樹 Trees

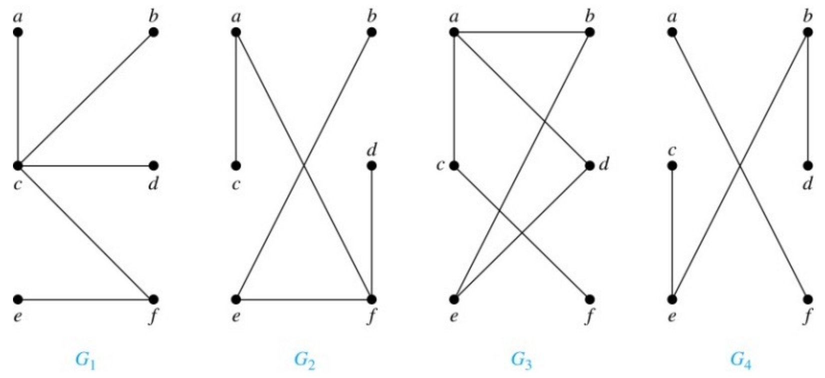
第十一章

Outline

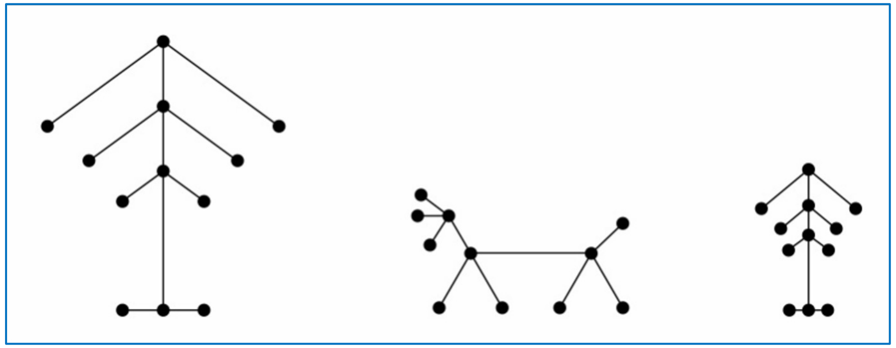
- 樹 Trees
- 樹的遍歷 Tree Traversal
- 生成樹 Spanning Trees

+ 樹 Trees

- 樹(Tree) : 沒有簡單回路(simple circuit)的連通無向圖(connected undirected graph)。
  - 例如:
    - G_1 , G_2 are trees;
    - G_3 , G_4 are not trees.



- 森林(forest) : 沒有簡單回路但不連通的圖，其每個連通分支皆為樹。
  - 例如:
    - H is a forest with 3 connected components.



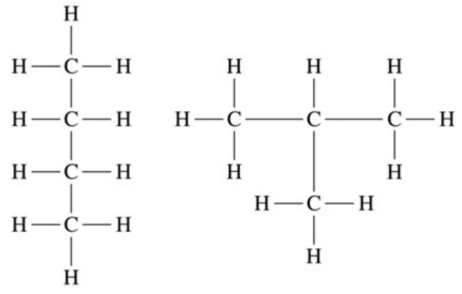
H

樹 Trees

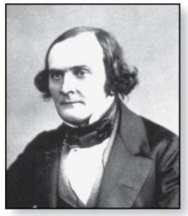
- (11.1定理1:) 一個無向圖是樹當且僅當在它的每對頂點之間存在唯一簡單通路。

+ 以樹作為模型 Trees as Models

- 以樹為模型的應用領域十分廣泛，如計算機科學、化學、地理學、植物學和心理學等。
  - 英國數學家 Arthur Cayley 在 1857 於他的研究中利用樹列舉了化合物的同分異構體並計數 (丁烷 vs. 異丁烷)

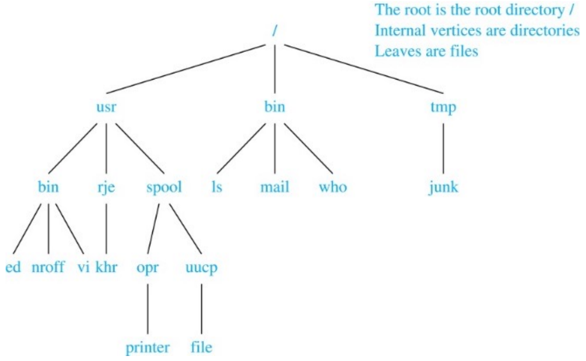


Butane　　Isobutane

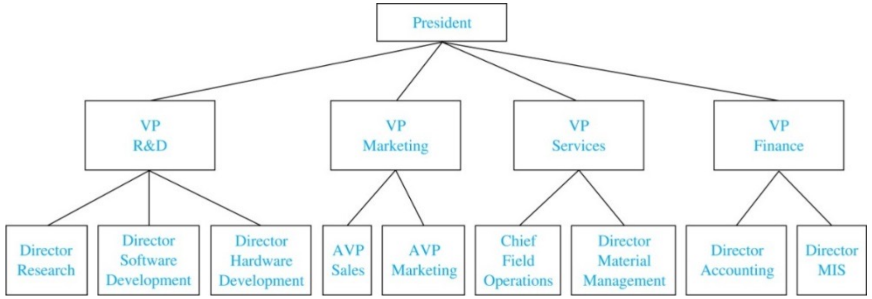


Arthur Cayley
(1821-1895)

