# 10.5 有向树与有序树

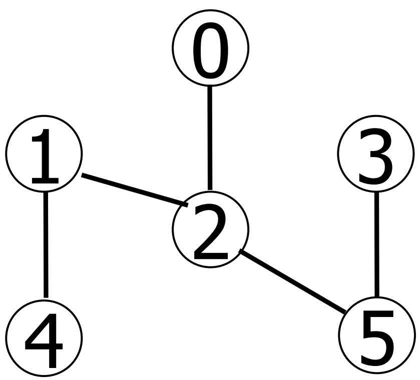
## 本节主要问题

一、有向树

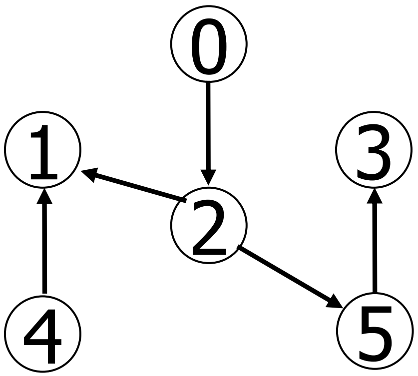
二、有根树

# 1、有向树的定义

有向树与树(无向树)相对应，树的定义是无圈的连通图。




树

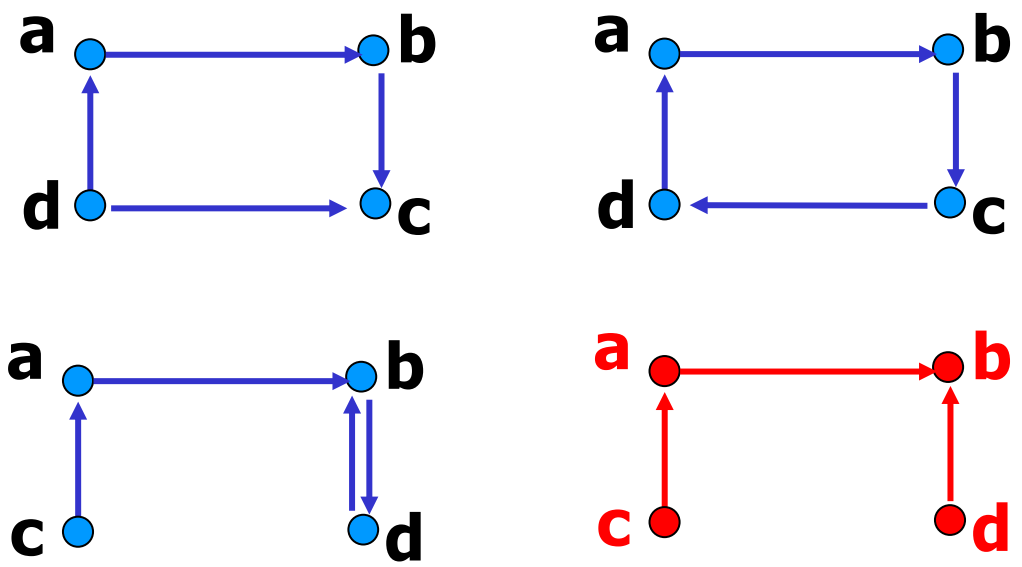



有向树

如果把树的边加上方向，就是有向树。

# 1、有向树的定义

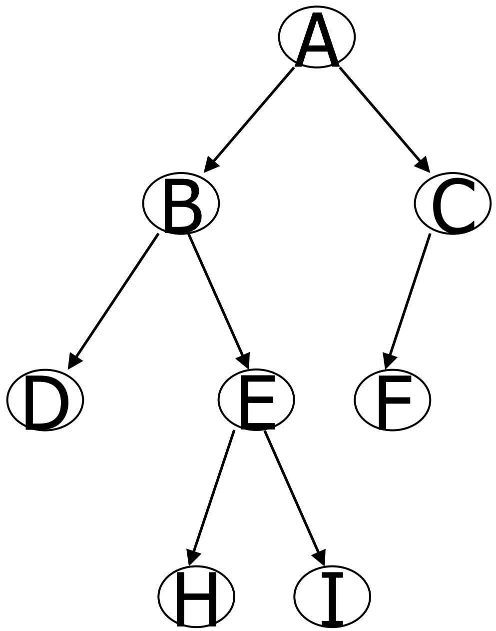
定义10.5.1 一个没有弱圈的弱连通的有向图称为有向树（一个顶点也是）。



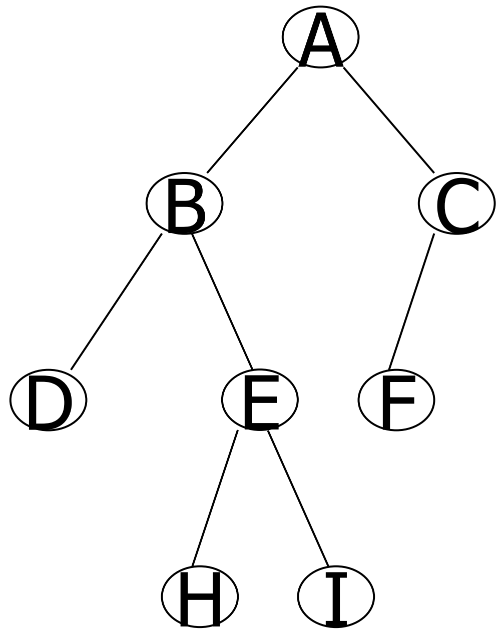

一个有向图是有向树当且仅当去掉方向后，它是一棵树。

## 2、有根树的定义

定义10.5.2 有向树D称为有根树，如果D中恰有一个顶点的入度为0，而其余每个顶点的入度均为1，有根树中入度为0的顶点称为有根树的根，出度为0的顶点称为叶子节点，其他的称为分枝点或内顶点。



