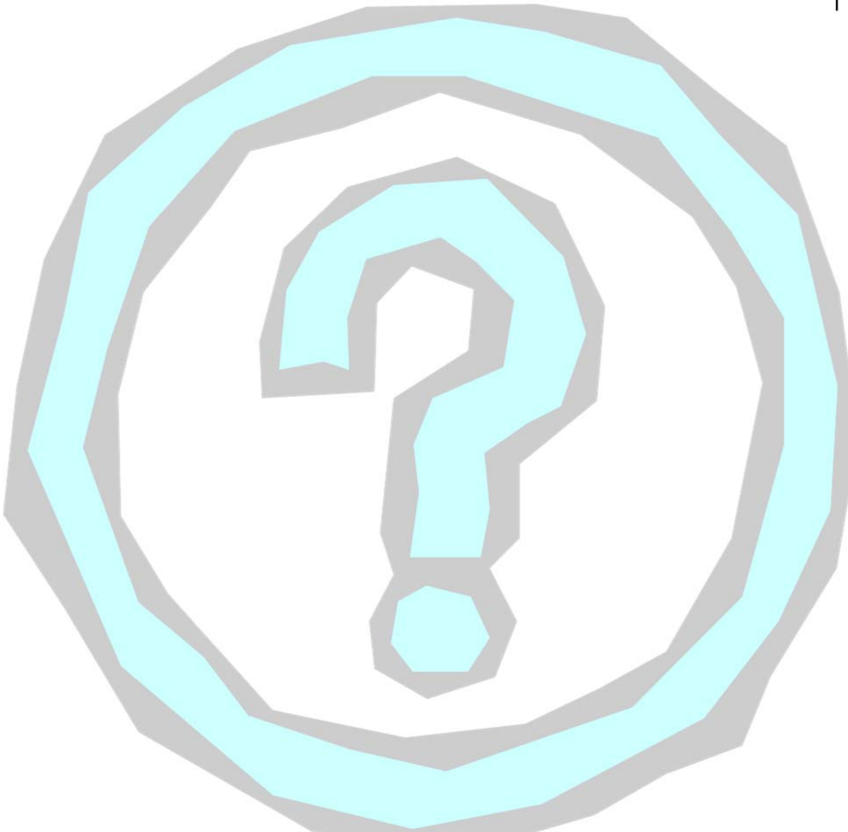
# 图论 (四)

树

南京大学计算机科学与技术系

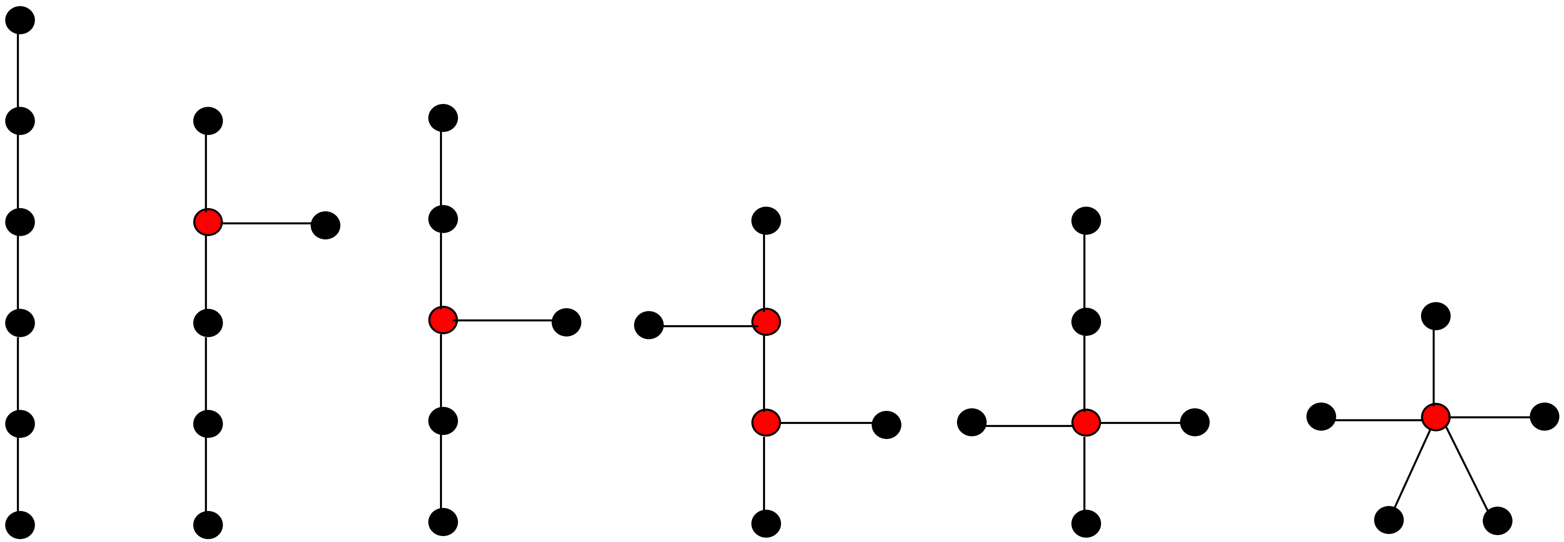
# 内容提要

- 树的定义
- 树的性质
- 根树
- 生成树



# 树的定义

- 定义：不包含简单回路的连通无向图称为树。
  - 森林（连通分支为树）
  - 树叶/分支点（度为1？）

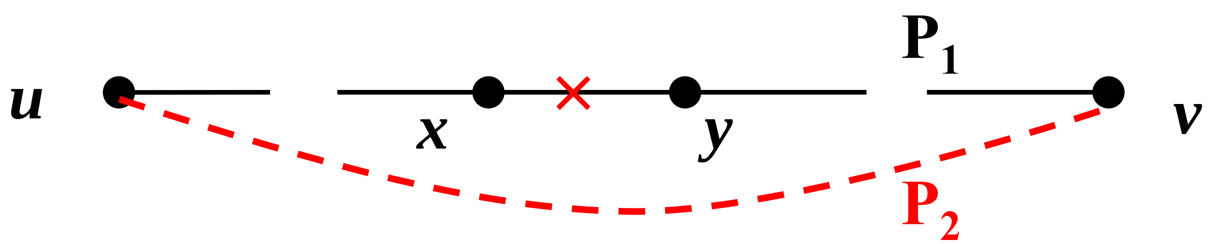


互不同构的6个顶点的树

# 树中的通路

- 设$T$是树，则$\forall u,v\in V_T$, $T$中存在唯一的 uv-简单通路。
  - 证明：$T$是连通图，$\therefore \forall u,v\in V_T$, $T$中存在uv-简单通路。

  假设$T$中有两条不同的uv-简单通路$P_1,P_2$。不失一般性，存在$e=(x,y)$满足：$e\in P_1$但$e\notin P_2$，且在路径$P_1$上$x$比$y$靠近u。令$T^*=T-\{e\}$，则$T^*$中包含$P_2$, 于是($P_1$中的$x$u-段)+$P_2$+( $P_1$中的v$y$-段)是$T^*$中的$xy$-通路，$\therefore T^*$中含xy-简单通路(记为P’)，则P’+e是$T$中的简单回路，与树的定义矛盾。