  - 計算機存儲器中的文件目錄



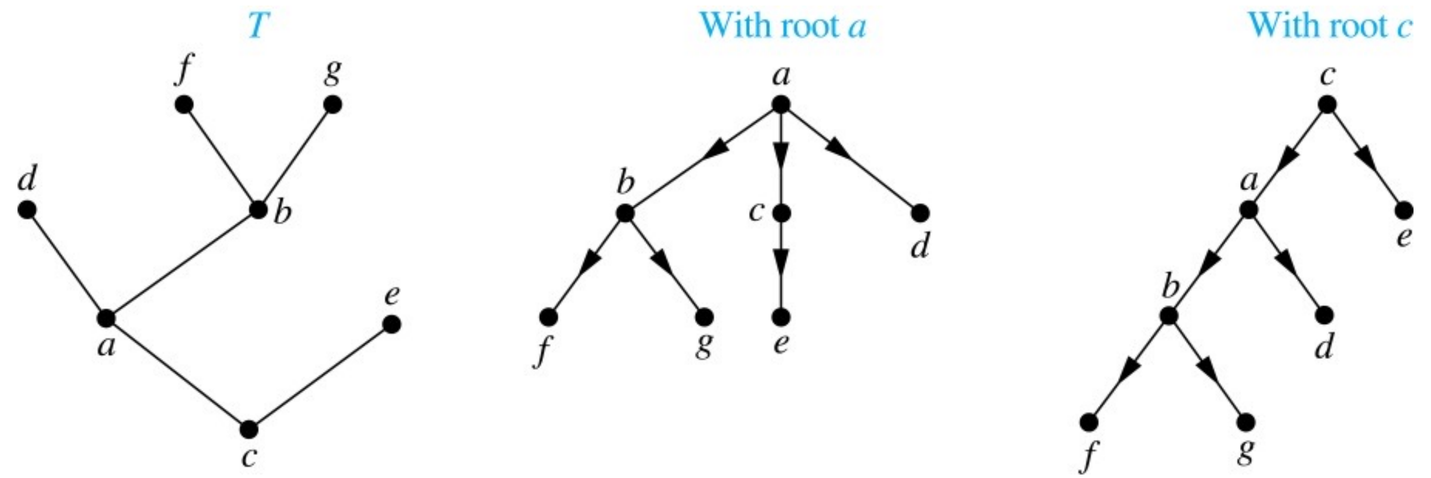
  - 組織架構



有根樹 Rooted Trees

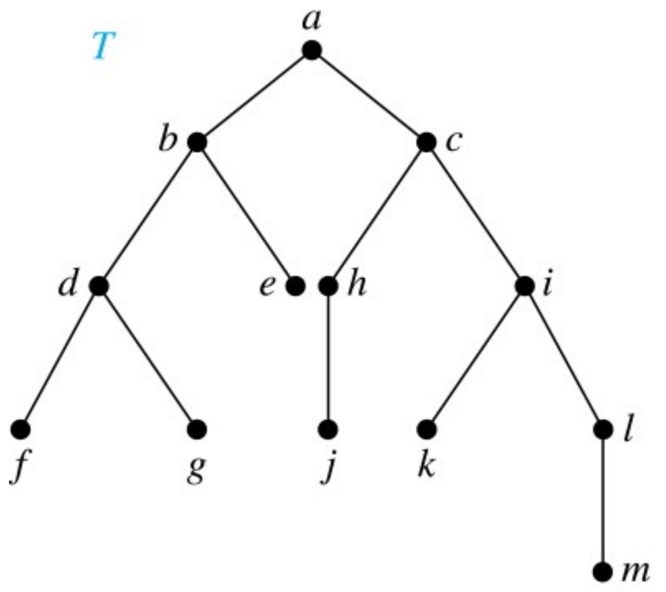
- 有根樹(Rooted Tree)：以一個頂點作為根(root)且每條邊的方向都離開(directed away)根的樹。

- 非有根樹(unrooted tree)可通過選取不同的根變成不同的有根樹(rooted tree)。



樹的術語 Tree Terminologies

- 對於有根樹 $T = (V, E).\, u, v \in V$, 其中 v 不是根：
  - v 的父母(Parent): 以 v 為終點的有向邊的起點
  - u 的孩子(Child): 以 u 為起點的有向邊的終點
  - v 的兄弟(Siblings): 具有相同父母的頂點
  - v 的祖先(Ancestors): 從根到 v 的通路上除 v 外的頂點
  - u 的後代(Descendants): 以 u 作為祖先的頂點
  - 樹葉(Leaf): 沒有孩子的頂點
  - 內點(Internal Vertices): 有孩子的頂點
  - 以 u 為根的子樹(Subtree with u): 由 u 和 u 的後代及這些頂點所關聯的邊所組成的該樹的子圖。



例1

A. 已知 $T=(V, E)$ 為以 a 為根的有根樹 (rooted tree) , 求:

(i) c 的父母(parent) b

(ii) g 的孩子(child) h,i,j

(iii) h 的兄弟(siblings) i,j

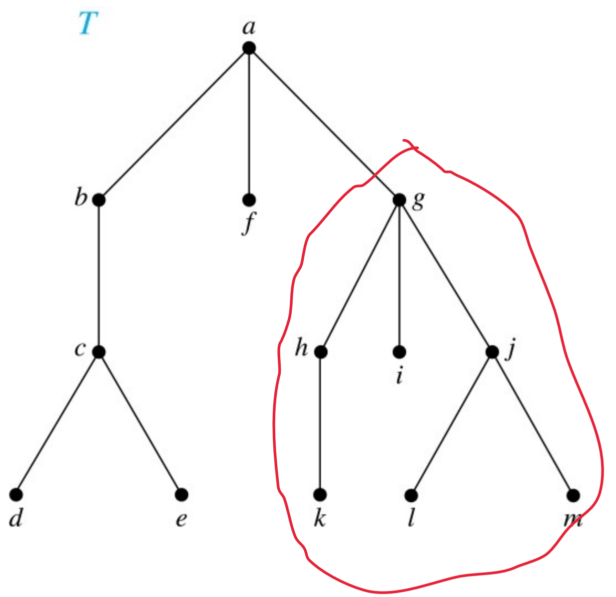
(iv) e 的祖先(ancestors) c,b,a

(v) b 的後代(descendants) c,d,e

(vi) 所有內點 (internal vertices) a,b,c,g,h,j

(vii) 所有樹葉(leaves) d,e,f,i,k,l,m

(viii) 以 g 為根的子樹(subtree)

