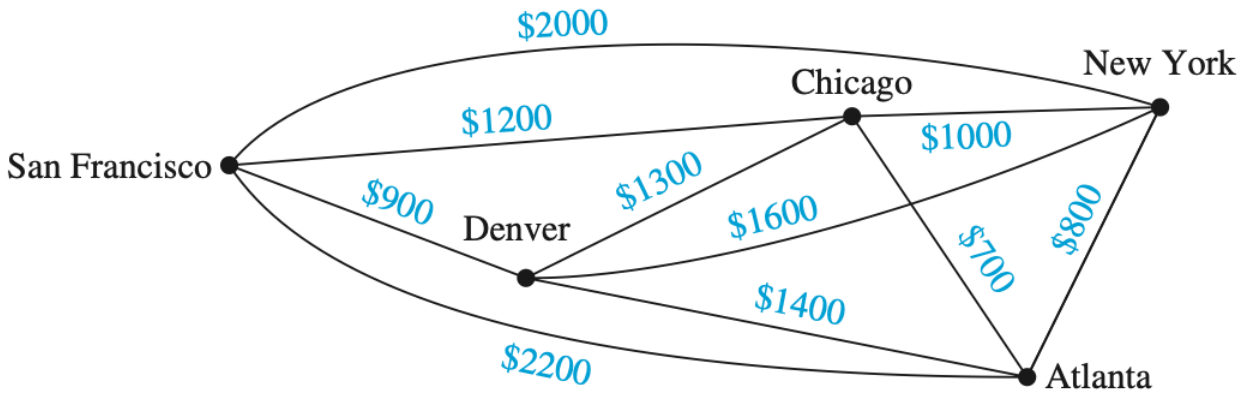


Table of Contents

FPT UNIVERSITY

Minimum Spanning Trees

Problem. A company plans to build a communications network connecting its five computer centers. Any pair of these centers can be linked with a leased telephone line. 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.

Minimum Spanning Trees (2)

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

Two universal algorithms for constructing minimum spanning trees:

- **Prim's algorithm** originally discovered by the Czech mathematician Vojtěch Jarník in 1930 and rediscovered in 1957 by Robert Prim
- **Kruskal's algorithm** discovered by Joseph Kruskal in 1956

FPT UNIVERSITY

Prim's Algorithm

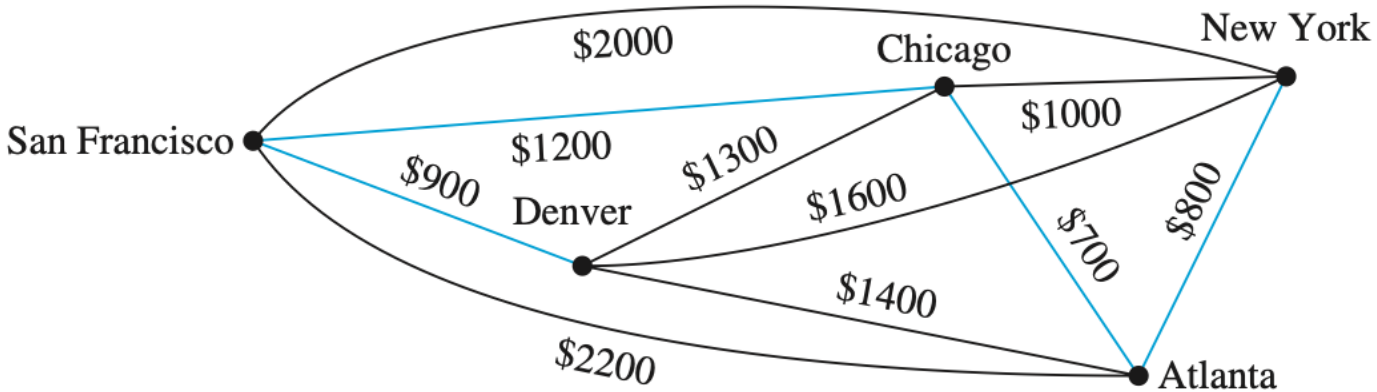
procedure Prim(G: weighted connected undirected graph with n vertices)
$T :=$ a minimum-weight edge
for $i := 1$ **to** $n - 2$
  $e :=$ an edge of minimum weight incident to a vertex in T
  and not forming a simple circuit in T if added to T
  $T := T$ with e added
return T {T is a minimum spanning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

Note. The choice of an edge to add at a stage of the algorithm is not determined when there is more than one edge with the same weight that satisfies the appropriate criteria. We need to order the edges to make the choices deterministic.