# 有关树的几个等价命题

- 设$T$是简单无向图，下列四个命题等价：
  (1) $T$是不包含简单回路的连通图。//树的定义
  (2) $T$中任意两点之间有唯一简单通路。
  (3) $T$连通，但删除任意一条边则不再连通。
  (4) $T$不包含简单回路，但在任意不相邻的顶点对之间加一条边则产生唯一的简单回路。
- 备注：
  - 树是边最少的连通图
  - 树是边最多的无简单回路的图

# 树中边和点的数量关系

- 设$T$是树，令$n=|V_T|$, $m=|E_T|$, 则$m=n-1$。
- 证明. 对n进行归纳证明。当n=1, $T$是平凡树，结论显然成立。

  假设当$n\le k$是结论成立。

  若$n=k+1$。因为$T$中每条边都是割边，任取$e\in E_T$, $T$-{e}含两个连通分支，设其为$T_1$, $T_2$, 并设它们边数分别是$m_1$, $m_2$, 顶点数分别是$n_1$, $n_2$, 根据归纳假设：$m_1=n_1-1$, $m_2=n_2-1$。注意：$n_1+n_2=n$, $m_1+m_2=m-1$。

  $\therefore m= m_1+m_2+1=n-1$。

# 连通图边数的下限

- 顶点数为n（ $n\geq 2$ ）的连通图，其边数 $m\geq n-1$ 。

  (对于树， $m=n-1$ ，“树是边最少的连通图”)

  - 证明：对n进行一般归纳。当 $n=2$ 时结论显然成立。

  设G是边数为m的连通图，且 $|V_G|=n>2$ 。任取 $v\in V_G$ ，令 $G'=G-v$ ,设G'有 $\omega(\omega\geq 1)$ 个连通分支 $G_1,G_2,\ldots,G_\omega$ ，且 $G_i$ 的边数和顶点数分别是 $m_i$ 和 $n_i$ 。

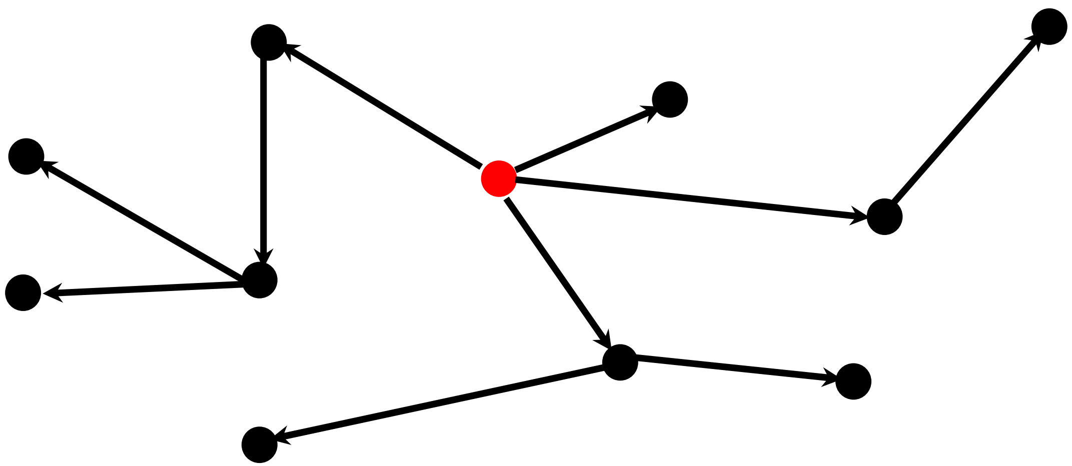
  我们有 $n=n_1+n_2+\ldots+n_\omega+1$ , $m\geq m_1+m_2+\ldots+m_\omega+\omega$ (每个连通分支中至少有一个顶点在G中与删除的 $v$ 相邻)。

  由归纳假设， $m_i\geq n_i-1(i=1,2,\ldots\omega)$ 。

  所以： $m\geq m_1+m_2+\ldots+m_\omega+\omega\geq n_1+n_2+\ldots+n_\omega-\omega+\omega=n-1$ 。