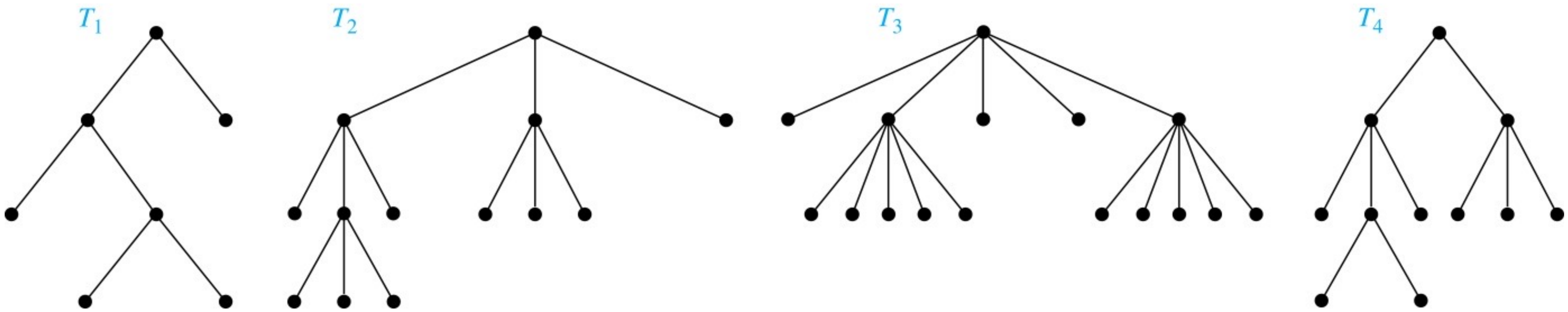


m叉樹

- m 叉樹 (m-ary tree) : 有根樹的每個內點都有不超過 m 個孩子。
  - 滿 m 叉樹 (full m-ary tree) : 該樹的每個內點都恰好有 m 個孩子。
  - 二叉樹 (binary tree): $m = 2$

+ 例2

- 下列何者為滿 m 叉樹 (full m-ary tree)?

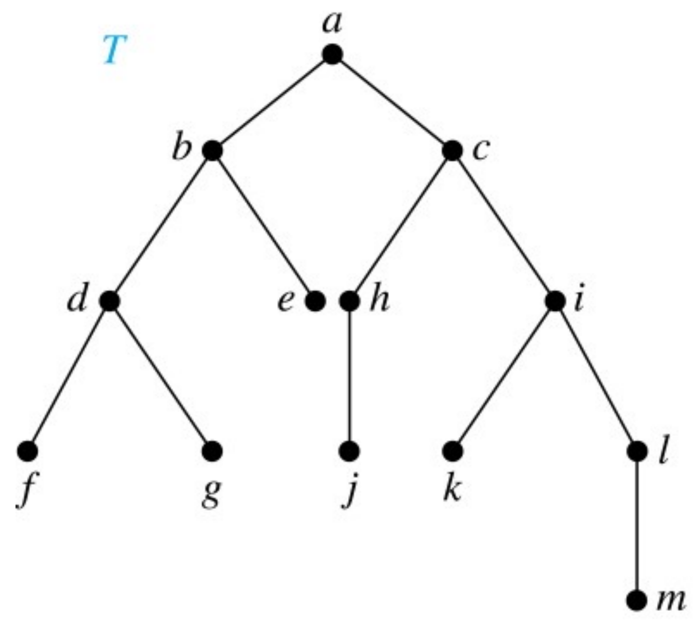


有序根樹 Ordered Rooted Trees

- 有序根樹(ordered rooted tree) : 把每個內點都排序的有根樹。
  - 一般在畫圖時會按順序進行
- 在二叉樹中:
  - 左子(left child): 內點的第一個孩子
  - 右子(right child): 內點的第二個孩子
  - 左子樹(left subtree): 以內點的左子為根的子樹
  - 右子樹(right subtree):以內點的右子為根的子樹

例3

- 已知 $T = (V, E)$ 為二叉樹(binary tree):
  A. 求d的左子(left child)和右子(right child);
  B. 求c的左子樹(left subtree)和右子樹(right subtree).



+ 樹的性質 Properties of Trees

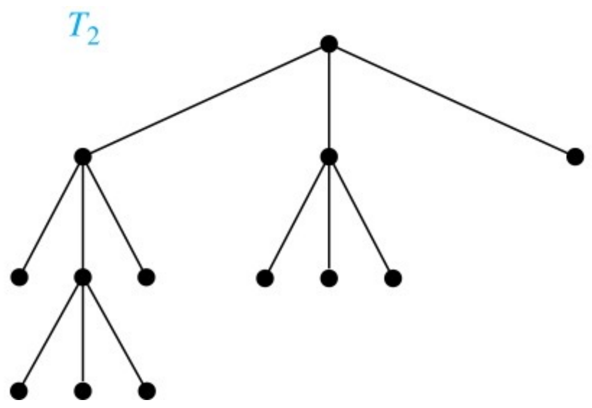
- (11.1定理2:) 帶有 n 個頂點的樹含有 $n-1$條邊。

例4

- 已知樹 $T_1 = (V_1, E_1), T_2 = (V_2, E_2)$, 且$|E_1| = 11$, $|V_2| = 2|V_1|$. 求: a) $|V_1|$; b) $|V_2|$; c) $|E_2|$。

計算滿 m 叉樹的頂點數

- (11.1定理3:) 帶有 i 個內點的滿 m 叉樹含有 $n = mi + 1$ 個頂點。
- (11.1定理4:) 給定一個滿 m 叉樹 $T = (V, E)$,
  其中$|V| = n$, l 為樹葉數和 i 為內點數，則:
  - $n = mi + 1 = \frac{ml-1}{m-1}$
  - $l = (m-1)i + 1$
  - $i = \frac{l-1}{m-1} = \frac{n-1}{m}$



