蘇州大學
SOOCHOW UNIVERSITY

Chap11 Trees

Jin-Hui Wu

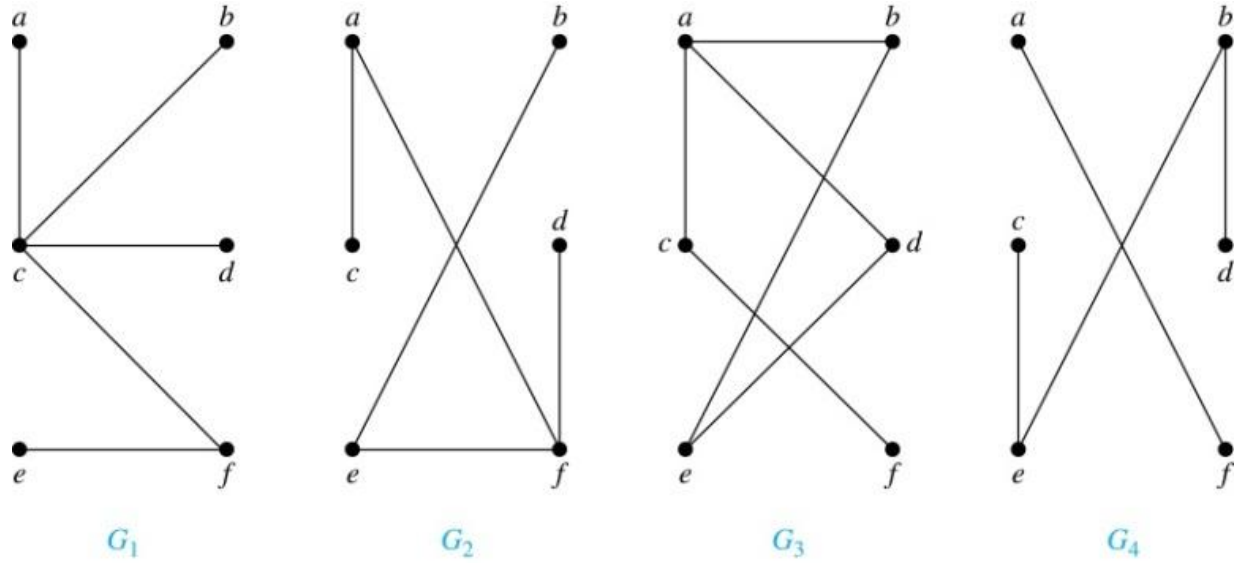
2026-04-24

大纲

- 树的基本概念 (11.1)
- 树的应用
- 树的遍历
- 生成树
- 最小生成树

树

- 树 (tree)
  - 树是没有简单回路的连通无向图
    - 无向、连通、无简单回路

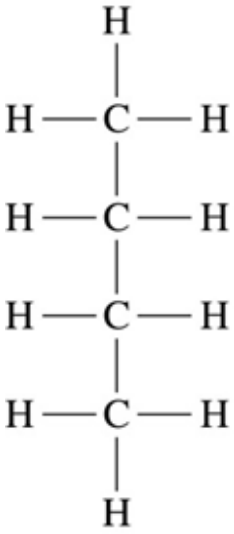


树

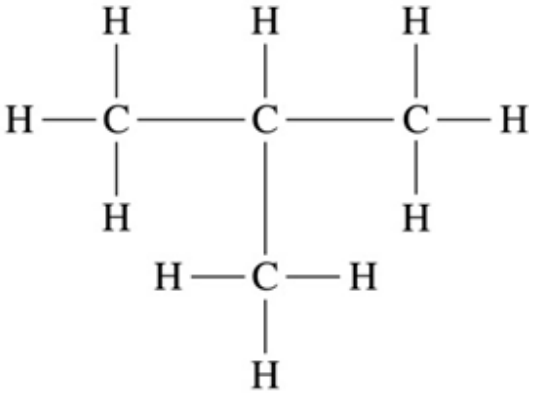
- 树 (tree)
  - 树是没有简单回路的连通无向图
    - 无向、连通、无简单回路
  - 无向图是树当且仅当任意两点间有唯一简单通路