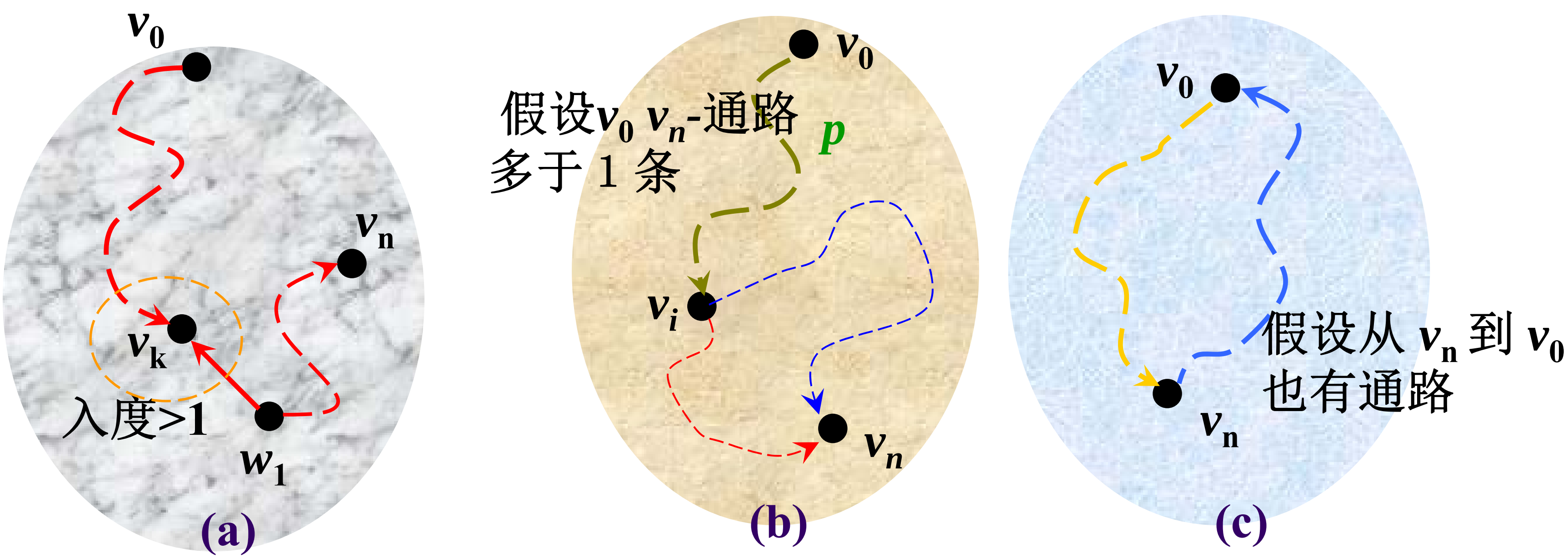
# 根树的定义

- 定义：底图为树的有向图称为*有向树*。
- 定义：若有向树恰含一个入度为0的顶点，其它顶点入度均为1，则该有向树称为*根树*，那个入度为0的顶点称为*根*。



# 根树中的有向通路

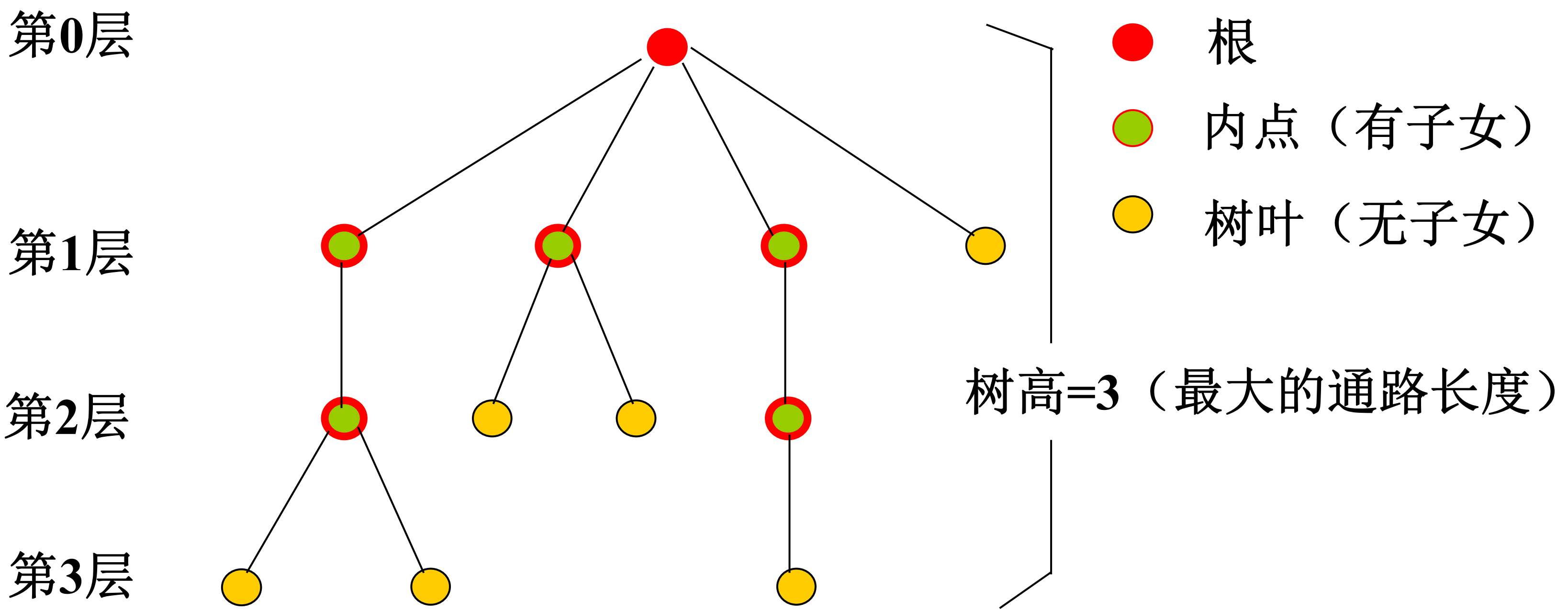
- 若$v_0$是根树T的根,则对T中任意其它顶点$v_n$ ,存在唯一的有向$v_0v_n$-通路,但不存在$v_nv_0$-通路。