+ 例5

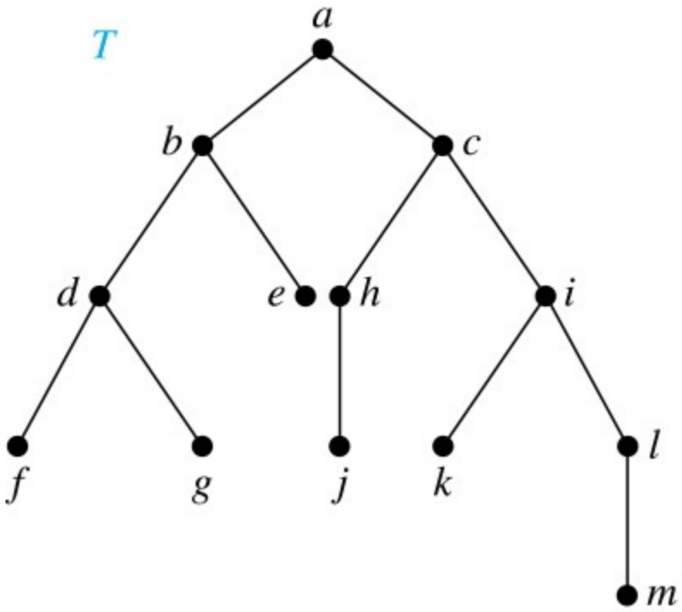
- 甲寄出一封連環信, 要求收到信的每個人再把信寄給另外 4 個人。有些人按要求做了, 但有些人沒有寄信，若沒有人收到多於一封信，現已知已看過信但沒有寄信的有100人且連環信已終止，問包括甲在內一供有多少人:

A. 看過信?

B. 寄過信?

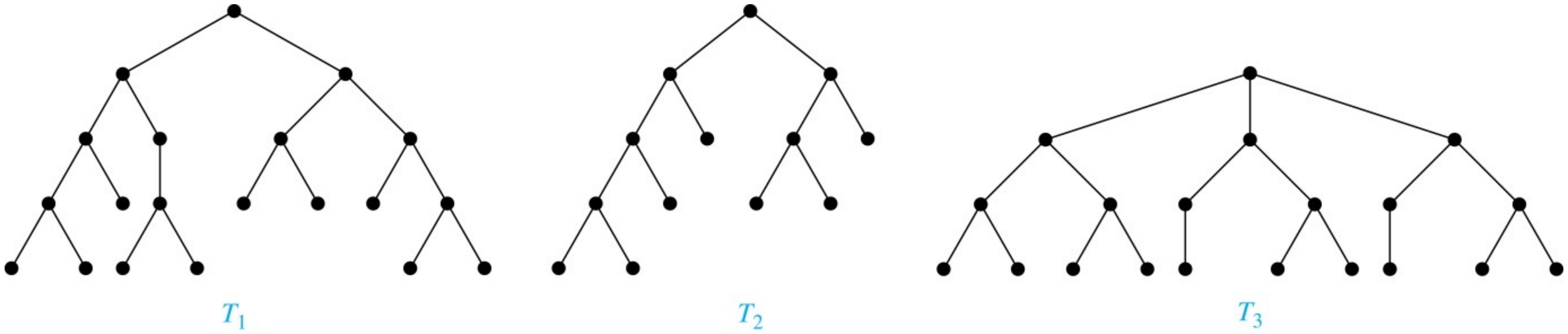
+ 頂點的層與高

- 頂點 v 的層(level): 從根至這個頂點的唯一通路的長度。
- 有根樹的高度(height): 頂點層數的最大值。(從根到任意點的最長通路的長度)



平衡的 m 叉樹

- 高度為 h 的有根 m 叉樹若所有樹葉都在 h 層或 $h-1$ 層稱該樹為平衡的(balanced)。
  - 例如:
    - T_1 and T_3 – balanced
    - T_2 – not balanced



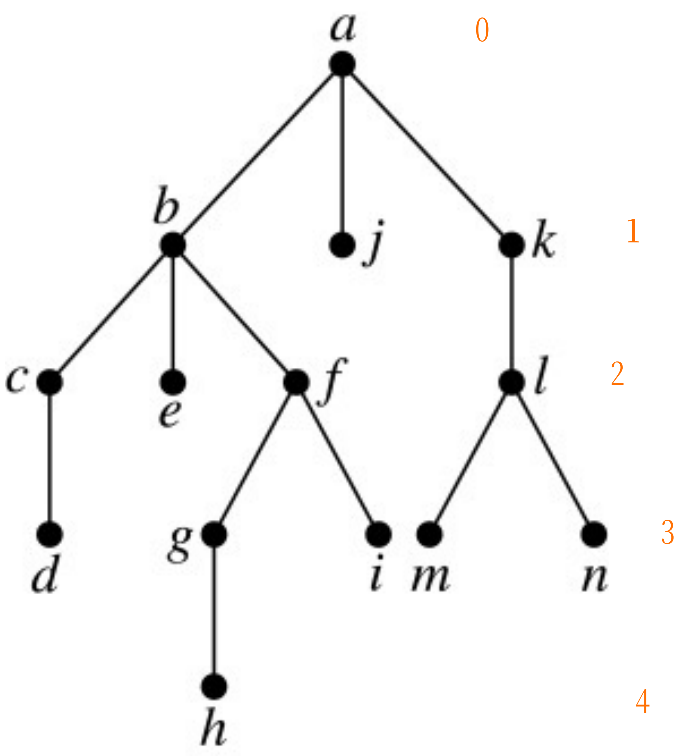
例6

- 已知右圖為以 a 為根的有根樹 T：

A. 求 T 所有頂點的層數(level);

B. T 的高度(height)是多少? 4

C. T 是否平衡? 不平衡



m叉樹中的樹葉數

- (11.1定理5:) 高度為 h 的 m 叉樹中的樹葉數至多為m^h。

- (11.1推論1:) 高度為 h 的 m 叉樹 T 若有 l 個樹葉，則

$$h \geq \lceil log_m l \rceil$$

向上取整

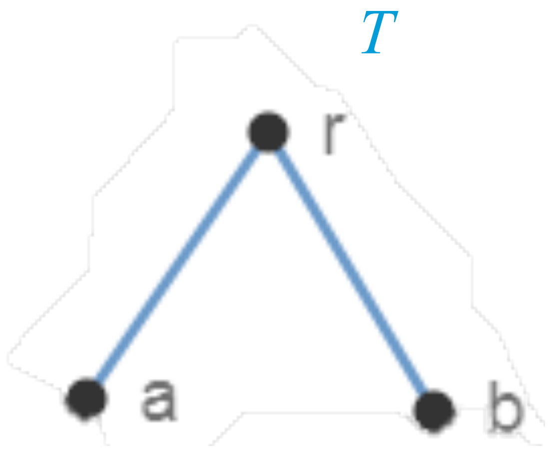
  - 若 T 為滿的且平衡的，則

$$h = \lceil log_m l \rceil.$$

樹的遍歷 Tree Traversal (11.3)