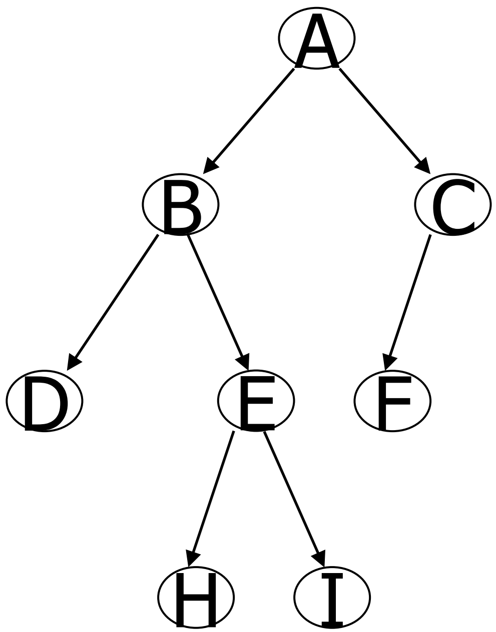

有根树




箭头可以省略

# 2、有根树的性质

定理10.5.1 有向图D=(V,A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。



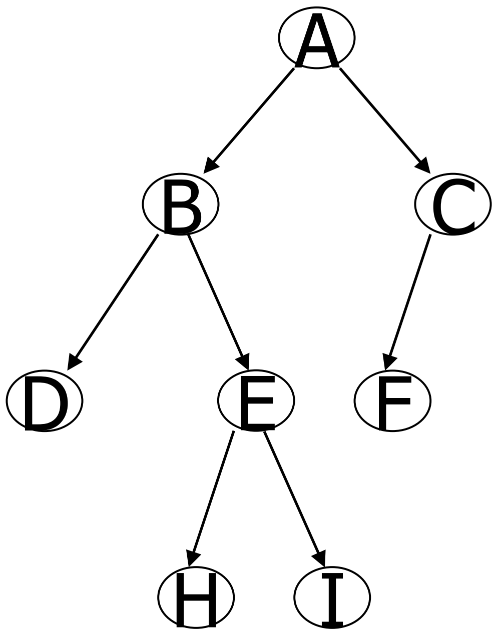

证明必要性：当D是有根树时，证明D有一个顶点可以达到其他任一顶点且D中没有弱圈。

定义10.5.1 一个没有弱圈的弱连通的有向图称为有向树（一个顶点也是）。

定义10.5.2 有向树D称为有根树，如果D中恰有一个顶点的入度为0，而其余每个顶点的入度均为1。

## 2、有根树的性质

定理10.5.1 有向图D=(V, A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。




证明必要性：当D是有根树时，则D是弱连通的且没有弱圈。

根顶点到其他每个顶点有一条弱路。

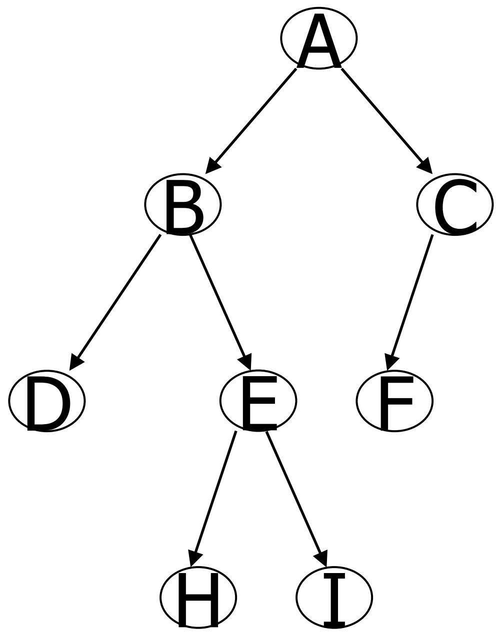
由于由D确定的无向图是无向树，所以根顶点到其他每一顶点的弱路是唯一的。

其次，再由除根顶点外每个顶点的入度为1，所以根顶点到其他每个顶点的弱路必是有向路。

因此，从根顶点可以达到其他每个顶点。

## 2、有根树的性质

定理10.5.1 有向图D=(V,A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。



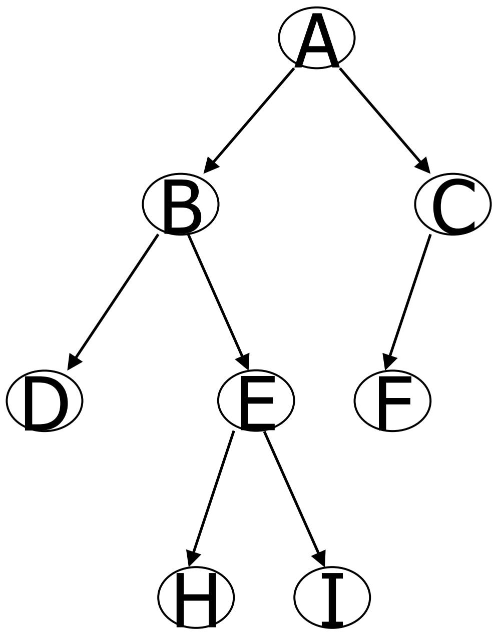

证明充分性：当D有一个顶点可以到达其它任一顶点且D中没有弱圈时，则D是有根树。

定义10.5.1 一个没有弱圈的弱连通的有向图称为有向树（一个顶点也是）。

定义10.5.2 有向树D称为有根树，如果D中恰有一个顶点的入度为0，而其余每个顶点的入度均为1。

# 2、有根树的性质

定理10.5.1 有向图D=(V,A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。



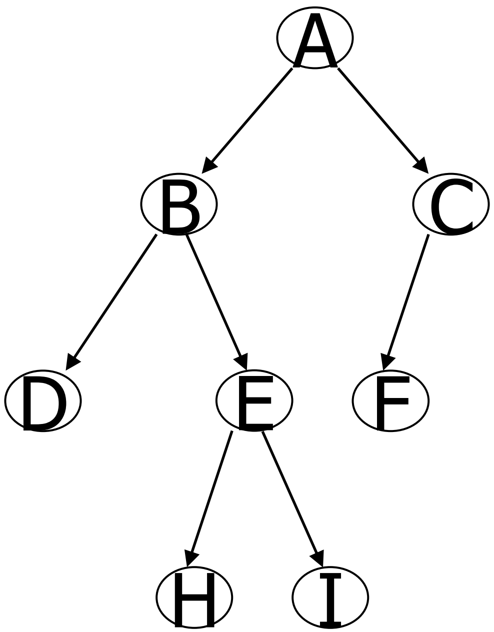

证明充分性:

1）证明是有向树：当有向图D中没有弱圈且有一个顶点v可达到其它每个顶点。因此，D是弱连通的且无弱圈，所以，D是有向树。

2）证明v的入度为0：再由v可达到其它任一顶点便知v的入度为0，因为若v的入度不为0，必有某顶点u使(u,v)为D的弧，于是，D中已存在的由v到u的路，与(u,v)合起来形成D中的圈，矛盾。

## 2、有根树的性质

定理10.5.1 有向图D=(V, A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。




3）证明其余每个顶点入度为1:

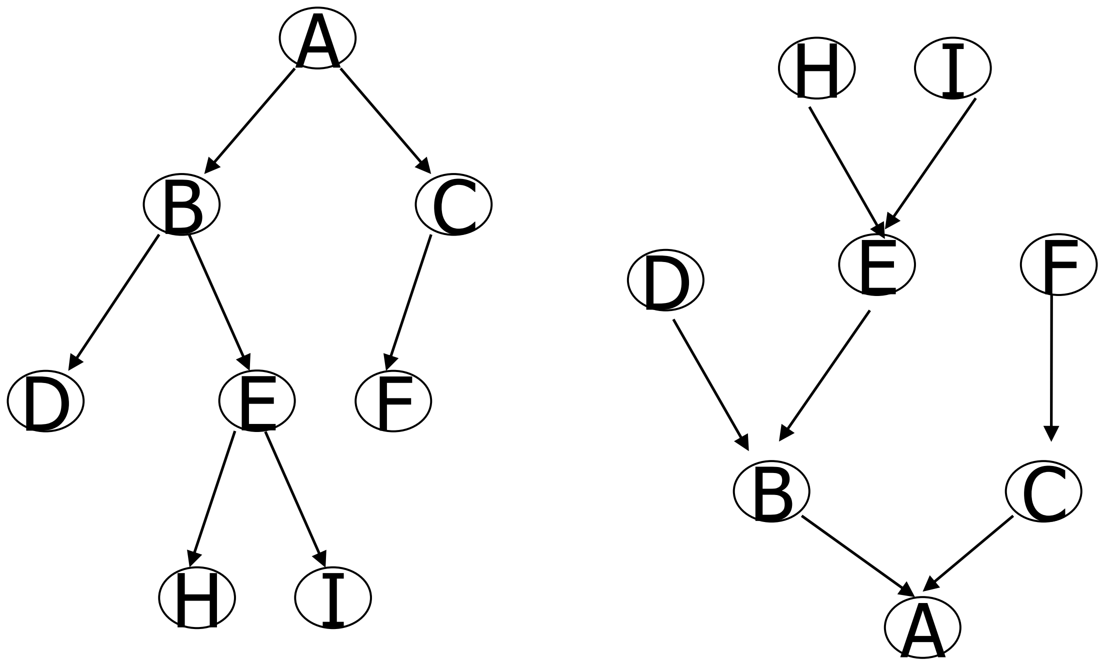
设w是D的任一顶点且w≠v，则$id(w)=1$。

否则，$id(w)\geq 2$，则有$v_1, v_2$使$(v_1, w)$，$(v_2, w)$是D的两条不同弧。

由于从v可达到$v_1$也可达到$v_2$，所以从v到w有两条不同有向路。所以，D中有弱圈，矛盾。因此，D是有根树。

# 2、有根树的性质

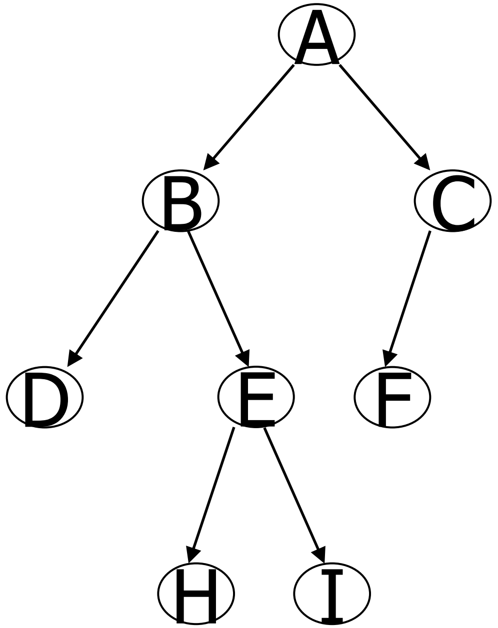
定义10.5.3 有根树的反向树称为入树。

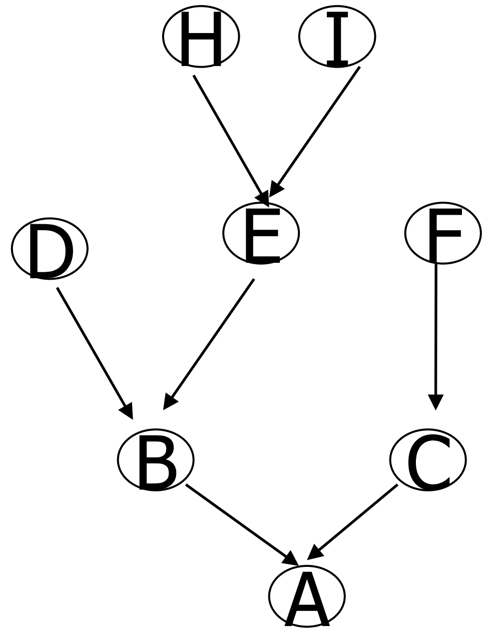



有根树又叫做出树，与之对应的称作入树

## 2、有根树的性质

定理10.5.2 有向图D是一个入树当且仅当D中没有弱圈且有顶点$v_0$使之其余的每个顶点均可达到$v_0$。

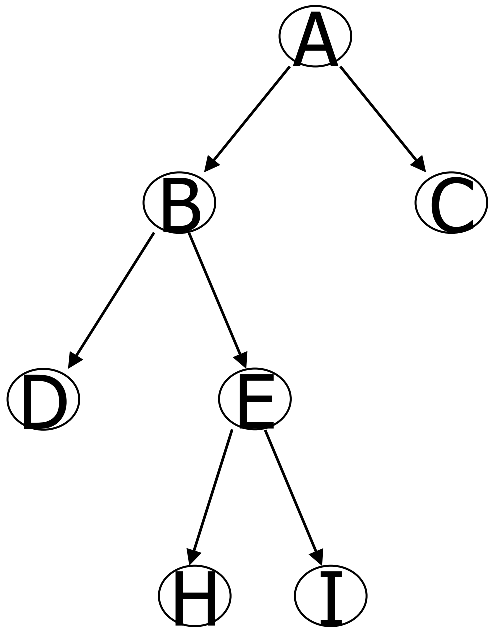



定理10.5.1 有向图D=(V,A)是一个有根树当且仅当D有一个顶点可以达到其他任一顶点且D中没有弱圈。

# 2、有根树的性质

例10.5.1 设D=(V,A)是一个有根树，其每一个顶点的出度不是0就是2.如果D有n个叶子，试求D的弧的条数。




解:

设顶点数为p，边数q=p−1

总出度为2(p−n)=2p−2n

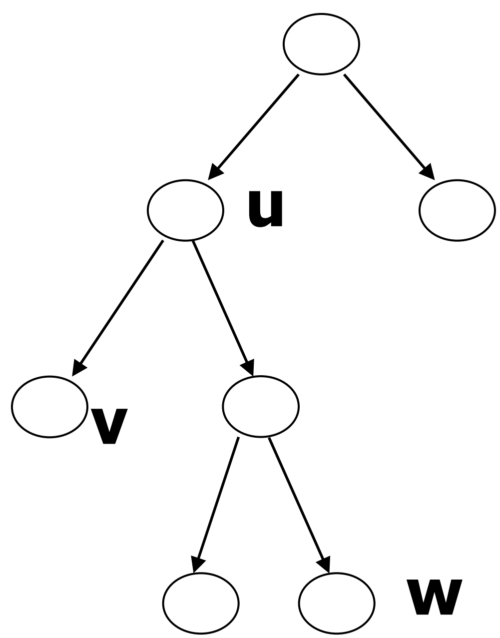
p−1=2p−2n

p=2n−1

q=2n−2

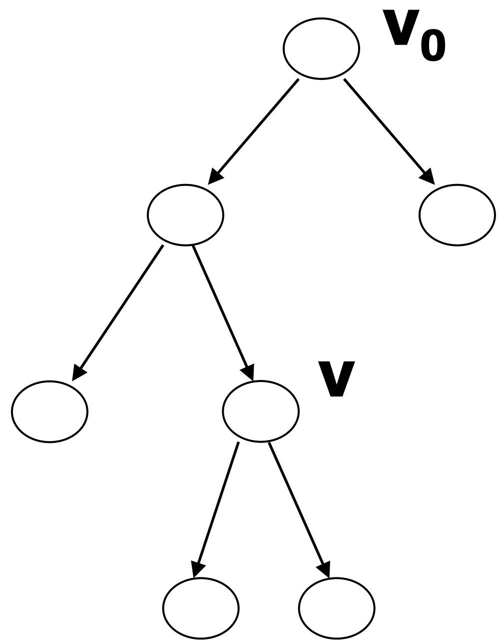
## 2、有根树的性质

定义10.5.4 设D =(V, A)是有根树。如果(u, v) ∈A, 则称v是u的儿子，而顶点u称为v的父亲。如果从顶点u能达到w，则w称为u的子孙，而u称为w的祖先。如果u是w的祖先且u≠w, 则u称为w的真祖先，而w称为u的真子孙。



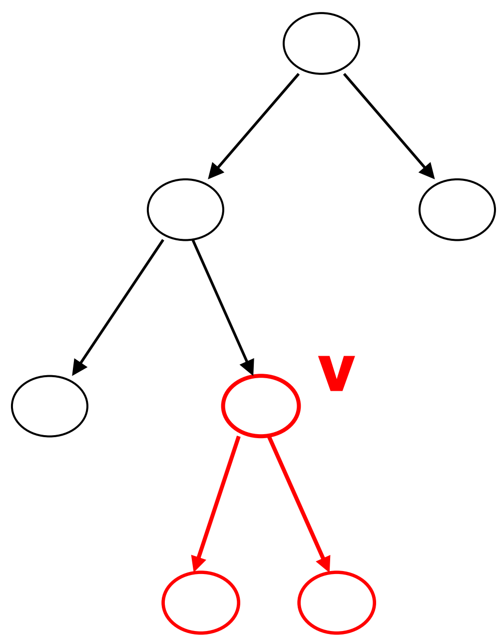
## 2、有根树的性质

定义10.5.5 设T =(V, A)是以$v_0$为根的有根树。$v_0$到顶点v的有向路的长度称为T的顶点v的深度。从顶点v到T的叶子节点的最长有向路的长度称为顶点v在T中的高度。根节点$v_0$的高度称为T的高度。



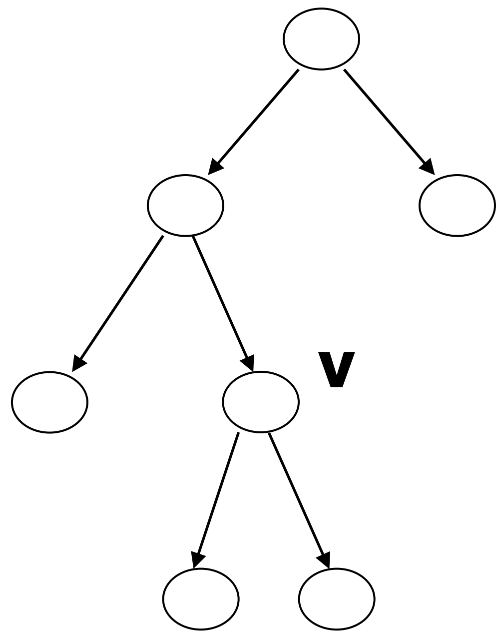
# 2、有根树的性质

定义10.5.6 设D =(V, A)是一个有根树。v是T的一个顶点，由v及其子孙所导出的T的子图称为T的以v为根的子树。



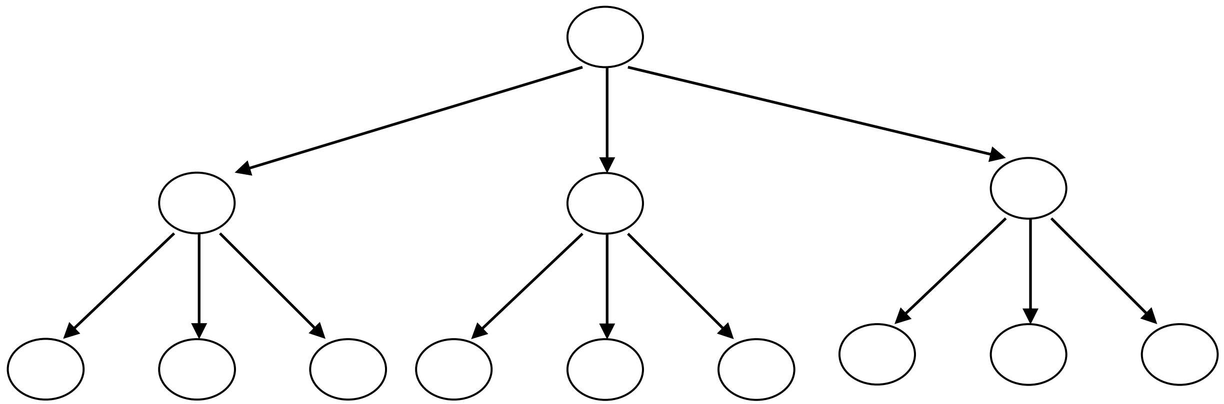
# 2、有根树的性质

定义10.5.7 设D =(V, A)是一个有根树。如果T的每个顶点的各个儿子排定了次序，则称T是有序树。



## 2、有根树的性质

定义10.5.8 有序树T称为m元有序树，如果T的每个顶点的出度≤m. 一个m元有序树T称为正则m元有序树，如果T的每个顶点的出度不是0就是m.