# 根树的图形表示

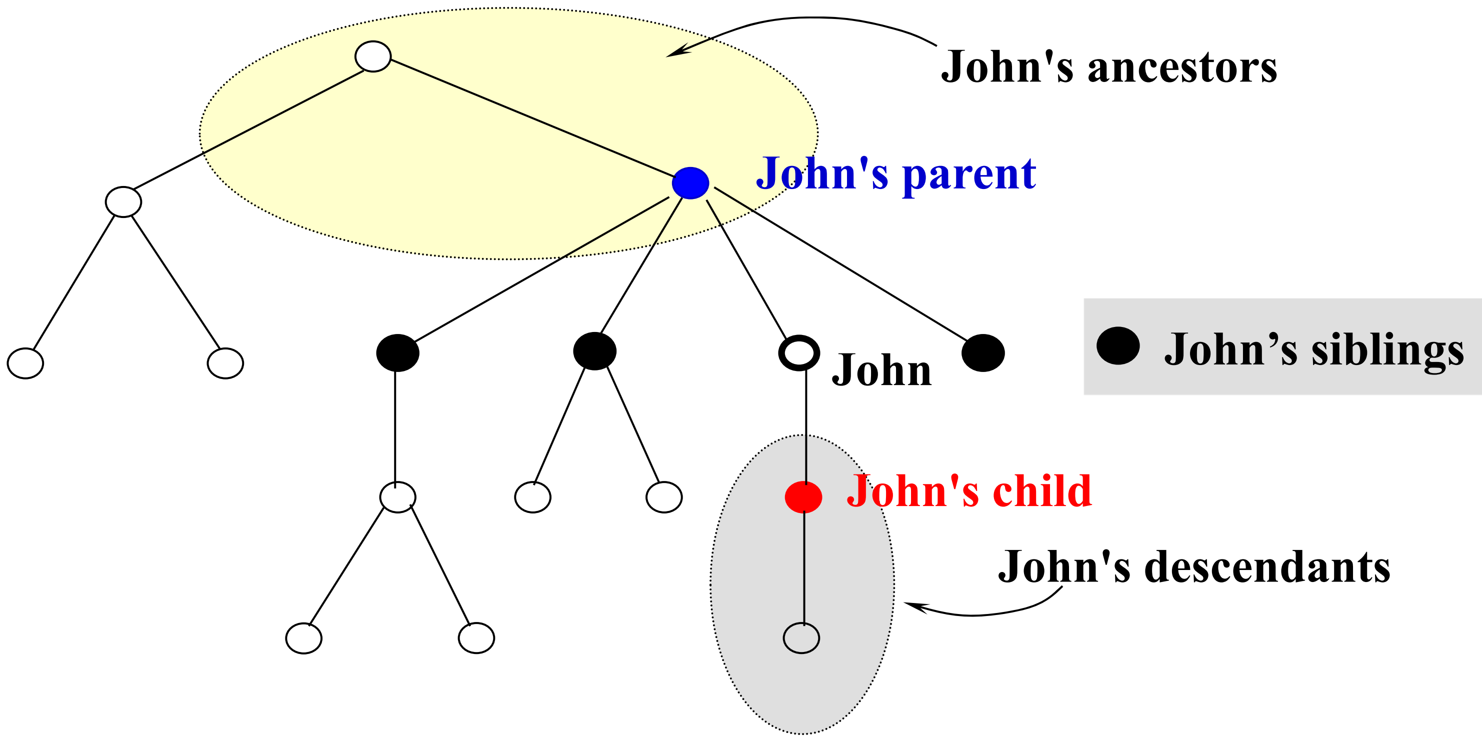
- 边上的方向用约定的位置关系表示

根也是内点，除非它是图中唯一顶点。



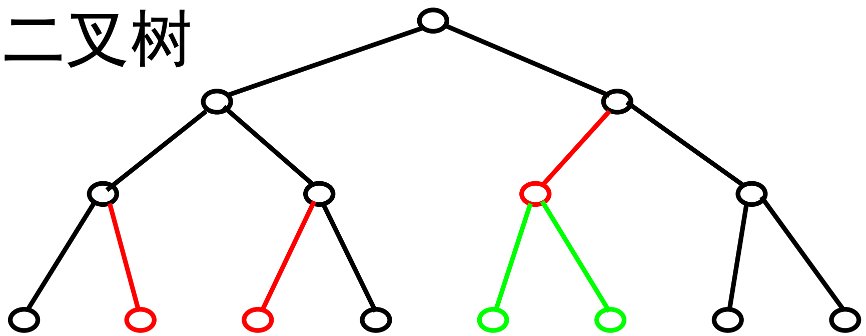
# 根树与家族关系

- 用根树容易描述家族关系，反之，家族关系术语被用于描述根树中顶点之间的关系。



# 根树的几个术语

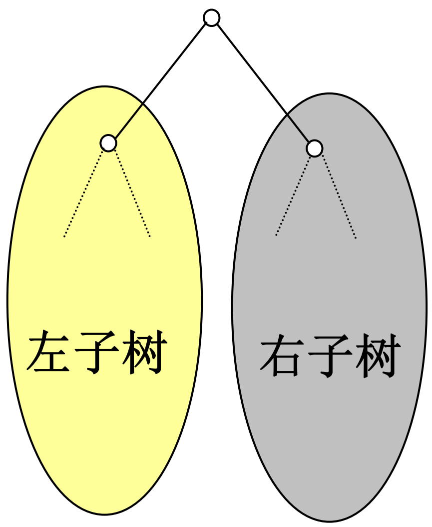
- m元树：每个内点至多有m个子女
  - 2元树也称为二叉树
- 完全*m*元树(*full* m-ary tree)
  - 每个内点恰好有m个子女
- 平衡：树叶都在h层或(h-1)层, h为树高。
- 有序：同层中每个顶点排定次序
  - 有序二叉树通常也简称为二叉树



# 根树的几个术语（续）

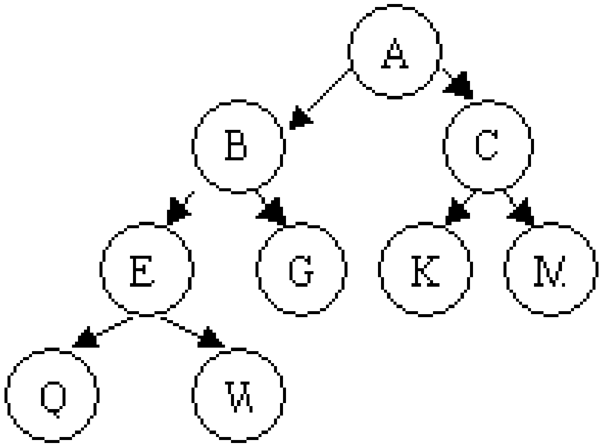