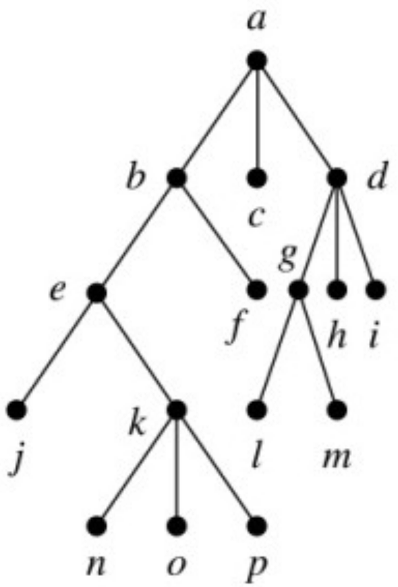
- 遍歷算法(Traversal algorithms)：系統地訪問有序根樹每個頂點的過程。

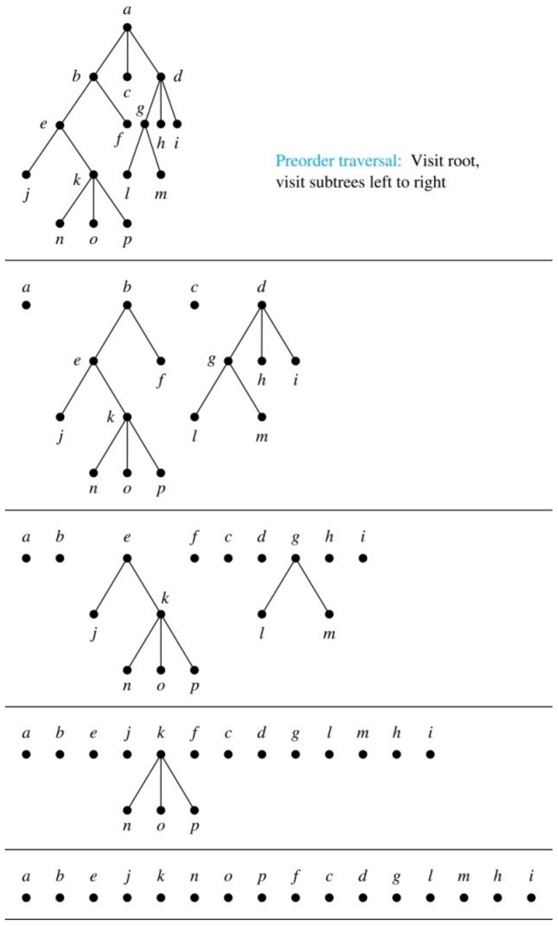
- 最常用的三種遍歷算法:
  - 前序遍歷 (Preorder traversal) r a b
  - 中序遍歷 (Inorder traversal) a r b
  - 後序遍歷 (Postorder traversal) a b r



Preorder Traversal 前序遍歷

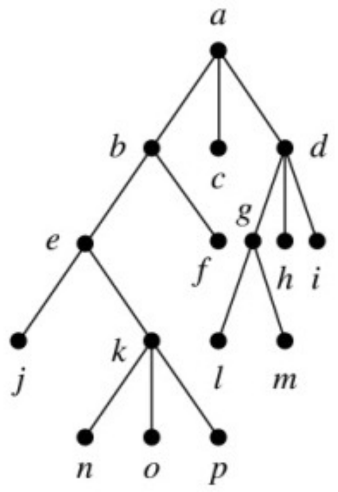
procedure *preorder* (*T*: ordered rooted tree)
r := root of *T*
list *r*
for each child *c* of *r* from left to right
    T(*c*) := subtree with *c* as root
    preorder(*T*(*c*))