树

- 树作为模型
  - 碳氢化合物



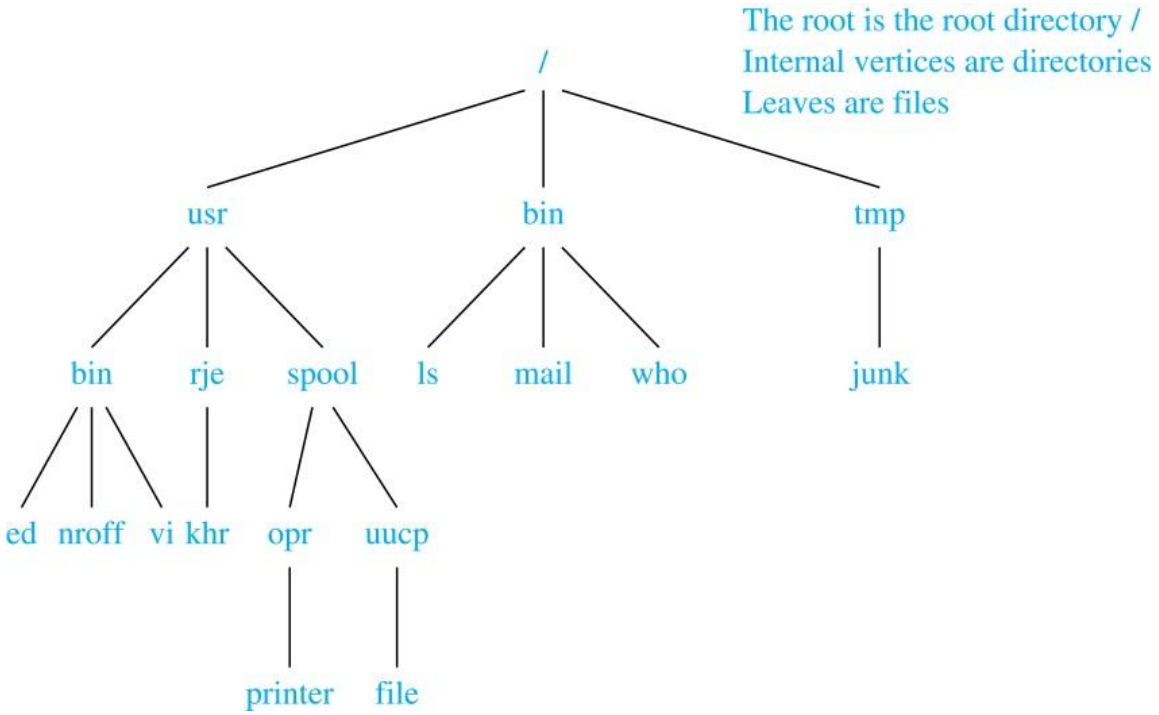
Butane



Isobutane

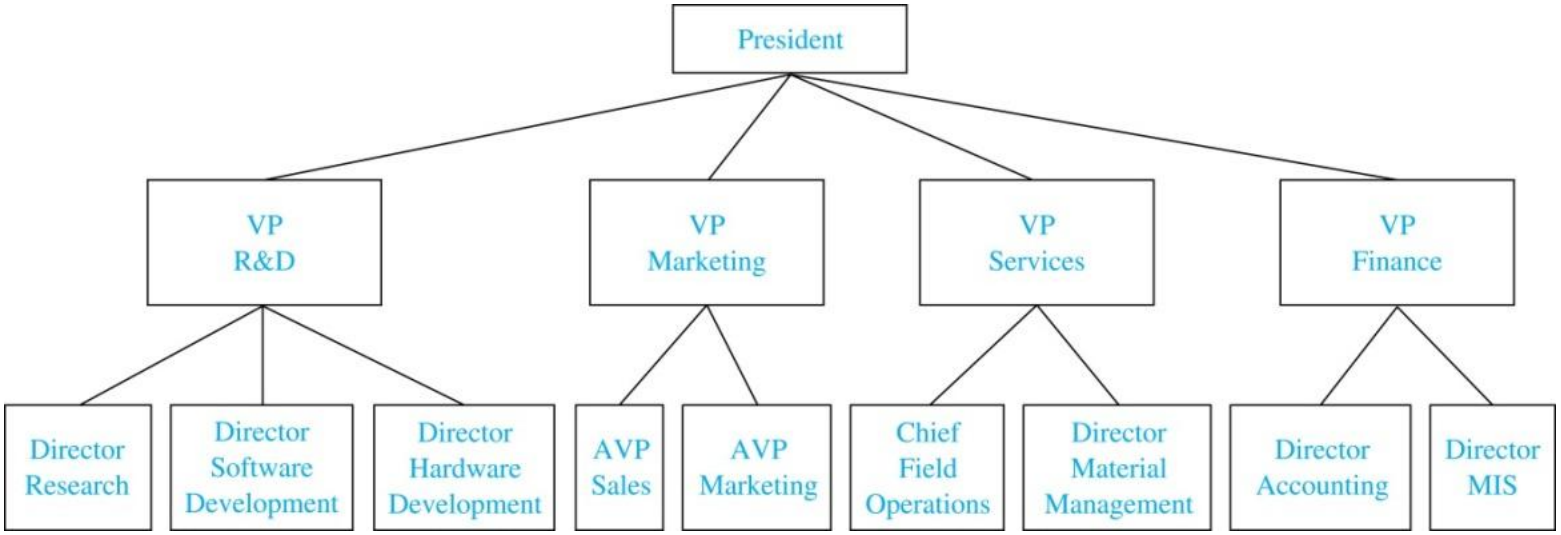
树

- 树作为模型
  - 碳氢化合物
  - 计算机文件系统



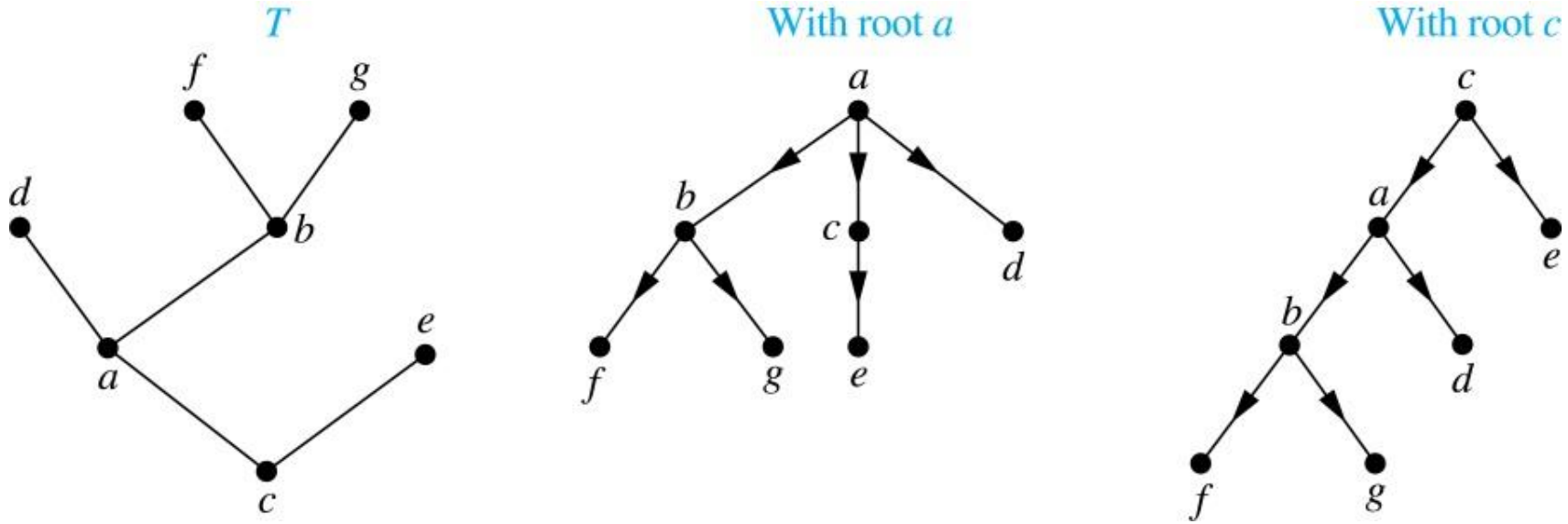
树

- 树作为模型
  - 碳氢化合物
  - 计算机文件系统
  - 组织结构



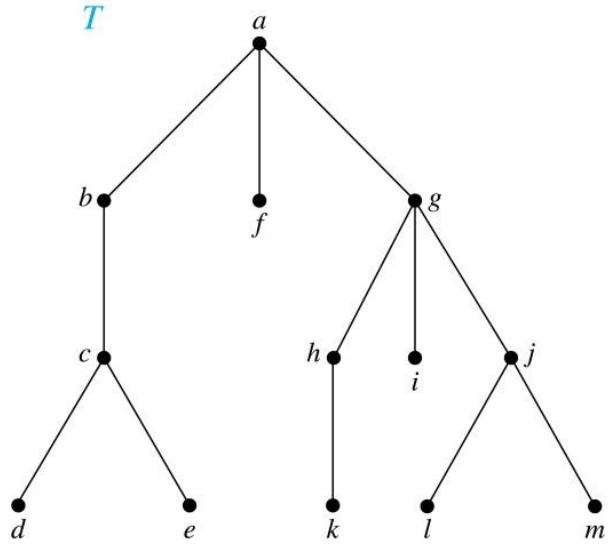
树

- 有根树 (rooted tree)
  - 指定一个顶点作为根，且每条边都离开根的树