- 定义：设$T$是根树，$T$中任一顶点$v$及其所有后代的导出子图显然也是根树(以$v$为根)，称为$T$的根子树。
- 有序二叉树的子树分为左子树和右子树

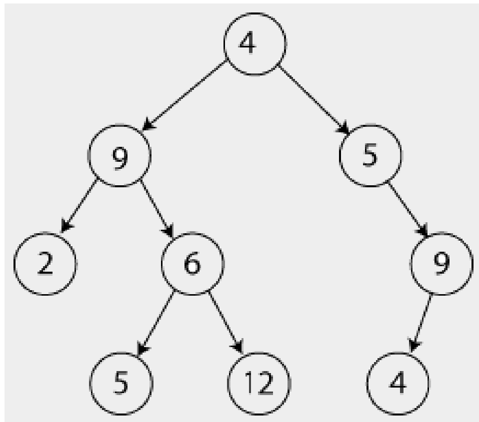
即使不是完全二叉数，也可以分左、右，必须注意顶点位置

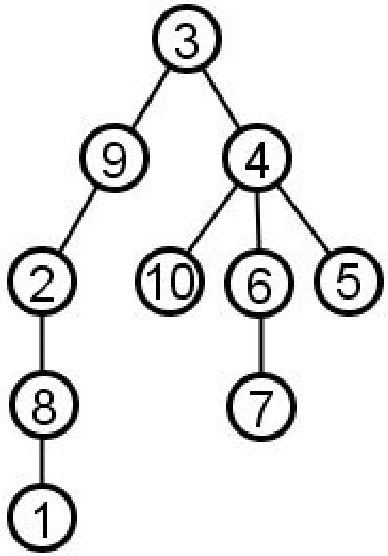


# 根树（举例）

- 树的高度、各顶点所处的层数
- 完全、平衡

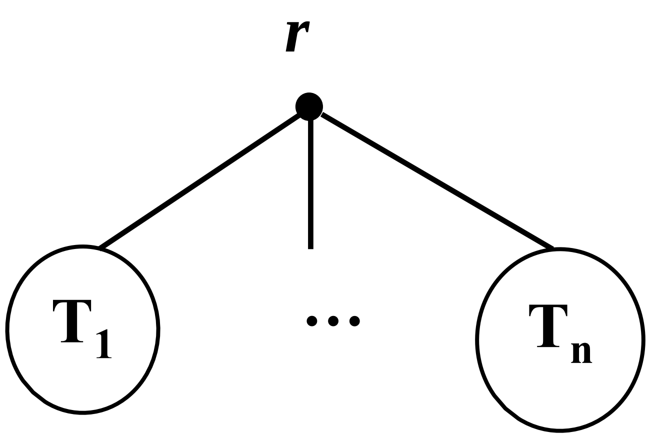


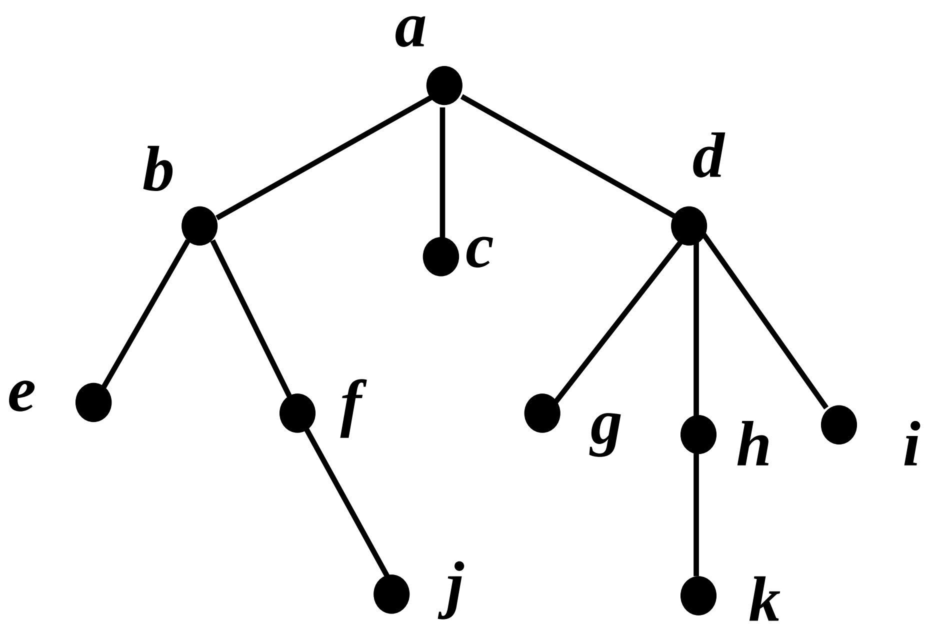
# 有序根树的遍历

- 前序遍历 （preorder）
  - T只包含根$r$，则为 $r$；
  - T的子树为$T_1$, …, $T_n$, 则为 $r$, preorder($T_1$), …, preorder($T_n$)