Example: Prim's Algorithm

Use Prim's algorithm to design a minimum-cost communications network connecting all the computers presented at the beginning of Section Minimum Spanning Trees.

Solution.



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

Kruskal's Algorithm

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}

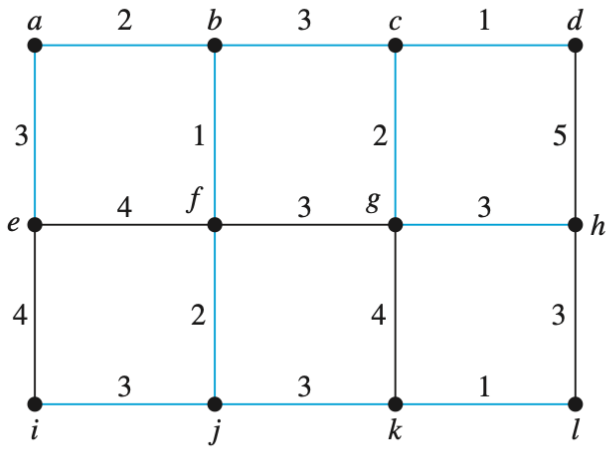
Note. The difference between Prims and Kruskals algorithms:

- Prim's algorithm: **edges** of minimum weight that are **incident to a vertex already in the tree**, and not forming a circuit, are chosen
- 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.

Example: Kruskal's Algorithm

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

Solution.



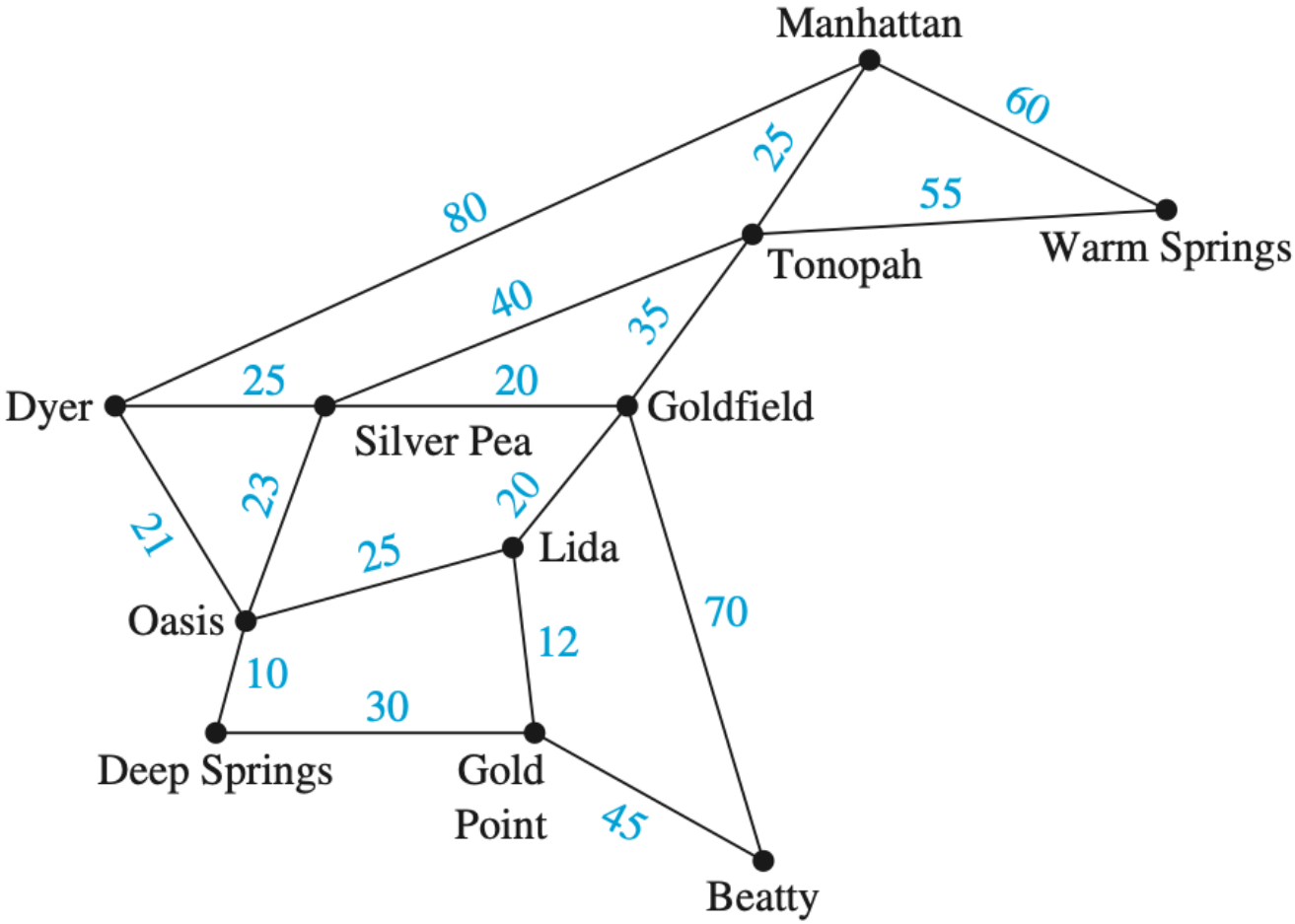
(a)