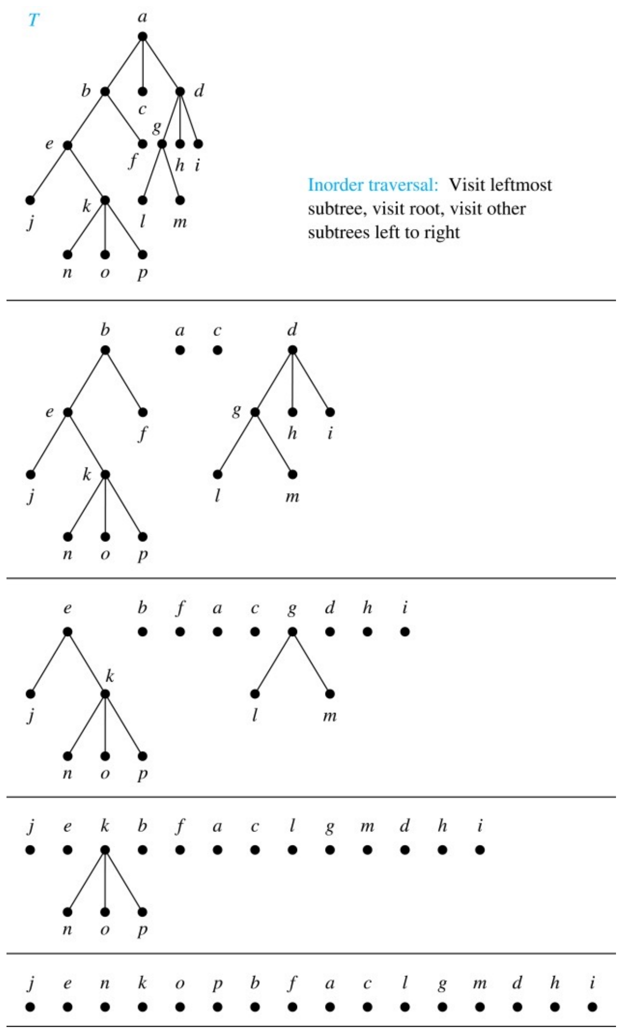




+ Inorder Traversal 中序遍歷

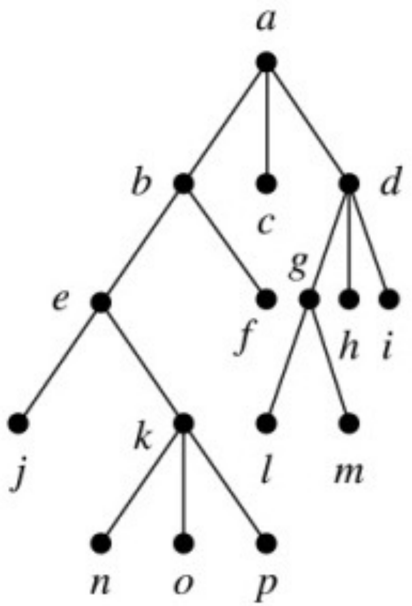
procedure *inorder* (T: ordered rooted tree)
r := root of T
if r is a leaf **then** list r
else
  l := first child of r from left to right
  $T(l)$:= subtree with l as its root
  inorder($T(l)$)
  list(r)
  for each child c of r from left to right
    $T(c)$:= subtree with c as root
    inorder($T(c)$)

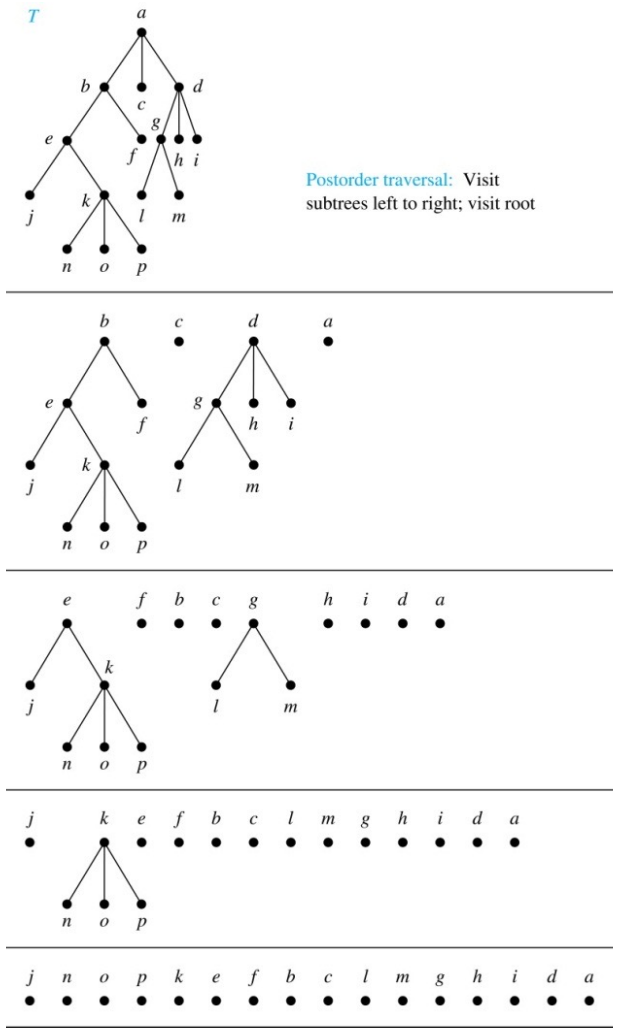




+ Postorder Traversal 後序遍歷

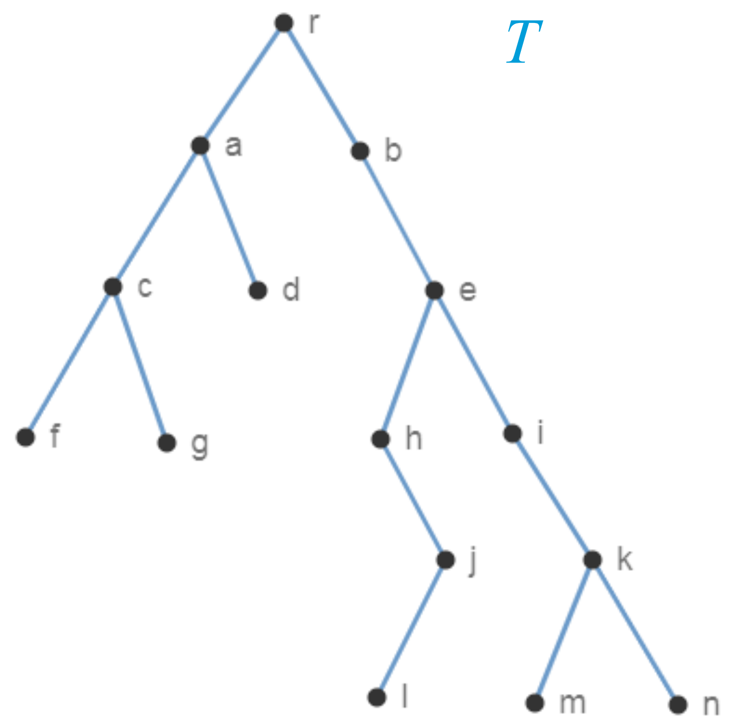
procedure *postordered* (T: ordered rooted tree)
r := root of T
for each child c of r from left to right
  $T(c)$:= subtree with c as root
  postorder($T(c)$)
list r





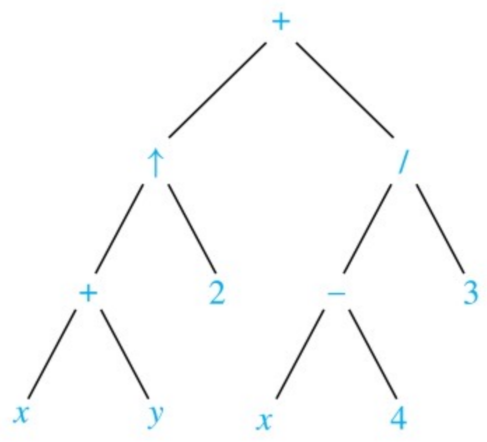
例7

- 求下列遍歷算法訪問有序根樹(ordered rooted tree) T 各頂點的順序:
  A. 前序遍歷 (Preorder traversal) racfgdbehjlikmn
  B. 中序遍歷 (Inorder traversal) fcgadrbhljeimkn
  C. 後序遍歷 (Postorder traversal) fgcdaljhmnkiebr