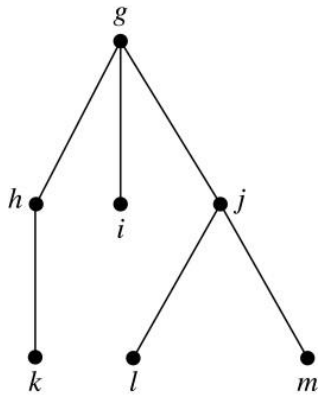


树

- 有根树 (rooted tree)
  - 指定一个顶点作为根，且每条边都离开根的树
  - 子树 (subtree)
    - a是树中顶点，以a为根的子树是由a和a的后代以及这些顶点所关联的边所组成的该树的子图



以g为根的子树

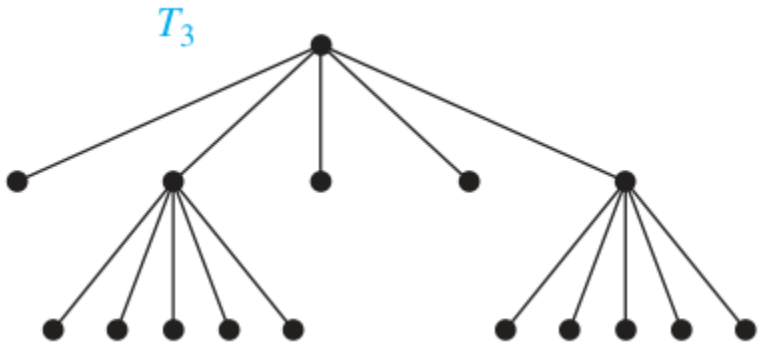


树

- 树的性质
  - 带有n个顶点的树含有$n-1$条边

树

- 树的性质
  - 带有n个顶点的树含有$n-1$条边
  - 带有i个内点的满m叉树有$n=mi+1$个顶点
    - 内点：具有分叉的顶点
    - 满m叉树：每个顶点或是叶子，或有m个分支