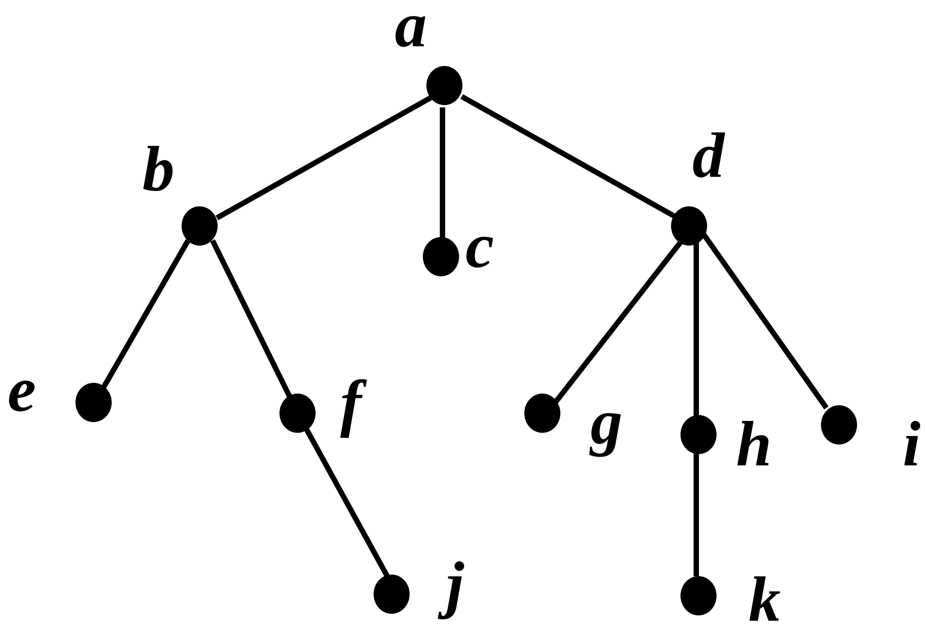
# 有序根树的遍历

- 前序遍历 （preorder）



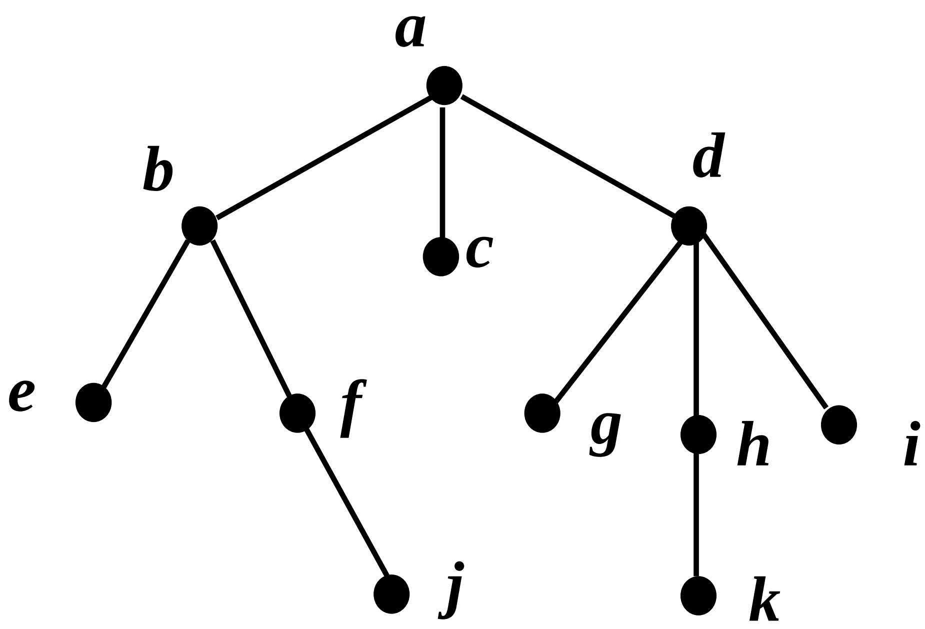
# 有序根树的遍历

- 后序遍历 （postorder）



# 有序根树的遍历

- 中序遍历 （inorder）//先访问第一棵子树

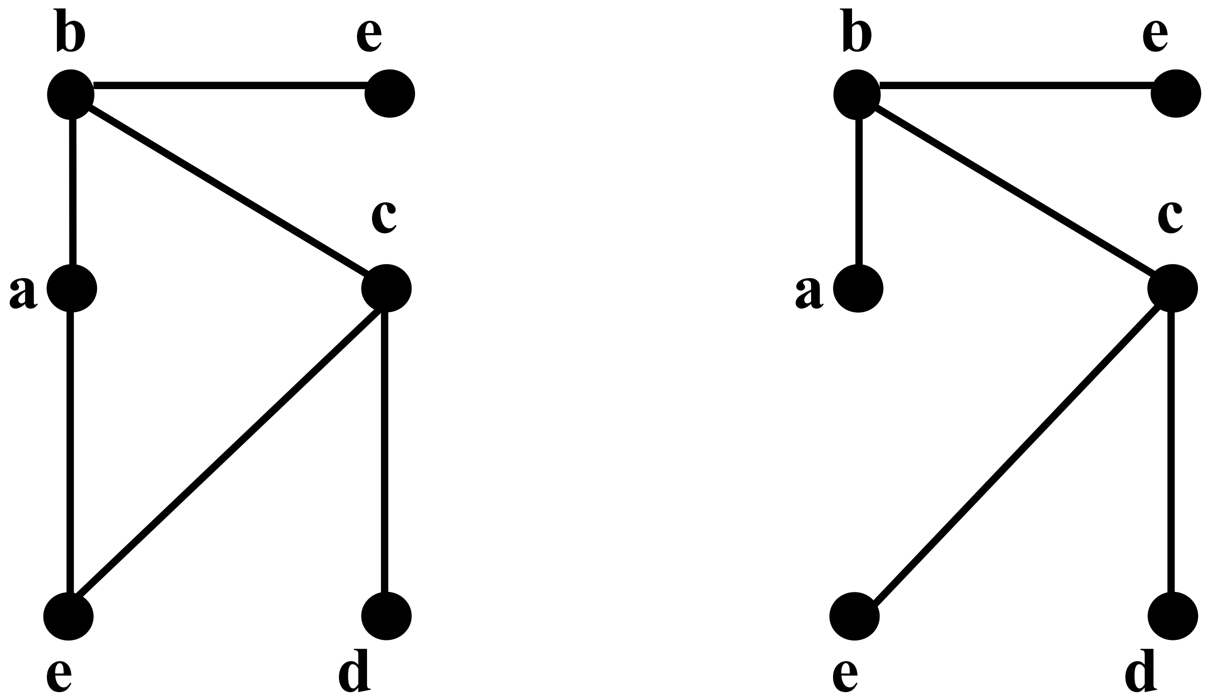


# 生成树

- 定义：若图G的生成子图是树，则该子图称为G的生成树。
- 无向图G连通 当且仅当 G有生成树
  - 证明(充分性显然)：

  ⇒ 注意：若G是有简单回路的连通图，删除回路上的一条边，G中的回路一定减少。(因此，用“破圈法”总可以构造连通图的生成树)
- 简单无向图G是树 当且仅当 G有唯一的生成树。
  - 注意：G中任一简单回路至少有三条不同的边。

# 构造生成树：深度优先搜索



# 深度优先搜索算法

**Procedure DFS(G: 带顶点$v_1$, …,$v_n$的连通图)**

**T:=只包含顶点$v_1$的树；**

**visit($v_1$)；**

**Procedure visit($v$: G的顶点)**

**for $v$每个邻居$w$ {**

**if $w$不在T中 then {**

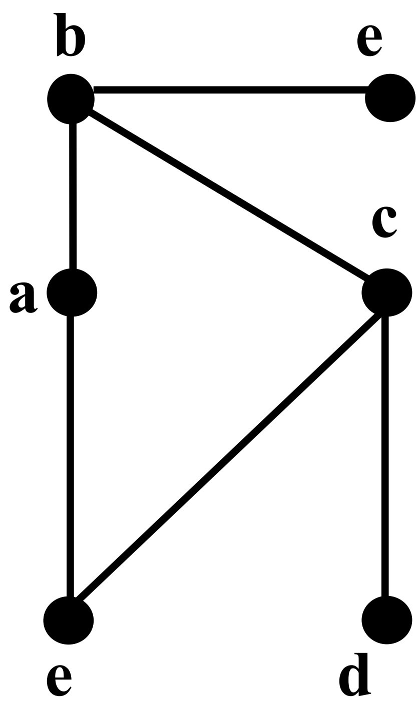
**加入顶点$w$和边{$v$, $w$}到T；**

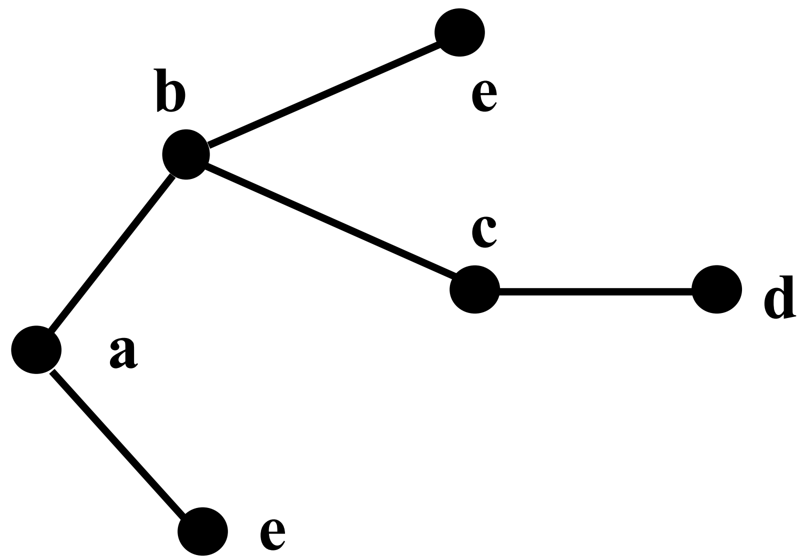
**visit($w$)；**

**}**

**}**

# 构造生成树：广度优先搜索



# 广度优先搜索算法

**Procedure BFS(G: 带顶点$v_1$, …,$v_n$的连通图)**

**T:=只包含顶点$v_1$的树；L:=空表; 把$v_1$放入表L中**

**While L非空 {**

**删除L中的第一个顶点$v$;**

**for $v$的每个邻居$w$ {**

**if $w$既不在L中也不在T中 then {**

**加入$w$到L的末尾;**

**加入顶点$w$和边{$v$, $w$}到T;**

**}**

**}**

**}**

# 最小生成树 MST
# Minimum Spanning Tree

- 考虑边有权重的连通无向图。其生成树可能不唯一。定义生成树的权重为其所含各边之和。一个带权连通图的最小生成树是其权重最小的生成树。
  - 注意，这里的最小(Minimum)并不意味着唯一。
- 最小生成树有广泛的应用。