+ Expression Trees

- 表達式可以有序根樹(ordered rooted tree)表示:
  - 內點表示運算，分別為: + (加)、- (減)、*(乘)、/(除)、↑(冪)
  - 樹葉表示變量或數字
- 表達式: $(x+y)^2+\frac{x-4}{3}$ → $((x+y)\uparrow 2)+((x-4)/3)$
- 對應二叉樹(binary tree):



Infix Notation 前綴形式

- 前綴形式 (Prefix form) – 以前綴遍歷算法所得順序
- 中綴形式 (Infix form) – 以中綴遍歷算法所得順序
- 後綴形式 (Postfix form) – 以後綴遍歷算法所得順序
- 例如: $(x+y)*z$
  - Prefix form: $*+\ x\ y\ z$
  - Infix form: $(x+y)*z$
  - Postfix form: $x\ y+z\ *$

例8

- 寫出表達式 $t+\frac{uv}{w+(x-y^z)}$ 的:
  A. 前綴形式 (Prefix form)
  B. 後綴形式 (Postfix form)

例9

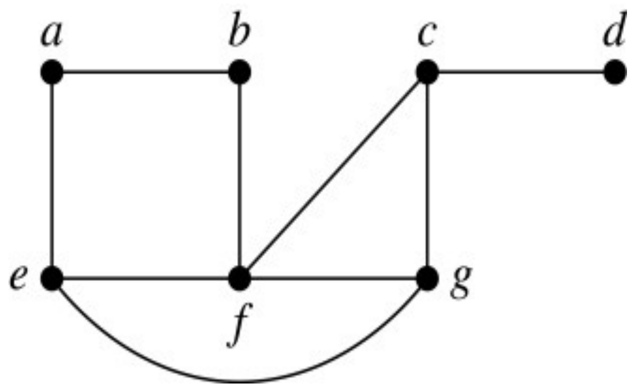
求下列表達式的值:

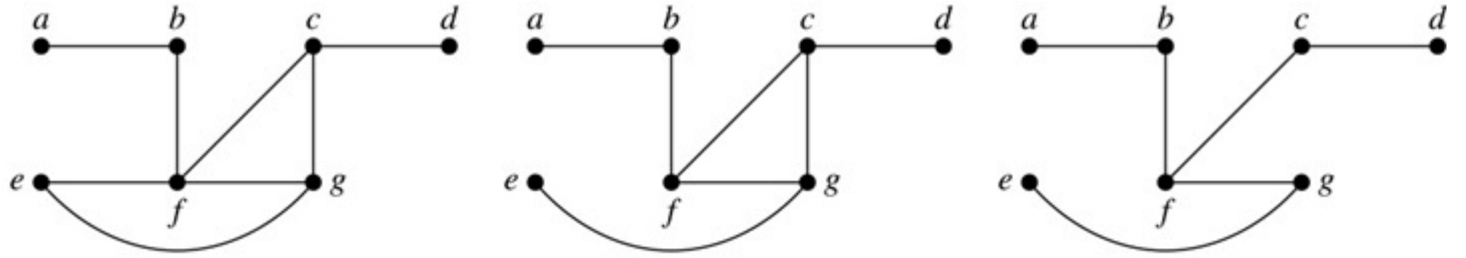
A. 前綴表達式 : $+ + 4 * 3\ 4\ +\ 5 / 6\ 3$

B. 後綴表達式 : $9\ 3 / 6 + 8\ 3 - *$

+ Spanning Trees 生成樹 (11.4)

- 已知 $G = (V, E)$為簡單圖，G 的生成樹 (Spanning Tree of G) $T = (V', E')$為包含 G 的子圖，其中 $V' = V$。
  - 例如:





Edge removed: $\{a, e\}$ $\{e, f\}$ $\{c, g\}$

(a) (b) (c)

+ Spanning Trees 生成樹

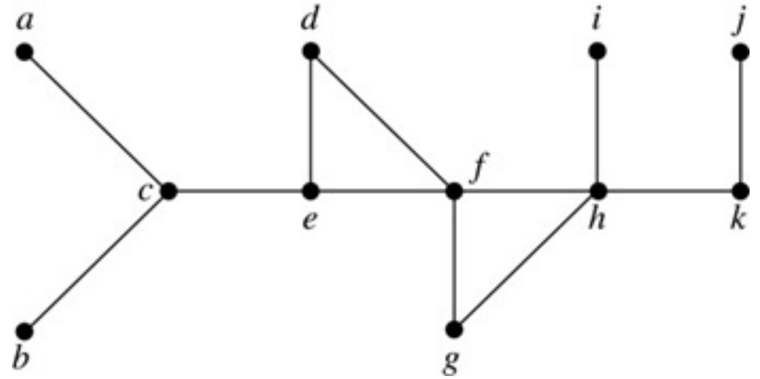
- (11.4定理1:) 簡單圖是連通的(connected)當且僅當它有生成樹(spanning tree)。