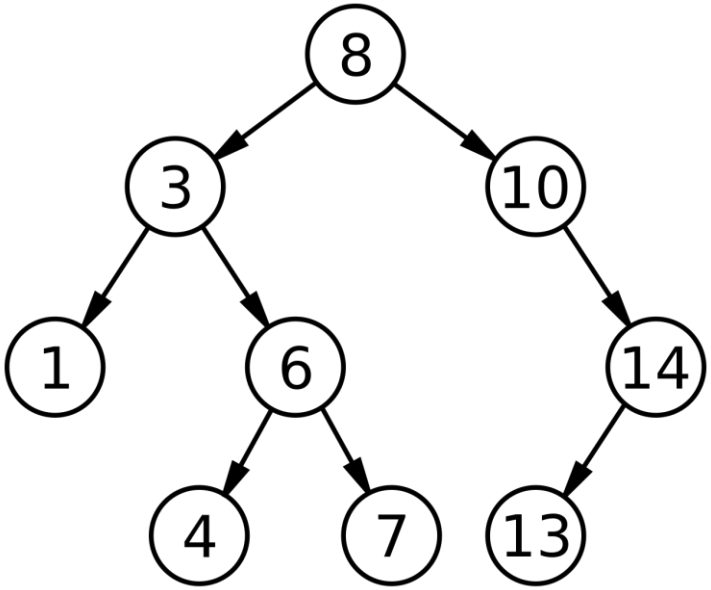


大纲

- 树的基本概念
- 树的应用 (11.2)
- 树的遍历
- 生成树
- 最小生成树

二叉搜索树

- 二叉搜索树 (binary search tree)
  - 左子树都更小
  - 右子树都更大
  - 左右子树均是二叉搜索树



二叉搜索树

- 二叉搜索树中
  查找或添加元素
  - 平衡时复杂度$O(\log n)$

```
1   procedure insert(T, x)
2       if T.root = null then
3           T.root := new Node(x)
4           return T.root
5       end if
6
7       v := T.root
8       while true do
9           if x < v.label then
10              if v.left = null then
11                  v.left := new Node(x)
12                  return v.left
13              else
14                  v := v.left
15              end if
16          else if x > v.label then
17              if v.right = null then
18                  v.right := new Node(x)
19                  return v.right
20              else
21                  v := v.right
22              end if
23          else  // x == v.label: already exists
24              return v    // 或抛出异常/忽略，依需求而定
25          end if
26      end while
```