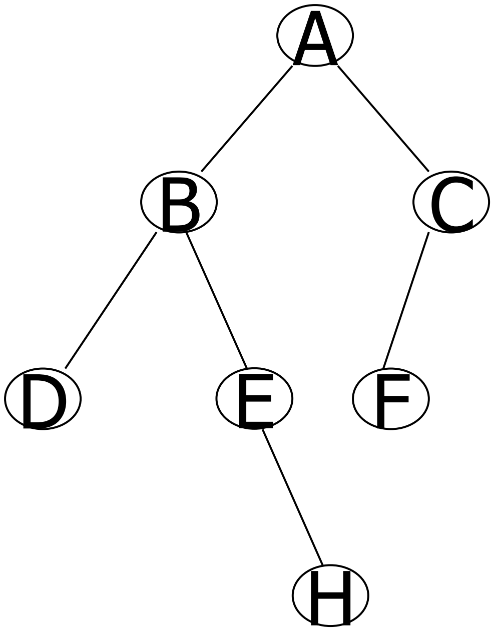


正则三元（叉）树

# 二叉树(二元树)

二叉树的定义

二叉树的性质



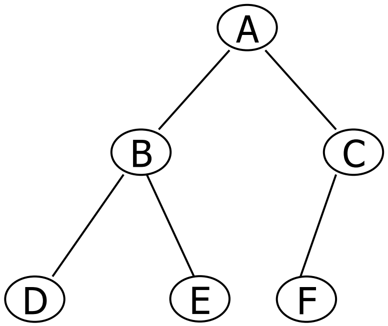
# 二叉树(二元树)

## 1. 定义

二叉树(Binary Tree)是有序树，它有如下特点。

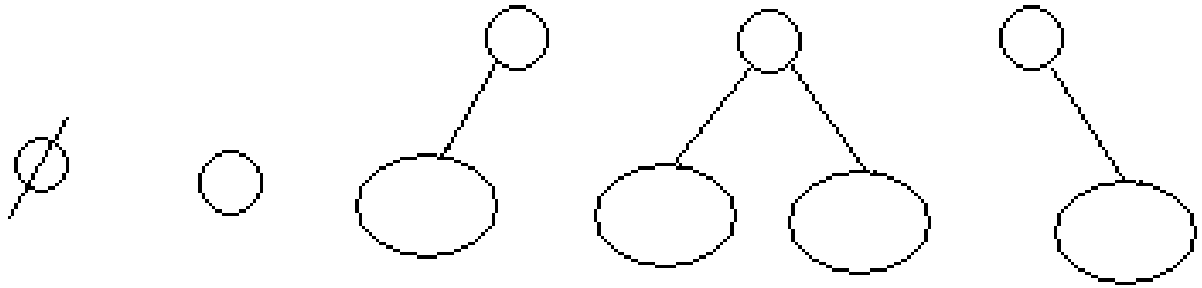
(1)任一顶点的儿子或被区分为左儿子，或被分为右儿子。特别是一个顶点若只有一个儿子时，也要指明它是左儿子还是右儿子；