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

(b)

Figure: A Minimum Spanning Tree Produced by Kruskals Algorithm

Problem. The roads are all unpaved whose lengths 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.)



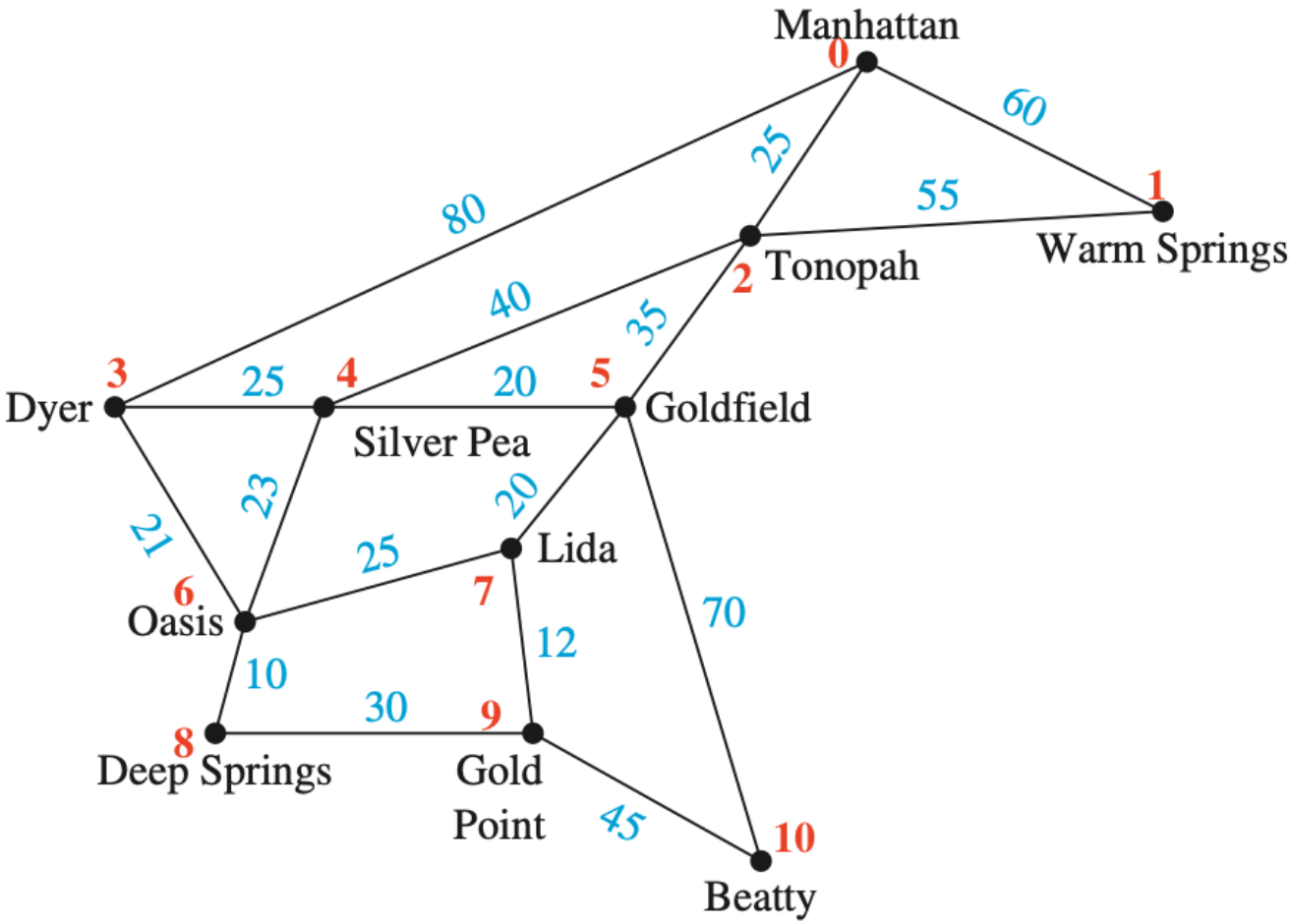
To implement Prim's & Kruskal's Algorithms in Python language, we need