空树

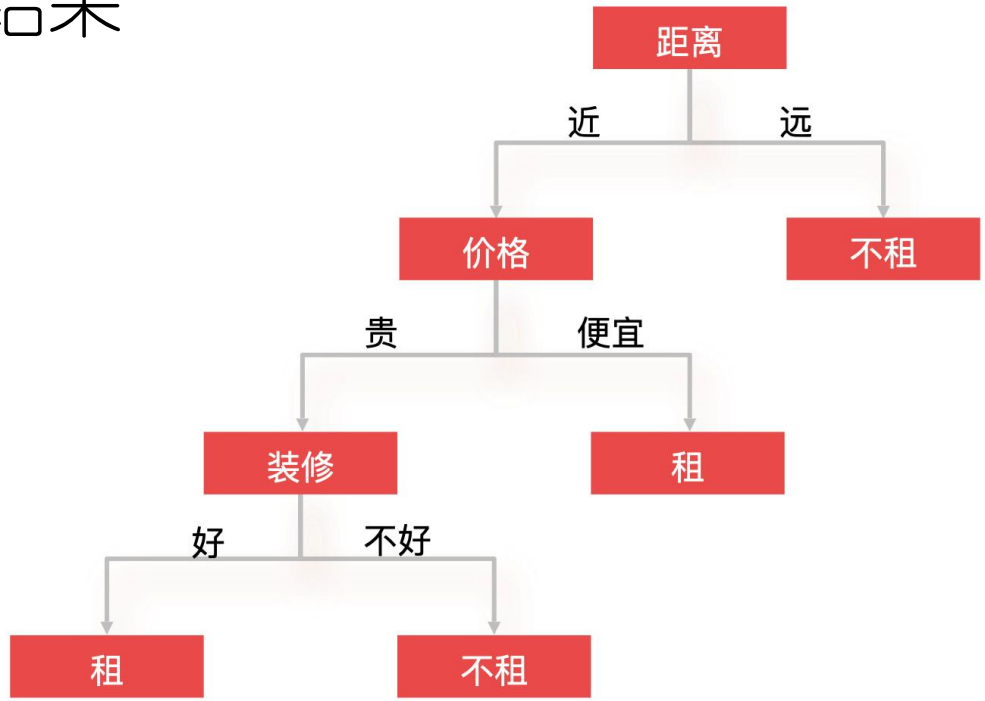
向左

向右

找到

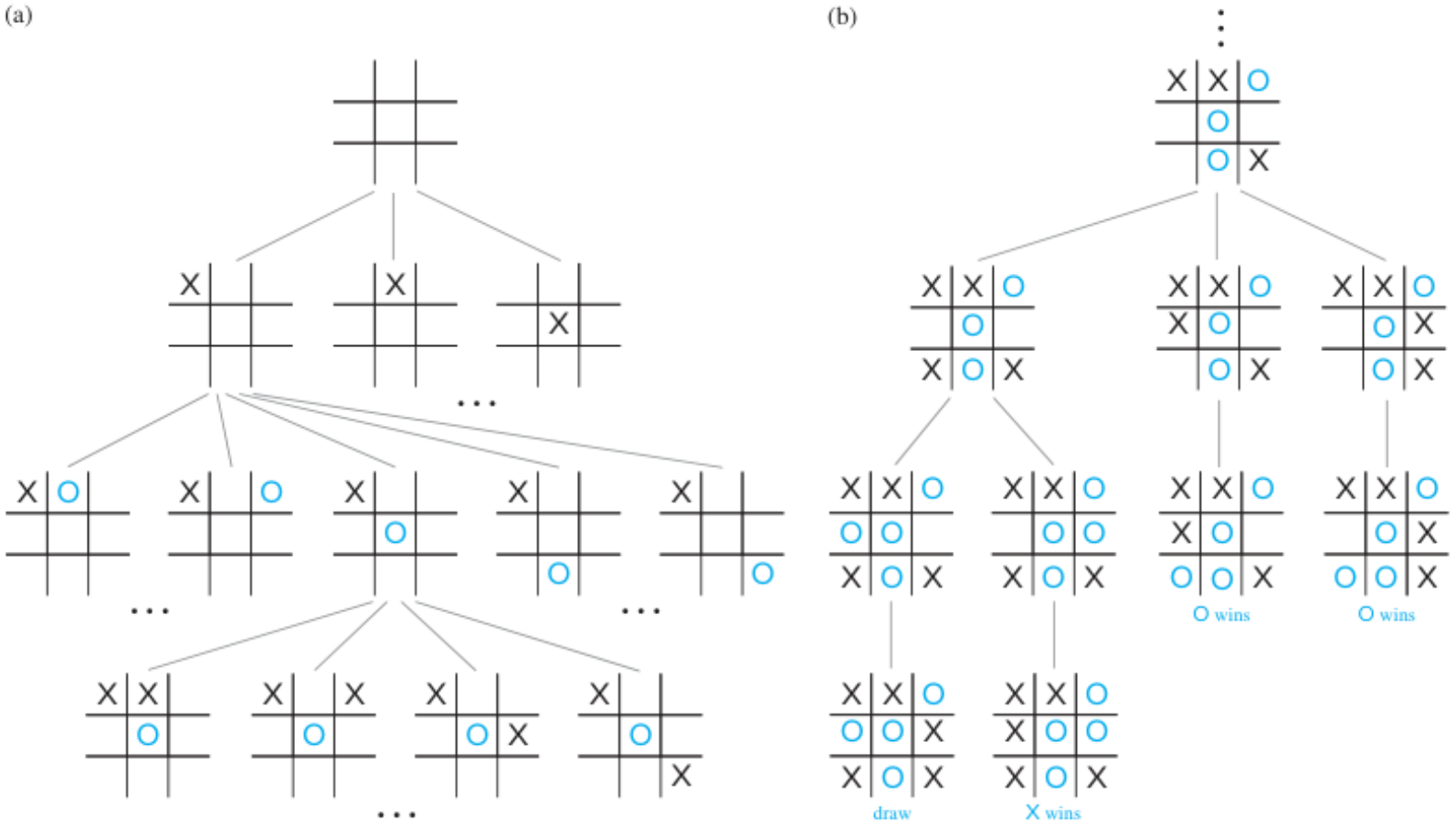
决策树

- 决策树 (decision tree)
  - 内点：一次决策
  - 子树：决策后的结果



博弈树

- 博弈树 (game tree)



大纲

- 树的基本概念
- 树的应用
- 树的遍历 (11.3)
- 生成树
- 最小生成树

树的遍历

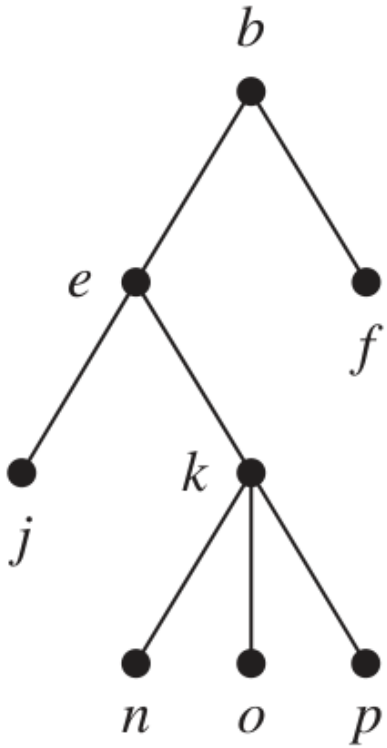
- 遍历 (traversal)
  - 访问有序根树每个顶点的过程

树的遍历

- 遍历 (traversal)
  - 访问有序根树每个顶点的过程
    - 前序遍历：根 → 左子树 → 右子树
    - 中序遍历：左子树 → 根 → 右子树
    - 后序遍历：左子树 → 右子树 → 根

例

- 给出前/中/后序遍历访问下列根树中顶点的顺序



树的遍历

- 前序遍历 (preorder traversal)

ALGORITHM 1 Preorder Traversal.

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($T(c)$)

树的遍历

- 后序遍历 (postorder traversal)

ALGORITHM 3 Postorder Traversal.