(2)没有一个顶点有一个以上的左儿子或一个以上的右儿子。



# 二叉树(二元树)

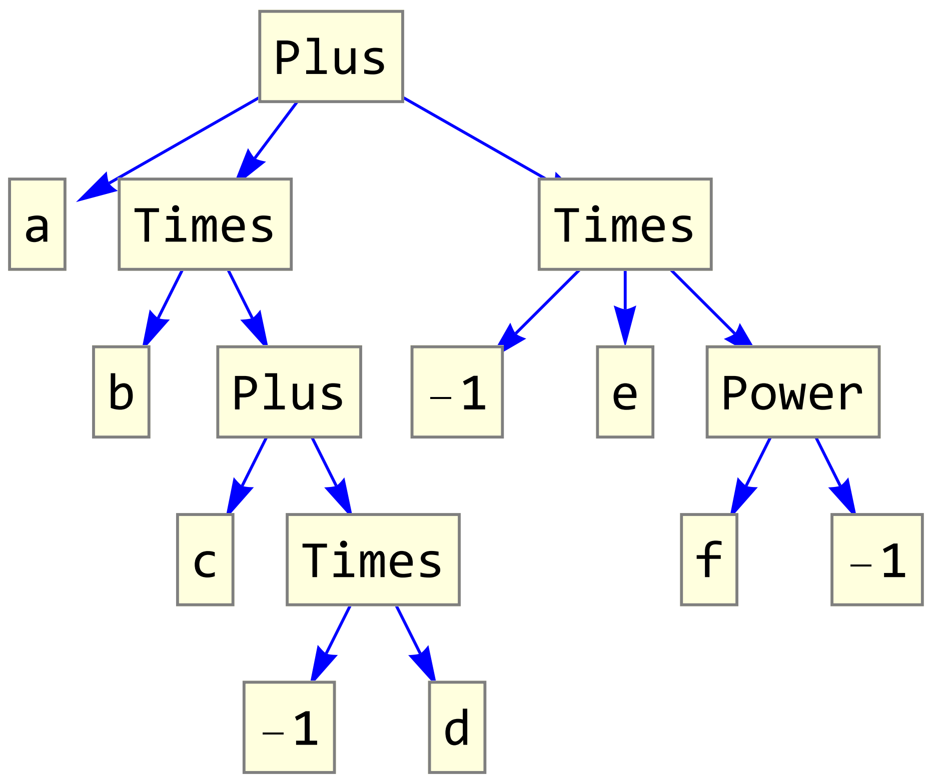
二叉树的五种基本形态



# 表达式的二叉树表示与求值

表达式的树型表示：**a+b×(c-d)-e/f**

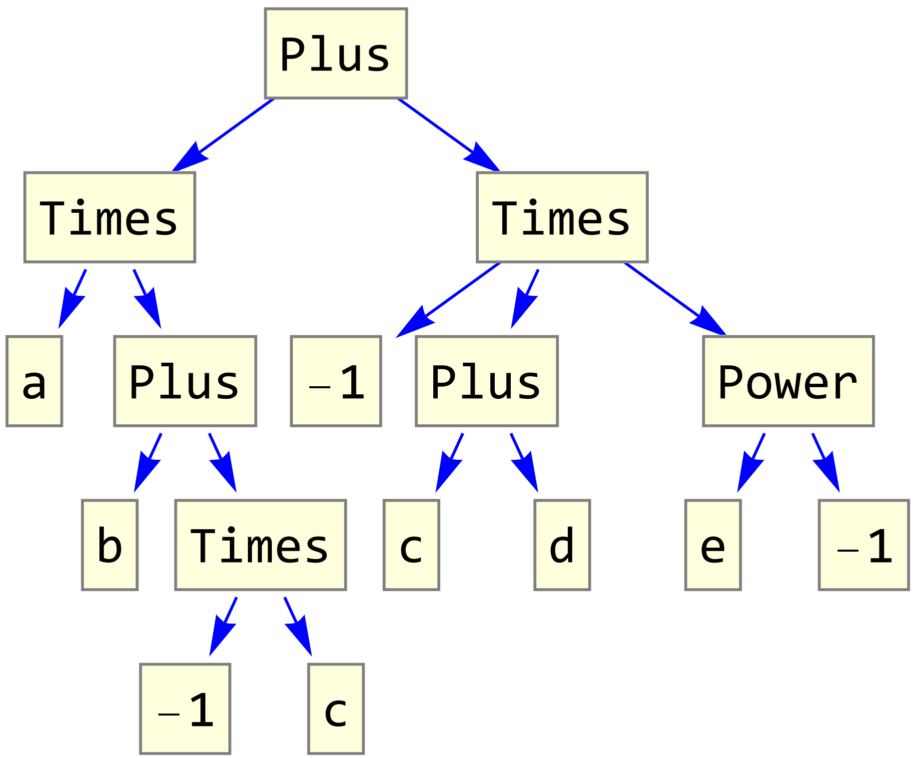
把每个操作数作为叶结点，操作符作为非叶结点（分支点/内节点）：



# 表达式的二叉树表示与求值

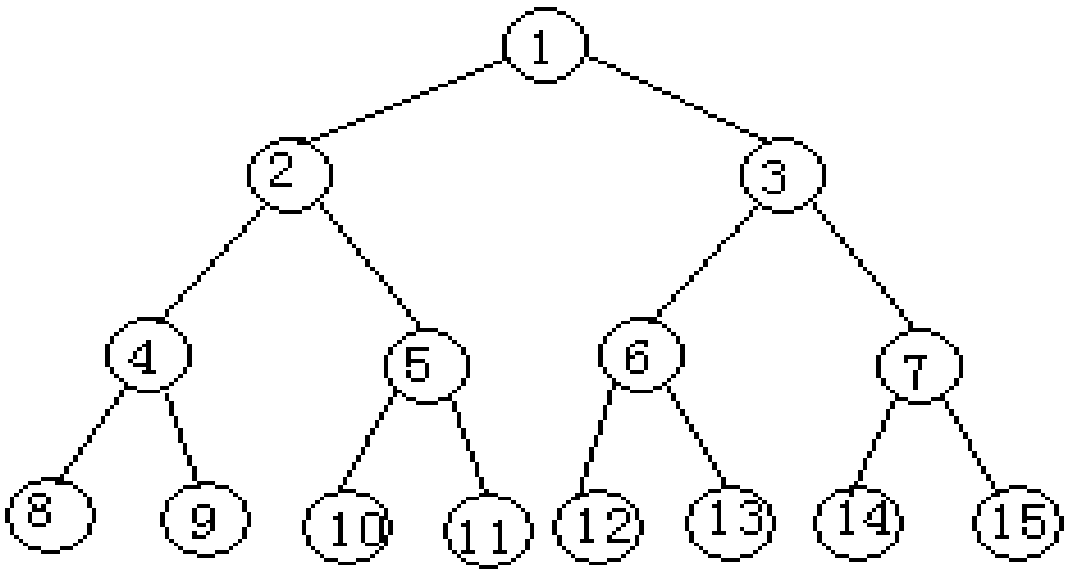
表达式的树型表示：**a× (b - c) -(c+ d)/e**

把每个操作数作为叶结点，操作符作为非叶结点（分支点/内节点）：



# 二叉树的性质

**性质1 在二叉树的第i(i≥1)层上至多有$2^i$个结点。**



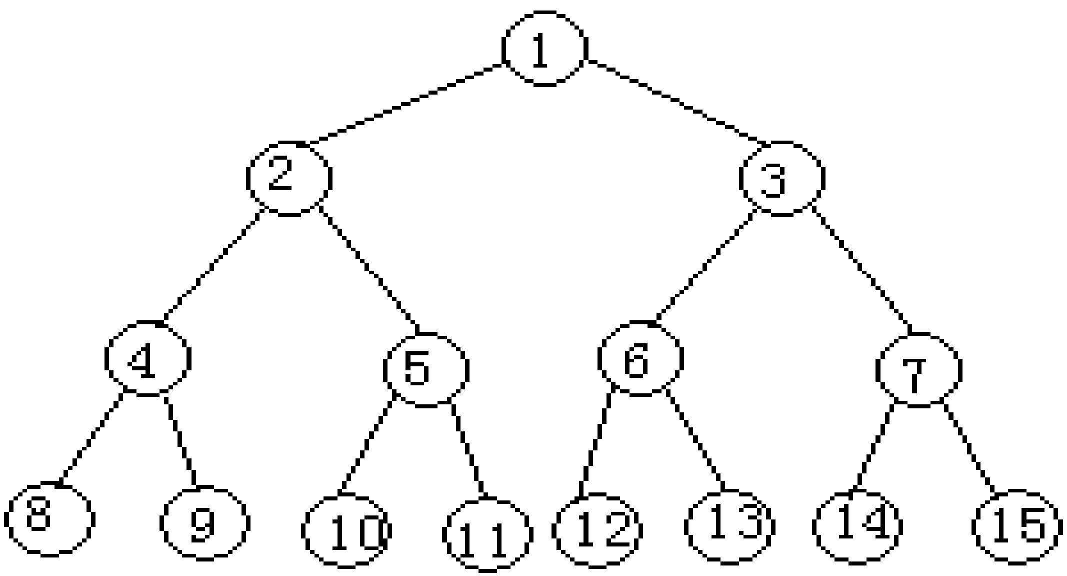

**性质2 深度为k的二叉树最多有$2^{k+1}$-1个结点。**

# 二叉树的性质

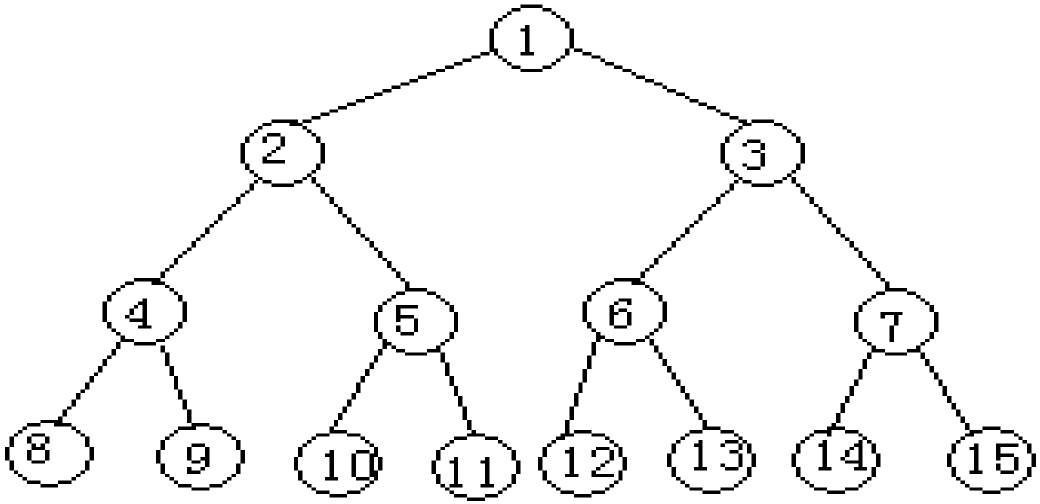
**满二叉树:**

如果一个二叉树的叶子结点都在最后一层上，且不存在度数为1的结点，则称该二叉树为满二叉树。

设高为k，则有$2^{k+1}-1$个结点。



# 二叉树的性质




**特点:**

**(1)对给定的高度，他有(最多)结点;**

**(2)不存在度数为( 1 )的结点;**

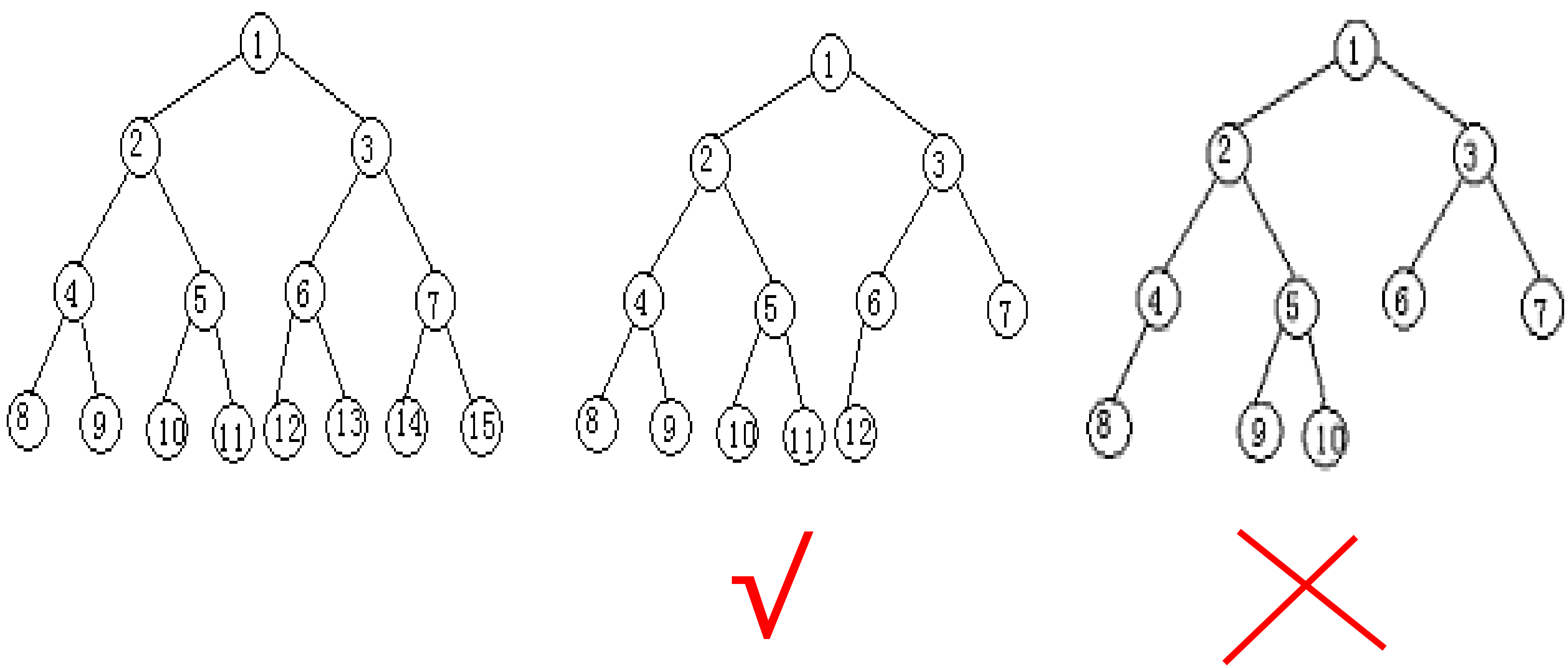