# Prim算法（求最小生成树）

1: E={e}, e是权最小的边

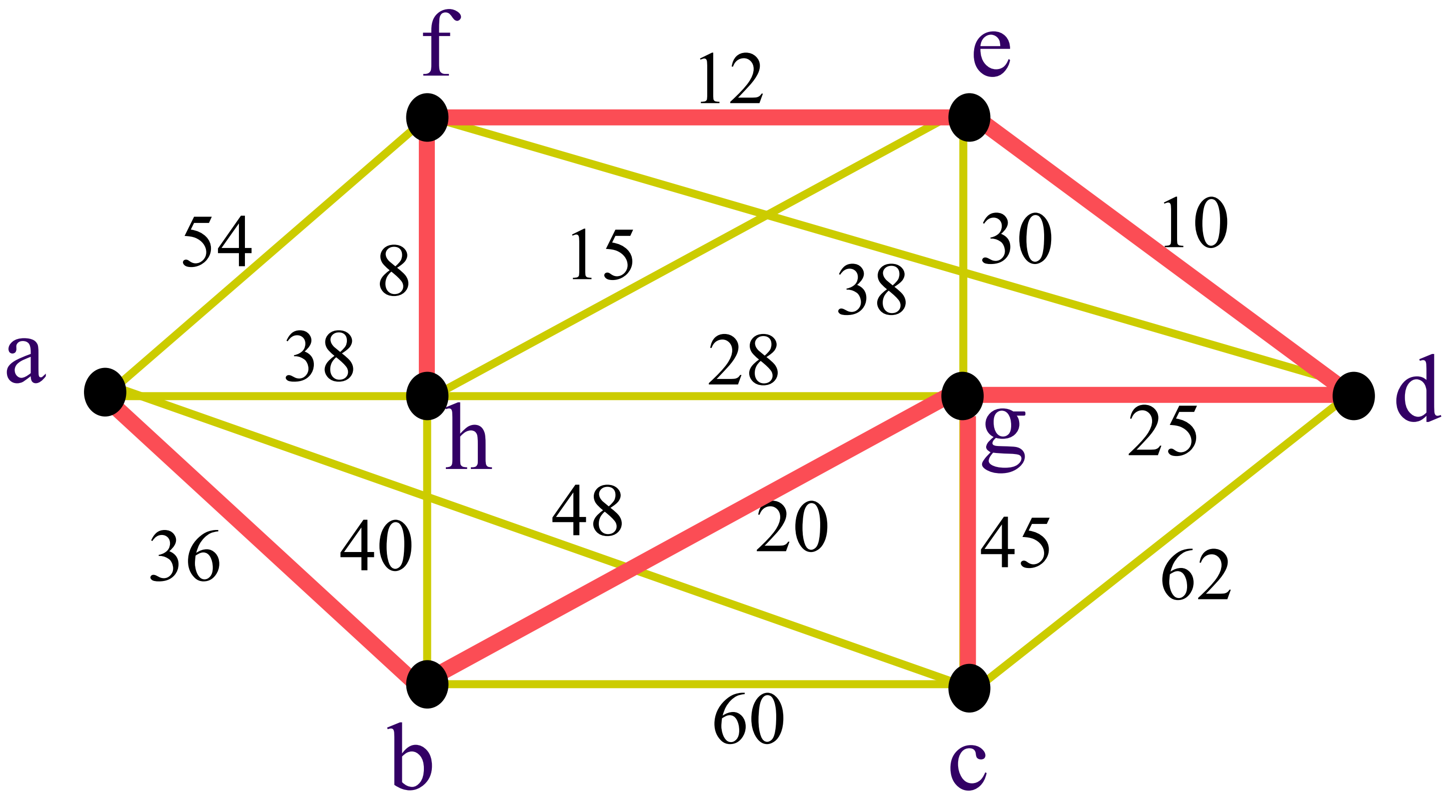
2: 从E以外选择与E里顶点关联，又不会与E中的边构成回路的权最小的边加入E

3: 重复第2步，直到E中包含n-1条边

算法结束

# Prim算法（举例）

- 铺设一个连接各个城市的光纤通信网络（单位：万元）。



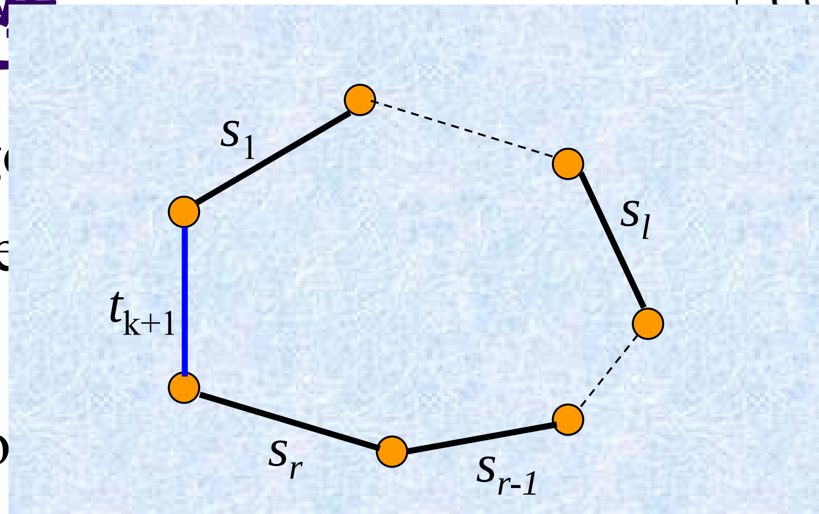
# Prim 算法的正确性

Let T be the output of Prim's alg

edges $t_1, t_2, \ldots, t_{n-1}$, as the order the

for $1 \le i \le n-1$, and $T_0 = \phi$.

It can be proved that each $T_i$ is co




Assume that $T_k$ is contained in a MST $T'$, then $\{t_1, t_2, \ldots, t_k\} \subseteq T'$.

If $t_{k+1} \notin T'$, then $T' \cup \{t_{k+1}\}$ contains a cycle, which cannot wholly be in $T_k$.

Let $s_l$ be the edge with smallest index $l$ that is not in $T_k$. Exactly one of the vertices of $s_l$ must be in $T_k$, which means that when $t_{k+1}$ was chosen, $s_l$ available as well. So, $t_{k+1}$ has no larger weight than $s_l$. So, $(T' - \{s_l\}) \cup \{t_{k+1}\}$ is a MST containing $T_{k+1}$.