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

树的遍历

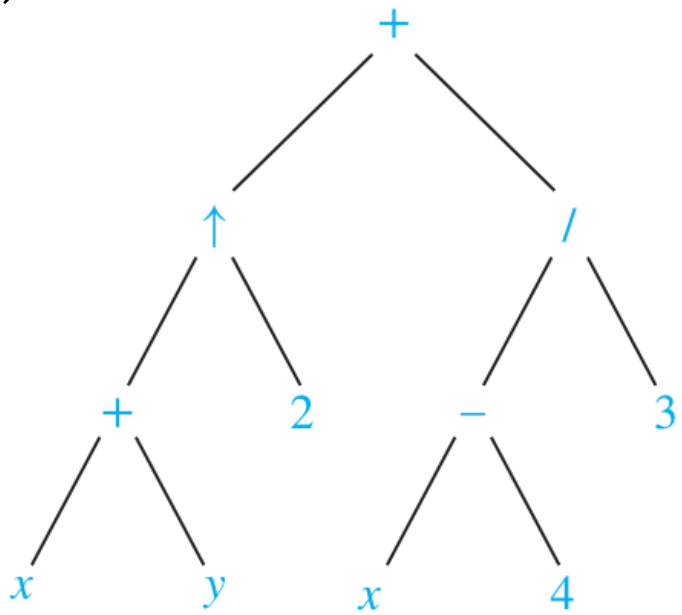
- 中序遍历 (inorder traversal)

ALGORITHM 2 Inorder Traversal.

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($T(c)$)

树的遍历

- 表达式的中缀记法 (infix notation)
  - ((x + y) ↑ 2) + ((x − 4) / 3)



大纲

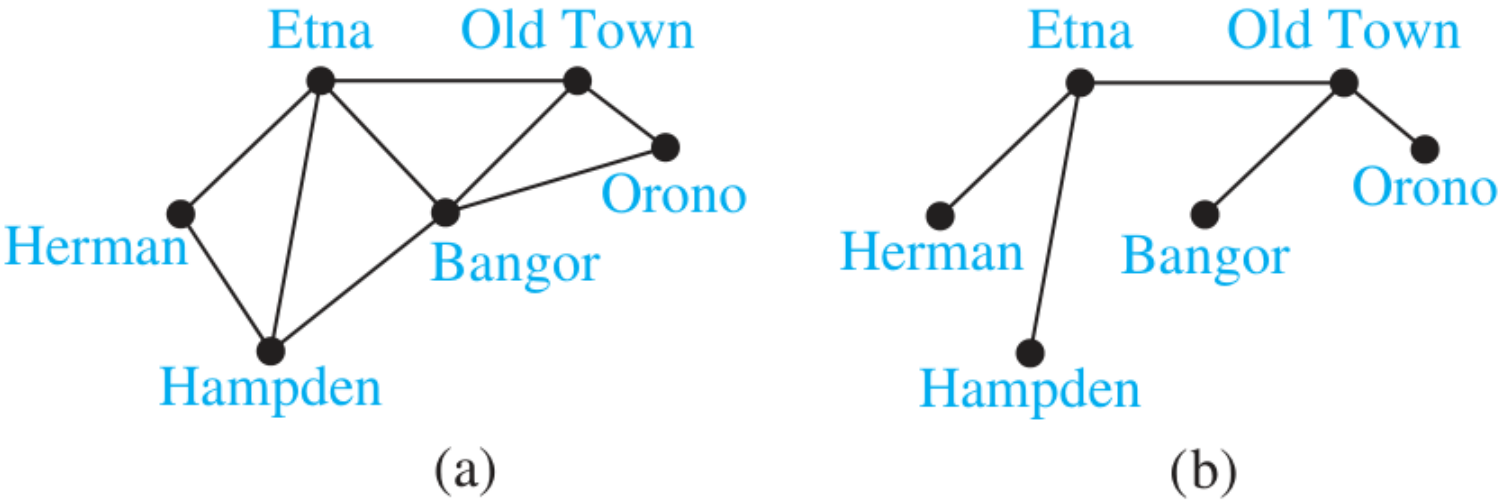
- 树的基本概念
- 树的应用
- 树的遍历
- 生成树 (11.4)
- 最小生成树