**(3)每个分支都有两棵高度(相同 )的子树;**

**(4)叶子结点都在( 最后一层上 )。**

# 二叉树的性质

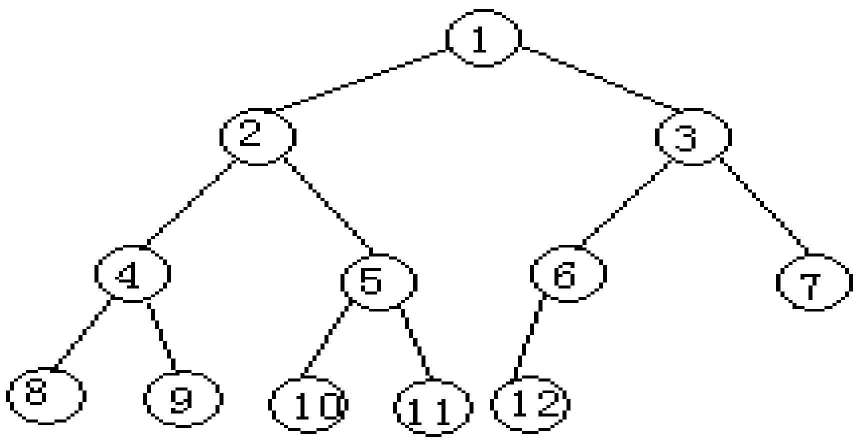
完全二叉树：注意！它是树的顺序存储结构的基础

如果存在一棵二叉树，对树中的结点自上而下、自左而右连续编号，若编号为i的结点与满二叉树中编号为i的结点的位置相同，则称此二叉树为完全二叉树。



# 二叉树的性质

完全二叉树的特点:




特点:

a.叶子结点只可能在(层数最大的两层上)出现;

b.对任一结点，若有右子树，则( 必有 )左子树。

# 二叉树的性质

性质3 对任意二叉树T，如果其终端结点数为$n_0$(度数为0的结点数),$n_1$,$n_2$分别表示度数为1,2的结点个数，则$n_0=n_2+1$。

证明：从两个方面考虑 ①节点总数 ②总度数

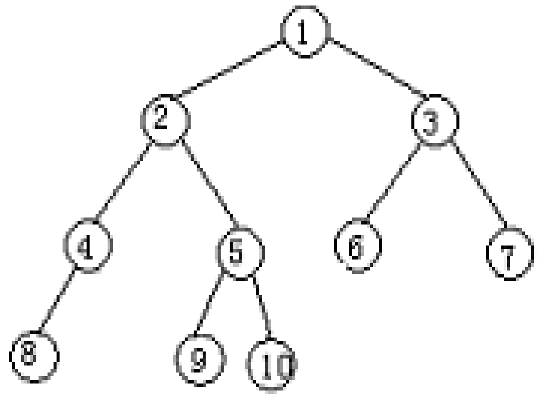
①节点总数：设n为二叉树T的结点总数，则有：

$n=n_0+n_1+n_2$

②总度数：总度数为n-1，

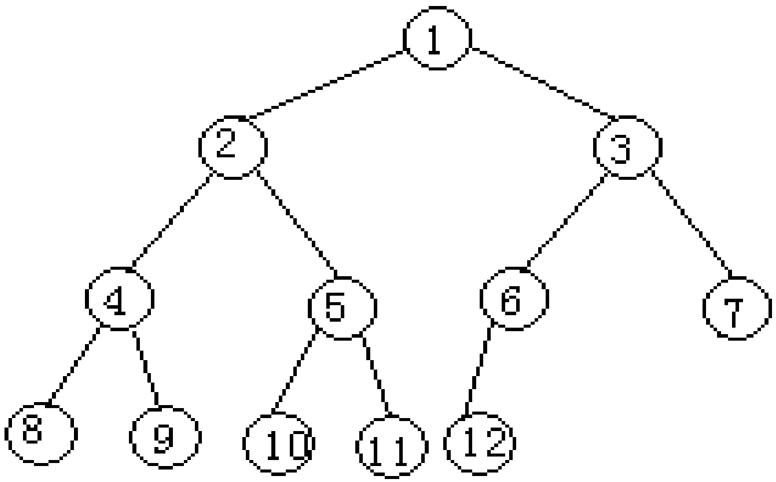
$n-1=n_1+2*n_2$

求得：$n_0=n_2+1$



# 二叉树的性质

性质4  具有n个结点的完全二叉树的深度为$\lfloor \log_2 n \rfloor$或者($\lceil \log_2(n+1) \rceil -1$)。