# Kruskal算法（求最小生成树）

1: E={ }

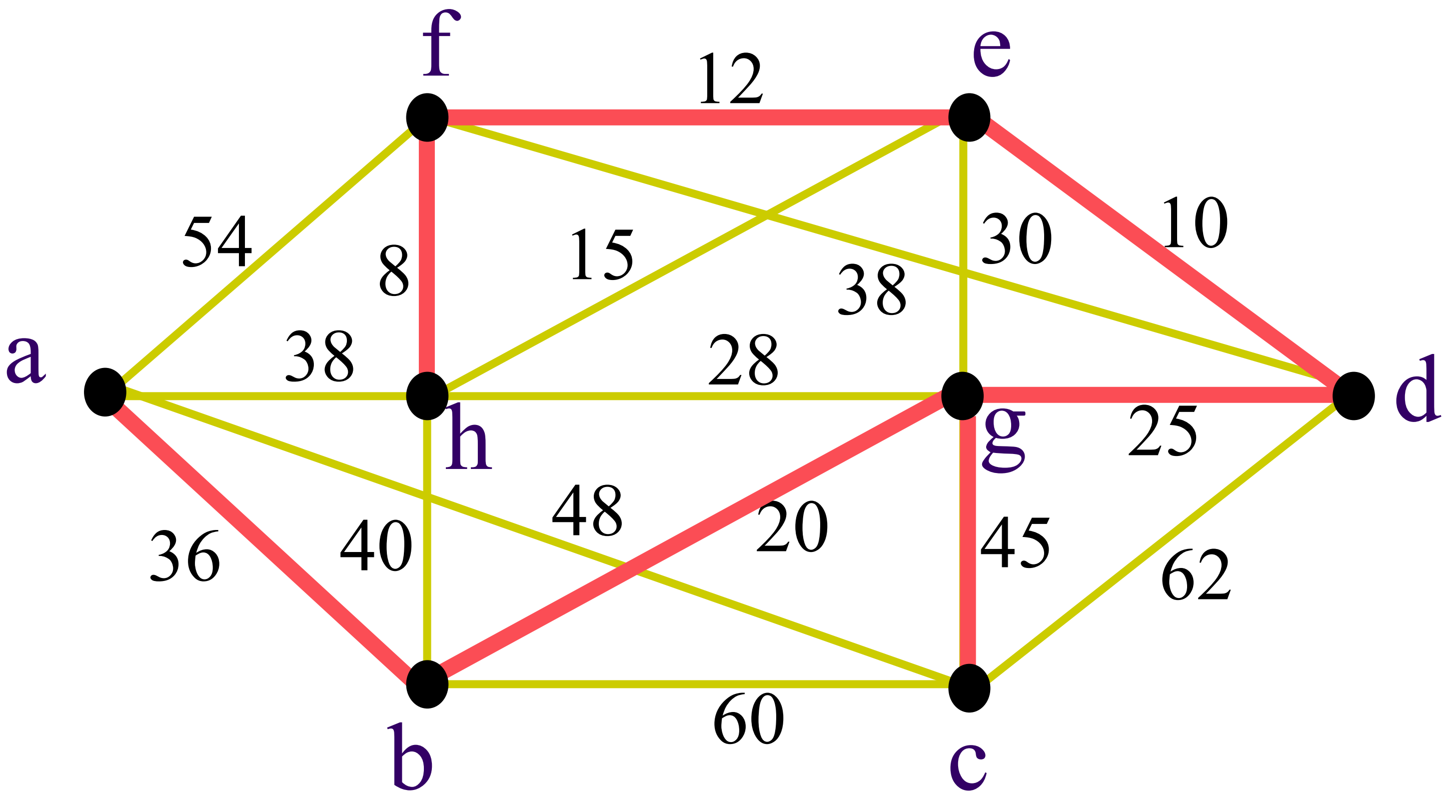
2: 从E以外选择不会与E中的边构成回路的权最小的边加入E

3: 重复第2步，直到E中包含n-1条边

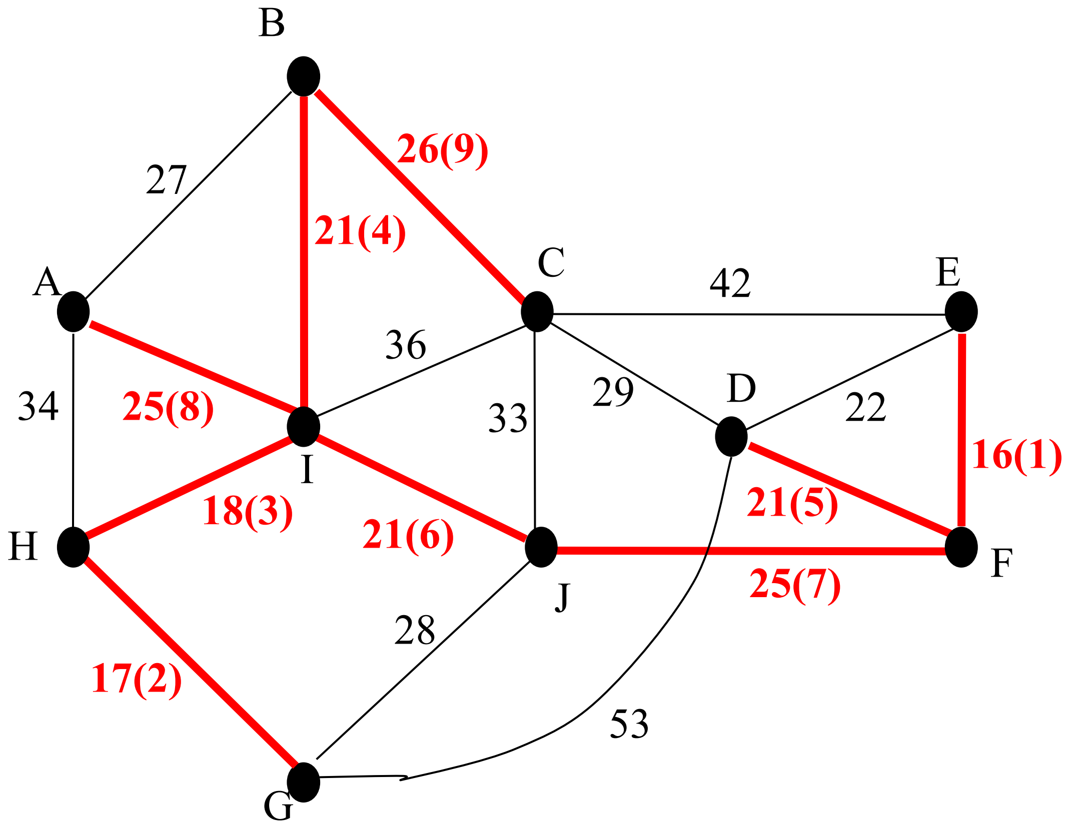
算法结束

# Kruskal算法（举例）

- 铺设一个连接各个城市的光纤通信网络（单位：万元）。



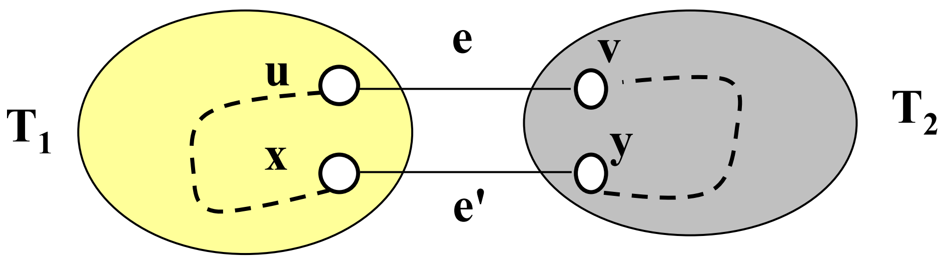
# Kruskal算法（举例）




后面证明：**Kruskal**算法的正确性

# 引理（更换生成树的边）

- T与T'均是图G的生成树，若$e\in E_T$且$e\notin E_{T'}$，则必有$e'\in E_{T'}$，$e'\notin E_T$，且$T-\{e\}\cup\{e'\}$和$T'-\{e'\}\cup\{e\}$均是G的生成树。
  - 设e=uv，T-{e}必含两个连通分支，设为$T_1$，$T_2$。因T'是连通图，T'中有uv-通路，其中必有一边满足其两个端点x,y分别在$T_1$, $T_2$中，设其为e'，显然$T-\{e\}\cup\{e'\}$是生成树。
  
    而$T'-\{e'\}$中x,y分属两个不同的连通分支，但在$T^*=T'-\{e'\}\cup\{e\}$中，xu-通路+e+vy通路是一条xy-通路，因此$T'-\{e'\}\cup\{e\}$连通，从而$T'-\{e'\}\cup\{e\}$是生成树。



# Kruskal算法的正确性

- 显然T是生成树。
- 按在算法中加边顺序，T中边是$e_1,e_2,\ldots e_{k-1}$, $e_k,\ldots e_{n-1}$。
- 假设T不是最小生成树。对于任意给定的一棵最小生成树T’, 存在唯一的k, 使得$e_k \notin E_{T'}$，且$e_i \in E_{T'}$使得($1 \le i < k$). 设T’是这样的一棵最小生成树，使得上述的k达到最大。
- 根据前述引理，T’中存在边e’，e’不属于T, 使得$T^* = T' - \{e'\} \cup \{e_k\}$也是生成树。 $e' \in T'$与$e_1,e_2,\ldots e_{k-1}$不会构成回路，因此$w(e') \ge w(e_k)$. 所以$w(T^*) \le w(T')$, 即T*也是最小生成树。但T*包含$e_1,e_2,\ldots e_{k-1},e_k$，矛盾。