生成树

- 生成树 (spanning tree)
  - 简单图的生成树是包含每个顶点且是树的子图

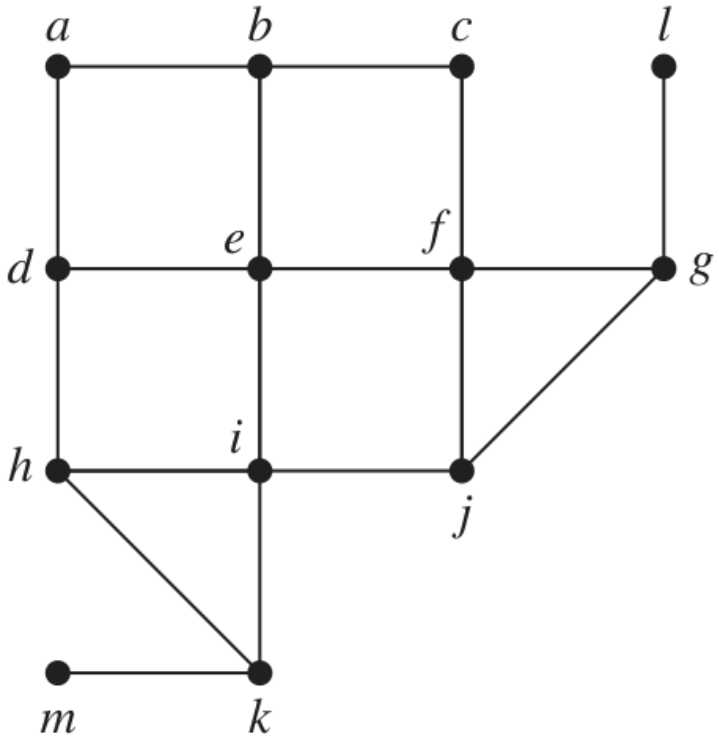
生成树

- 生成树 (spanning tree)
  - 简单图的生成树是包含每个顶点且是树的子图
  - 例：道路铲雪



生成树

- 宽度优先搜索 (breadth-first search, BFS)
  - 思路：由近到远，逐层往外走



生成树

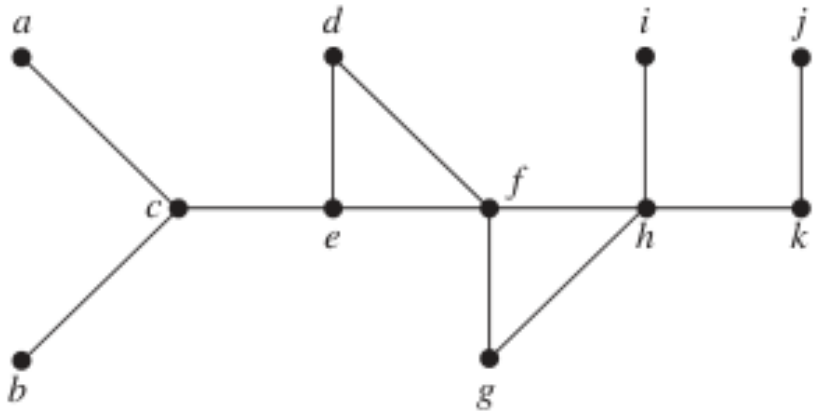
- 宽度优先搜索 (breadth-first search, BFS)
  - 思路：由近到远，逐层往外走

ALGORITHM 2 Breadth-First Search.

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

生成树

- 深度优先搜索 (depth-first search, DFS)
  - 思路：一个分支走到底，再走其他分支



生成树

- 深度优先搜索 (depth-first search, DFS)
  - 思路：一个分支走到底，再走其他分支

ALGORITHM 1 Depth-First Search.

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*)

生成树

- 回溯 (backtracking)
  - 找到失败解时，向上回到具有其他分支的父节点

- 回溯的应用
  - 求正整数集合$\{x_1, \ldots, x_n\}$的和为M的子集

  - 例
    - 求 {31, 27, 15, 11, 7, 5} 的和为39的子集

生成树

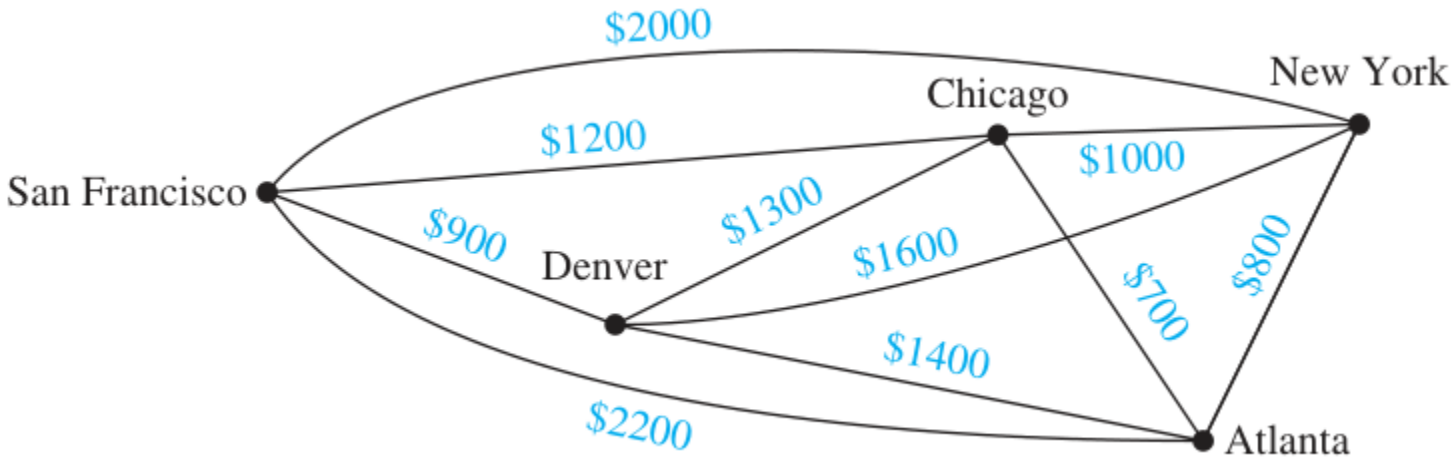
- 回溯的应用
  - n皇后问题：在$n \times n$棋盘上如何放置n个皇后，使得没有两个皇后可以互相攻击
  - 皇后：可以攻击同行、同列、同对角线

大纲

- 树的基本概念
- 树的应用
- 树的遍历
- 生成树
- 最小生成树 (11.5)