证明：设深度为k

①前k-1层的节点总数是$2^k-1$

②前k层最多有节点是$2^{k+1}-1$

∵ $2^k-1 < n \le 2^{k+1}-1$， $2^k \le n < 2^{k+1}$

∴ $k \le \log_2 n < k+1$，所以k是≤ $\log_2 n$的最大整数，即$k=\lfloor \log_2 n \rfloor$

同时，$2^k \le n < 2^{k+1}$也可写成 $2^k -1 < n \le 2^{k+1} -1$，即$n+1 \le 2^{k+1}$

∴ k是≥$\log_2(n+1)-1$的最小整数，即$k=\lceil \log2(n+1) \rceil -1$

# 10.6 判定树

## 本节主要问题

一、判定树的应用例子

二、判定树的性质

# 机器学习十大算法（各个时期有区别）

1、线性回归算法 Linear Regression

2、支持向量机算法 Support Vector Machine,SVM

3、最近邻居/k-近邻算法 K-Nearest Neighbors,KNN

4、逻辑回归算法 Logistic Regression

5、决策树算法 Decision Tree

6、k-平均算法 K-Means

7、随机森林算法 Random Forest

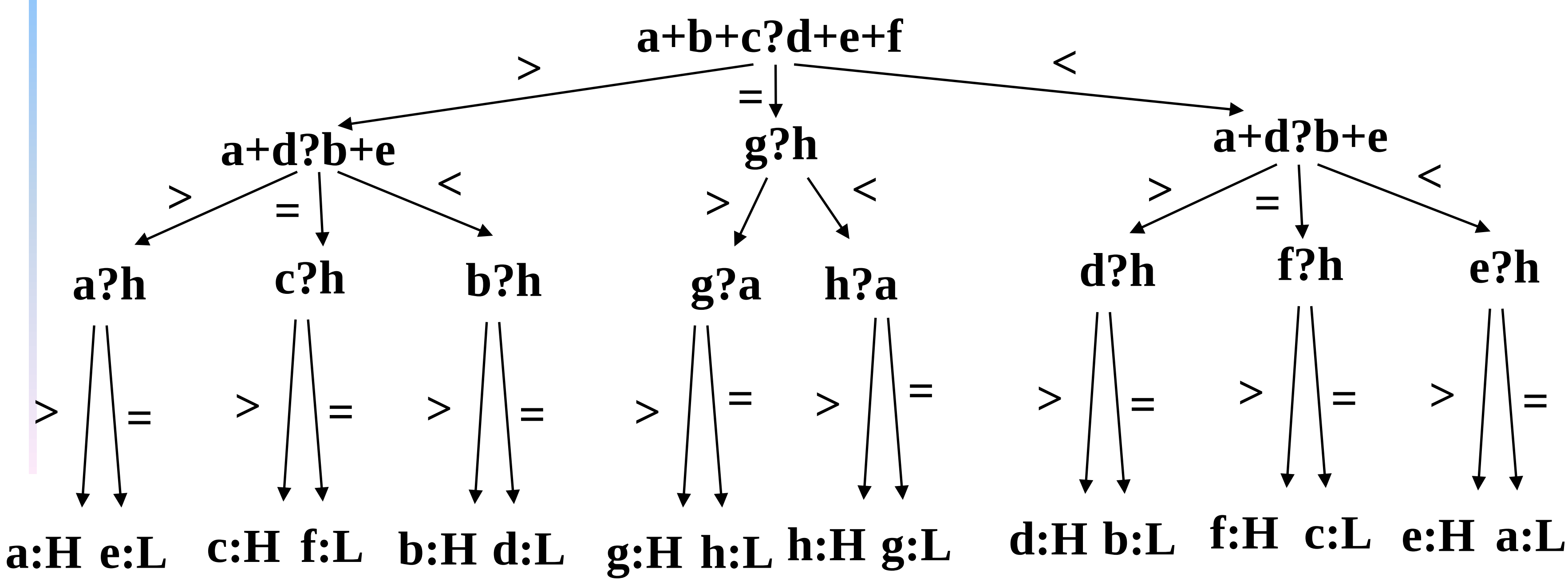
8、朴素贝叶斯算法 Naive Bayes

9、降维算法 Dimensional Reduction

10、梯度增强算法 Gradient Boosting

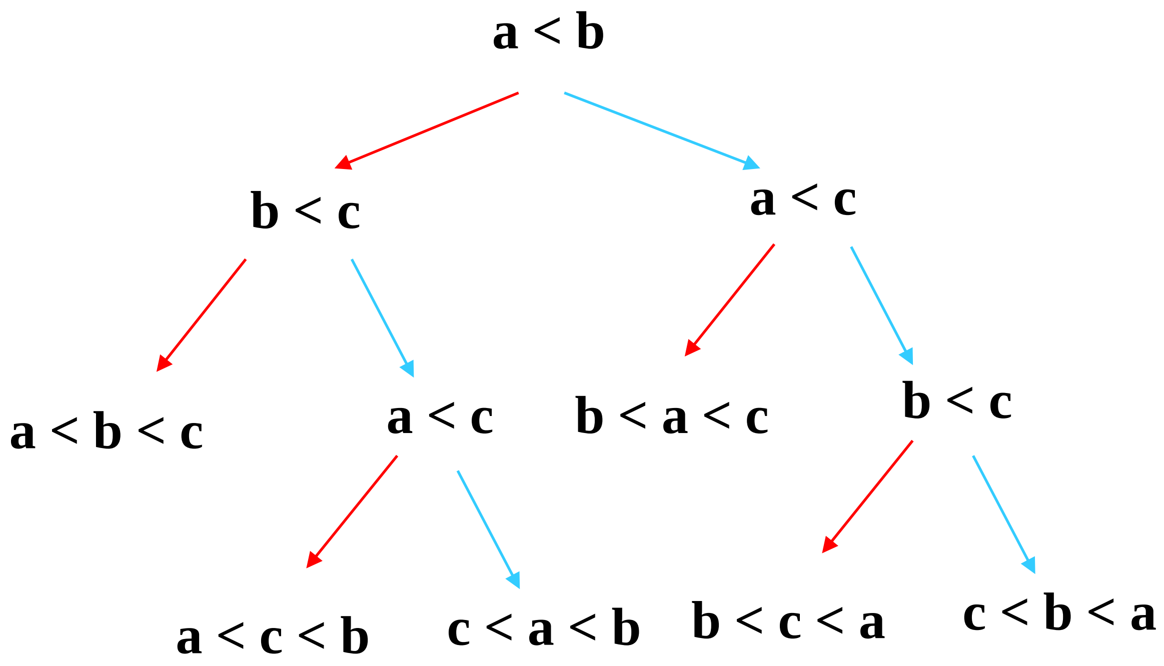
# 一、判定树的应用例子

八硬币问题：设有八枚硬币a, b, c, d, e, f, g, h。已知其中有一枚假的，硬币外表都一样，但假的重量不同。试用天平秤出哪个硬币是假的。



# 一、判定树的应用例子

a, b, c三个数进行比较。



# 二、判定树的性质

引理10.6.1 高为h的二元树至多有$2^h$个叶子节点。

证明：用归纳法，对高度h用归纳法，当h=1时，最多有两个叶节点，引理成立。

假设$h \le k$时成立，当$h=k+1$时，

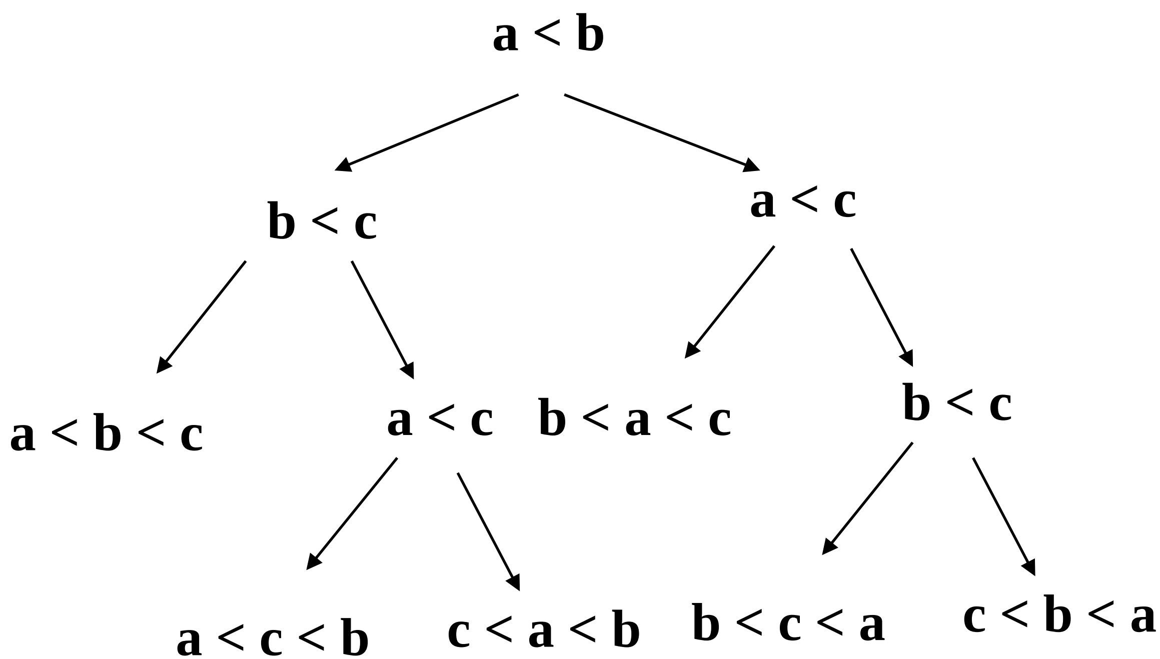
去掉根节点，则得到两棵高度最多为k的二元树。

由归纳假设：高度为k+1的二元树的叶子节点最多为$2 \times 2^h = 2^{h+1}$，因此，引理成立。

# 二、判定树的性质

定理10.6.1 将n个不同数排列的判定树的高度至少为$\log_2(n!)$。

a, b, c三个数进行比较。




引理10.6.1 高为h的二元树至多有$2^h$个叶子节点。

## 二、判定树的性质

定理10.6.1 将n个不同数排列的判定树的高度至少为$\log_2(n!)$。

证明:

因为n个不同的数有n!个排列,

这n个不同的数的排列结果可以是n!个排列的任意一个。

因此，对n个不同数进行排列的判定树种必有n!个叶子。

由引理10.6.1，判定树的高度至少为$\log_2(n!)$。

引理10.6.1 高为h的二元树至多有$2^h$个叶子节点。

# 二、判定树的性质

推论10.6.1 对n个数用比较分类算法排序时其最坏情况至少要求$O(n\log_2 n)$次比较。

证明：对n>1，有

$$n! \geqslant n(n-1)\ldots\lfloor n/2 \rfloor$$

$$\geqslant (n/2)^{n/2}$$

$$\log_2 n! \geq \left(\frac{n}{2}\right) \boldsymbol{log_2}\left(\frac{n}{2}\right) = \left(\frac{n}{2}\right)\log_2 n - \left(\frac{n}{2}\right)$$