1. Label vertices of the graph
2. Create graph by adjacency matrix method
3. Input number of the vertices and adjacency matrix into the code
4. Run code and read the output.

Visit https://www.programiz.com/dsa/prim-algorithm and https://www.programiz.com/dsa/kruskal-algorithm for the codes.

FPT UNIVERSITY

Return to Problem, we label towns and create the adjacency matrix as follows:



$$A = \begin{pmatrix} 0 & 60 & 25 & 80 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 60 & 0 & 55 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 25 & 55 & 0 & 0 & 40 & 35 & 0 & 0 & 0 & 0 & 0 \\ 80 & 0 & 0 & 0 & 25 & 0 & 21 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40 & 25 & 0 & 20 & 23 & 0 & 0 & 0 & 0 \\ 0 & 0 & 35 & 0 & 20 & 0 & 0 & 20 & 0 & 0 & 70 \\ 0 & 0 & 0 & 21 & 23 & 0 & 0 & 25 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 20 & 25 & 0 & 0 & 12 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 0 & 30 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12 & 30 & 0 & 45 \\ 0 & 0 & 0 & 0 & 0 & 70 & 0 & 0 & 0 & 45 & 0 \end{pmatrix}$$

In which, A_{ij} = weight of the edge $\{i-1, j-1\}$.

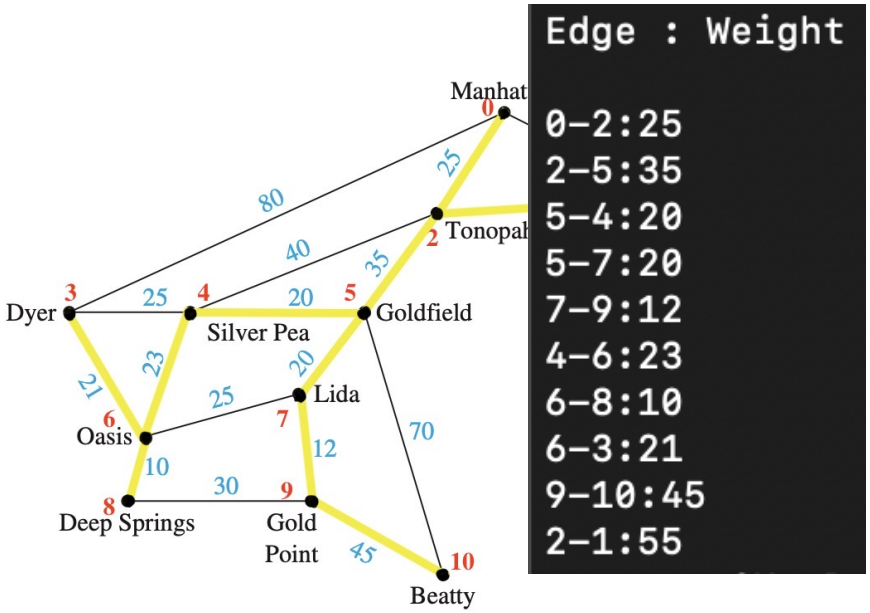


Figure: Roads should be paved marked in yellow and the output of Prim's Algorithm in Python.

1. Introduction to Trees
2. Applications of Trees
3. Tree Traversal
4. Spanning Trees
5. Minimum Spanning Trees

Thank you!

FPT UNIVERSITY