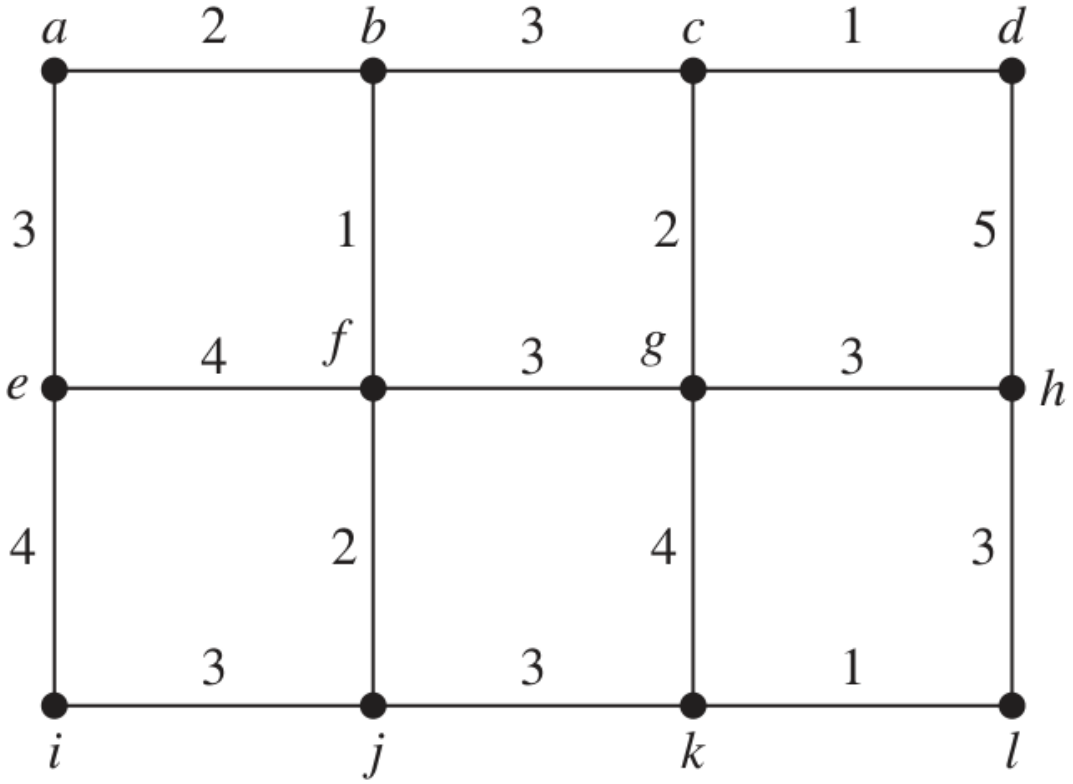
最小生成树

- 最小生成树 (minimum spanning tree)
  - 边权之和最小的生成树
  - 例
    - 连接计算中心的通信网络



最小生成树

- Kruskal算法
  - 思路：按权重从小到大考虑不形成回路的边



最小生成树

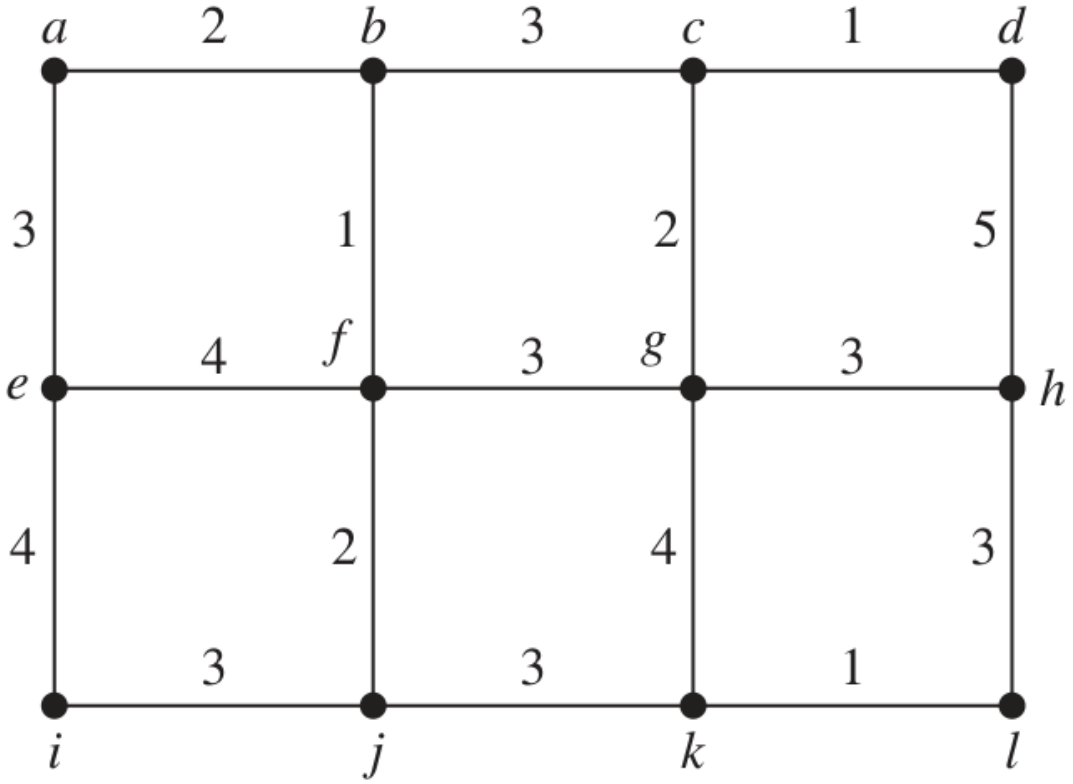
❑ Kruskal算法

ALGORITHM 2 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}

最小生成树

- Prim算法
  - 思路：按权重从小到大考虑树的生长



最小生成树

❑ Prim算法

ALGORITHM 1 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}

总结

- 树的基本概念
  - 树、有根树、子树、m叉树性质
- 树的应用
- 树的遍历
  - 前/中/后序遍历
- 生成树
  - 深度/宽度优先搜索、回溯算法
- 最小生成树
  - Kruskal算法