当$n \geq 4$时，$\log_2 n! \geq \left(\frac{n}{4}\right)\log_2 n$

快速排序、堆排序、归并排序的平均时间复杂性是$O(n\log_2 n)$。

引理10.6.1 高为h的二元树至多有$2^h$个叶子节点。

# 10.7 比赛图及其应用

## 本节主要问题

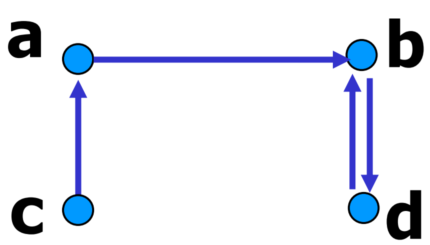
一、比赛图的定义

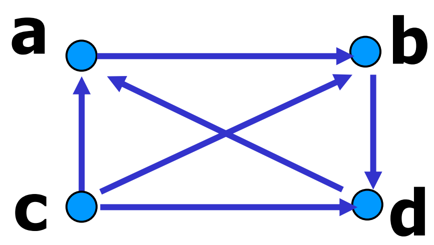
二、比赛图的性质

三、比赛图的应用

# 1、比赛图的定义

定义10.7.1 一个比赛图是一个定向完全图，即任何两个顶点间有且仅有一条弧。

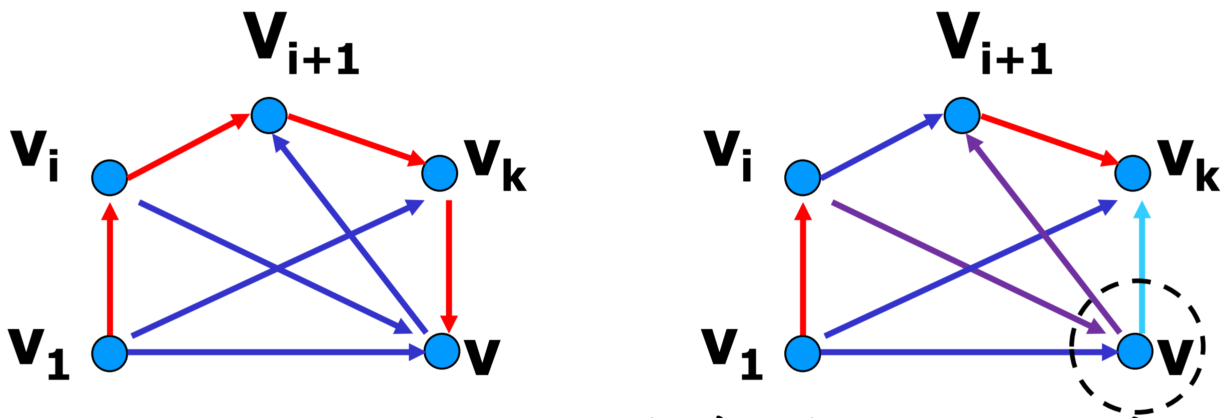



一组选手间每两个选手都进行比赛，顶点表示选手，两个顶点间画一条从胜者到负者的弧。

# 2、比赛图的性质

定理10.7.1 每一个比赛图有条生成有向路。




证明：设D = (V, A)是一个有p个顶点的比赛图，令P: $v_1 v_2 \ldots v_k$是一条最长的有向路。

假如k < p, 则$\exists v \in V$使得v不在路P上。

由D是比赛图且P是最长路，所以，$(v, v_1) \notin A$，$(v_k, v) \notin A$。

于是，必有$(v_1, v) \in A$，$(v, v_k) \in A$。

令$v_i$是路P上从$v_1$到$v_k$的最后一个使得$(v_i, v) \in A$的顶点，于是$(v, v_{i+1}), \ldots, (v, v_k) \in A$.

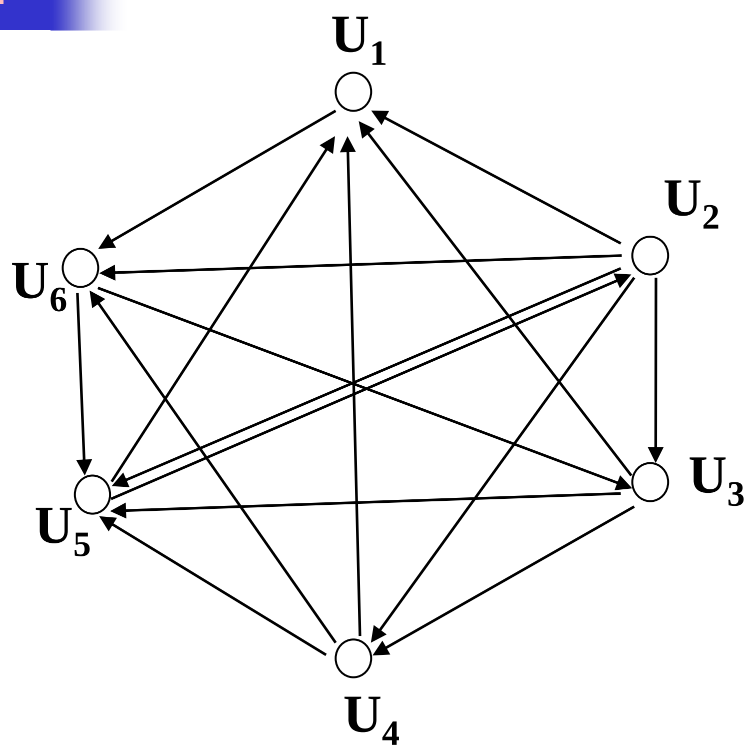
于是：$v_1 v_2 \ldots v_i v v_{i+1} \ldots v_k$是一条比P更长的路。

# 3、比赛图的应用

工件排序问题：设有某台机器必须加工多种工件：$J_1, J_2, ..., J_6$；在一种工件加工完毕之后，为了加工下一种工件，机器必须进行调整。如果从工件$J_i$到工件$J_j$的调整时间为$t_{ij}$，求这些工件的一个排序，使整个机器的调整时间最少。

| | $j_1$ | $j_2$ | $j_3$ | $j_4$ | $j_5$ | $j_6$ |
|---|---|---|---|---|---|---|
| $j_1$ | 0 | 5 | 3 | 4 | 2 | 1 |
| $j_2$ | 1 | 0 | 1 | 2 | 3 | 2 |
| $j_3$ | 2 | 5 | 0 | 1 | 2 | 3 |
| $j_4$ | 1 | 4 | 4 | 0 | 1 | 2 |
| $j_5$ | 1 | 3 | 4 | 5 | 0 | 5 |
| $j_6$ | 4 | 4 | 2 | 3 | 1 | 0 |

# 3、比赛图的应用




| | $j_1$ | $j_2$ | $j_3$ | $j_4$ | $j_5$ | $j_6$ |
|---|---|---|---|---|---|---|
| $j_1$ | 0 | 5 | 3 | 4 | 2 | 1 |
| $j_2$ | 1 | 0 | 1 | 2 | 3 | 2 |
| $j_3$ | 2 | 5 | 0 | 1 | 2 | 3 |
| $j_4$ | 1 | 4 | 4 | 0 | 1 | 2 |
| $j_5$ | 1 | 3 | 4 | 5 | 0 | 5 |
| $j_6$ | 4 | 4 | 2 | 3 | 1 | 0 |

1. 任选一个顶点，例如$U_1$，从其出边中选最小值，例如$U_6$.

2. 从$U_6$中按第一步选剩余顶点。若找不到符合要求的边，回退。

3. 重复1, 2步骤。 $U_1U_6U_5U_2U_3U_4$

# 3、比赛图的应用

工件排序问题　设有某台机器必须加工多种工件：$J_1, J_2, ..., J_n$；在一种工件加工完毕之后，为了加工下一种工件，机器必须进行调整。如果从工件$J_i$到工件$J_j$的调整时间为$t_{ij}$，求这些工件的一个排序，使整个机器的调整时间最少。

第1步：构造顶点$v_1, v_2, ..., v_n$的有向图$D=(V, A)$，使得$(v_i, v_j)\in A$ 当且仅当$t_{ij}\le t_{ji}$。

由此，D中含有一个生成比赛图。

第2步：求D的有向哈密顿路$v_{i1}v_{i2}...v_{in}$。按此哈密顿路安排工件的排序。