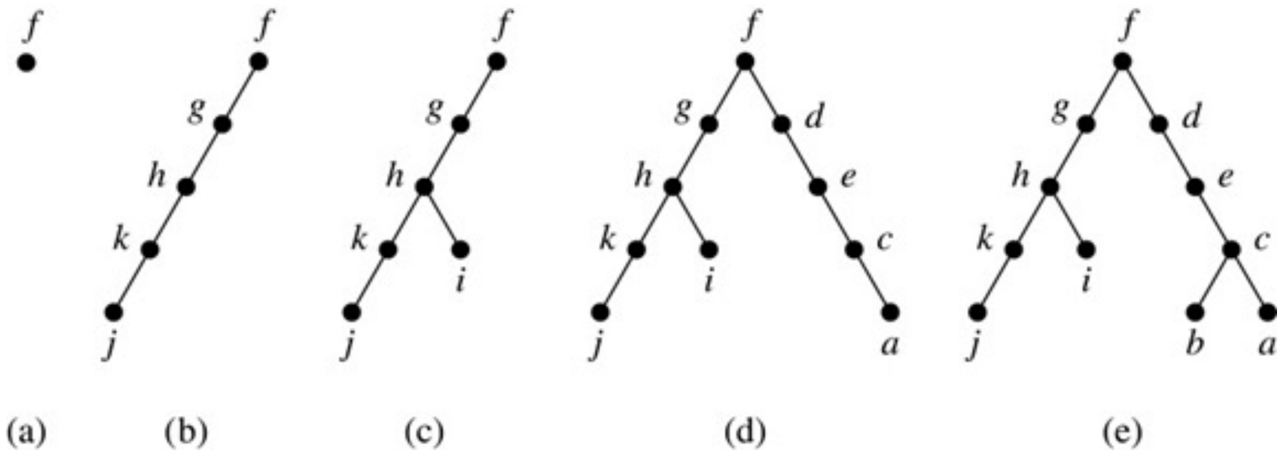
+ Depth-First Search (DFS) 深度優先搜索

- 用以產生連通簡單圖的生成樹

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)

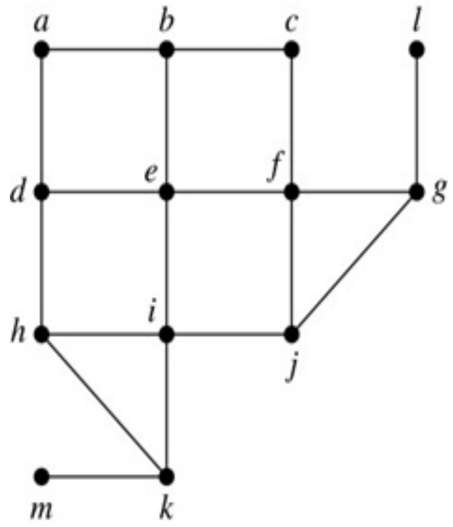


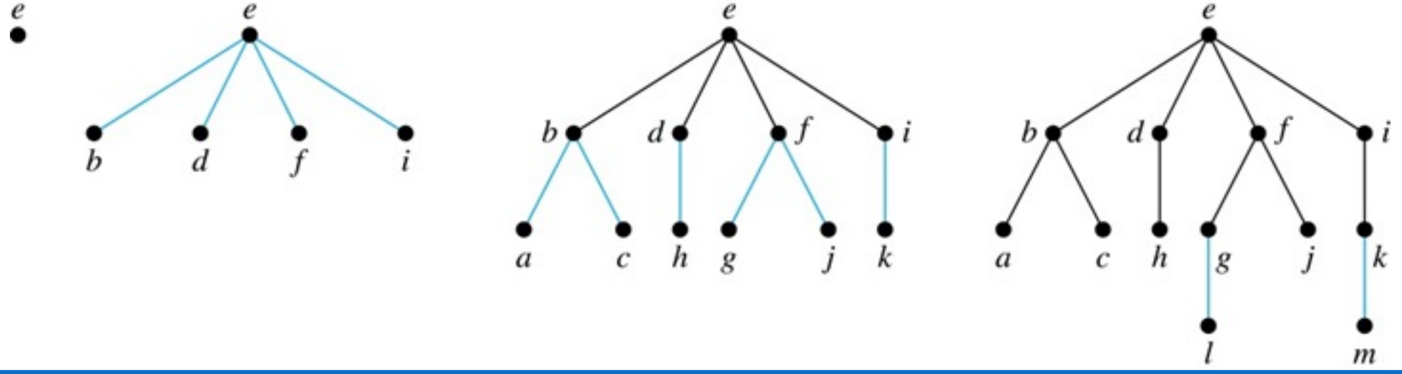


+ Breadth-First Search (BFS) 寬度優先搜索

- 用以產生連通簡單圖的生成樹

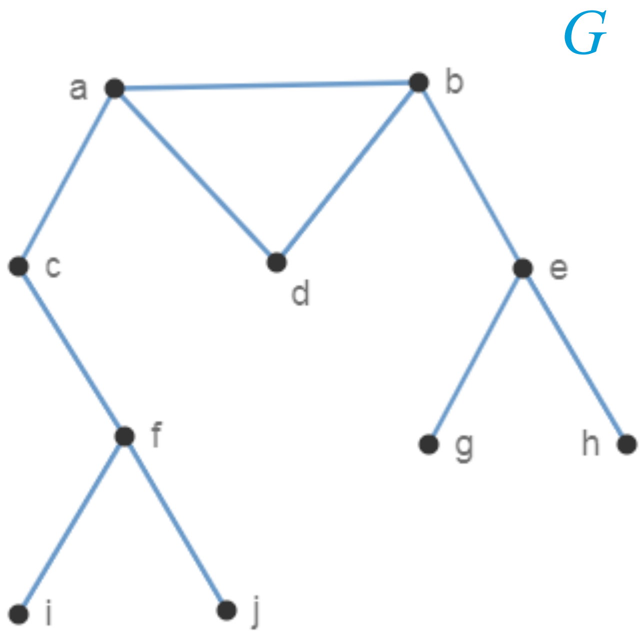
procedure *BFS*(G: connected graph with vertices $v_1, v_2, \ldots, v_n$)
T := tree consisting only of the vertex v_1
L := empty list *visit*(v_1)
put v_1 in the list L of unprocessed vertices
while L is not empty
  remove the first vertex, v, from L
  for each neighbor w of v
    if w is not in L and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





例4

- 以 a 為根分別用 DFS 和 BFS 求圖 G 的生成樹。



教材對應閱讀章節及練習

- 11.1(~Example 11), 11.2, 11.3(Traversal Algorithms~), 11.4(~Example 5)
- 對應習題: (可視個人情況定量)
  - 11.1: 1-10, 17-23
  - 11.3: 7-24
  - 11.4: 2-10, 13-18