二叉树、平衡二叉树、完全二叉树、满二叉树 - Terry的IT世界 - CSDN博客

基本概念

结点的层次（Level）从根开始定义，根为第一层，根的孩子为第二层。

二叉树的高度：树中结点的最大层次称为树的深度（Depth）或高度。

二叉树

在计算机科学中，二叉树是每个结点最多有两个子树的有序树。通常子树的根被称作“左子树”（left subtree）和“右子树”（right subtree）。二叉树常被用作二叉查找树和二叉堆。二叉树的每个结点至多只有二棵子树(不存在度大于2的结点)，二叉树的子树有左右之分，次序不能颠倒。二叉树的第i层至多有2的（i-1）次方个结点；深度为k的二叉树至多有2的k次 － 1个结点；对任何一棵二叉树T，如果其终端结点数为*n*0，度为2的结点数为*n*2，则*n*0 = *n*2 + 1。

树和二叉树的2个主要差别：

1. 树中结点的最大度数没有限制，而二叉树结点的最大度数为2；
2. 树的结点无左、右之分，而二叉树的结点有左、右之分。……

树是一种重要的非线性**数据结构**，直观地看，它是数据元素（在树中称为结点）按分支关系组织起来的结构，很象自然界中的树那样。树结构在客观世界中广泛存在，如人类社会的族谱和各种社会组织机构都可用树形象表示。树在计算机领域中也得到广泛应用，如在编译源程序如下时，可用树表示源源程序如下的语法结构。又如在**数据库**系统中，树型结构也是信息的重要组织形式之一。一切具有层次关系的问题都可用树来描述。

一、树的概述

树结构的特点是：它的每一个结点都可以有不止一个直接后继，除根结点外的所有结点都有且只有一个直接前趋。以下具体地给出树的定义及树的数据结构表示。

（一）树的定义

树是由一个或多个结点组成的有限集合，其中：

1.必有一个特定的称为根(ROOT)的结点；

2.剩下的结点被分成n>=0个互不相交的集合T1、T2、.....Tn，而且， 这些集合的每一个又都是树。树T1、T2、.....Tn被称作根的子树(Subtree)。

树的递归定义如下：（1）至少有一个结点（称为根）（2）其它是互不相交的子树

1.树的度——也即是宽度，简单地说，就是结点的分支数。以组成该树各结点中最大的度作为该树的度，如上图的树，其度为3;树中度为零的结点称为叶结点或终端结点。树中度不为零的结点称为分枝结点或非终端结点。除根结点外的分枝结点统称为内部结点。

2.树的深度——组成该树各结点的最大层次，如上图，其深度为4；

3.森林——指若干棵互不相交的树的集合，如上图，去掉根结点A，其原来的二棵子树T1、T2、T3的集合{T1,T2,T3}就为森林；

4.有序树——指树中同层结点从左到右有次序排列，它们之间的次序不能互换，这样的树称为有序树，否则称为无序树。

5.树的表示

树的表示方法有许多，常用的方法是用括号：先将根结点放入一对圆括号中，然后把它的子树由左至右的顺序放入括号中，而对子树也采用同样的方法处理；同层子树与它的根结点用圆括号括起来，同层子树之间用逗号隔开，最后用闭括号括起来。如上图可写成如下形式：

(A(B(E(K,L),F),C(G),D(H(M),I,J)))

5. 2 二叉树

1.二叉树的基本形态：

二叉树也是递归定义的，其结点有左右子树之分，逻辑上二叉树有五种基本形态：

(1)空二叉树——(a)；

(2)只有一个根结点的二叉树——(b)；

(3)只有右子树——(c)；

(4)只有左子树——(d)；

(5)完全二叉树——(e)

注意：尽管二叉树与树有许多相似之处，但二叉树不是树的特殊情形。

2.两个重要的概念：

(1)完全二叉树——只有最下面的两层结点度小于2，并且最下面一层的结点都集中在该层最左边的若干位置的二叉树；

(2)满二叉树——除了叶结点外每一个结点都有左右子叶且叶结点都处在最底层的二叉树,。

3.二叉树的性质

(1) 在二叉树中，第i层的结点总数不超过2^(i-1)；

(2) 深度为h的二叉树最多有2^h-1个结点(h>=1)，最少有h个结点；

(3) 对于任意一棵二叉树，如果其叶结点数为N0，而度数为2的结点总数为N2，

则N0=N2+1；

(4) 具有n个结点的完全二叉树的深度为int（log2n）+1

(5)有N个结点的完全二叉树各结点如果用顺序方式存储，则结点之间有如下关系：

若I为结点编号则 如果I<>1，则其父结点的编号为I/2；

如果2*I< NII>N，则无左儿子；

如果2*I+1< NII>N，则无右儿子。

(6)给定N个节点，能构成h(N)种不同的二叉树。

h(N)为卡特兰数的第N项。h(n)=C(n,2*n)/(n+1)。

4.二叉树的存储结构：

(1) 顺序存储方式

```
type node=record
data:datatype
l,r:integer;
end;
var tr:array[1..n] of node;
```

(2)链表存储方式，如：

```
type btree=^node;
node=record
data:datatye;
lchild,rchild:btree;
end;
```

5.普通树转换成二叉树：凡是兄弟就用线连起来，然后去掉父亲到儿子的连线，只留下父母到其第一个子女的连线。

二叉树很象一株倒悬着的树，从树根到大分枝、小分枝、直到叶子把数据联系起来，这种数据结构就叫做树结构，简称树。树中每个分叉点称为结点，起始结点称为树根，任意两个结点间的连接关系称为树枝，结点下面不再有分枝称为树叶。结点的前趋结点称为该结点的"双亲"，结点的后趋结点称为该结点的"子女"或"孩子"，同一结点的"子女"之间互称"兄弟"。

二叉树：二叉树是一种十分重要的树型结构。它的特点是，树中的每个结点最多只有两棵子树，即树中任何结点的度数不得大于 2 。二叉树的子树有左右之分，而且，子树的左右次序是重要的，即使在只有一棵子树的情况下，也应分清是左子树还是右子树。定义：二叉树是结点的有限集合，这个集合或是空的，或是由一个根结点和两棵互不相交的称之为左子树和右子树的二叉树组成。

（三）完全二叉树

对满二叉树，从第一层的结点(即根)开始，由下而上，由左及右，按顺序结点编号，便得到满二叉树的一个顺序表示。据此编号，完全二叉树定义如下：一棵具有 n 个结点，深度为K的二叉树，当且仅当所有结点对应于深度为K的满二叉树中编号由 1 至 n 的那些结点时，该二叉树便是完全二叉树。图4是一棵完全二叉树。

平衡二叉树

当且仅当两个子树的高度差不超过1时，这个树是平衡二叉树。（同时是排序二叉树）

平衡二叉树，又称AVL树。它或者是一棵空树，或者是具有下列性质的二叉树：它的左子树和右子树都是平衡二叉树，且左子树和右子树的高度之差之差的绝对值不超过1.。

常用**算法**有：红黑树、AVL树、Treap等。

平衡二叉树的调整方法

平衡二叉树是在构造二叉排序树的过程中，每当插入一个新结点时，首先检查是否因插入新结点而破坏了二叉排序树的平衡性，若是，则找出其中的最小不平衡子树，在保持二叉排序树特性的前提下，调整最小不平衡子树中各结点之间的链接关系，进行相应的旋转，使之成为新的平衡子树。具体步骤如下：

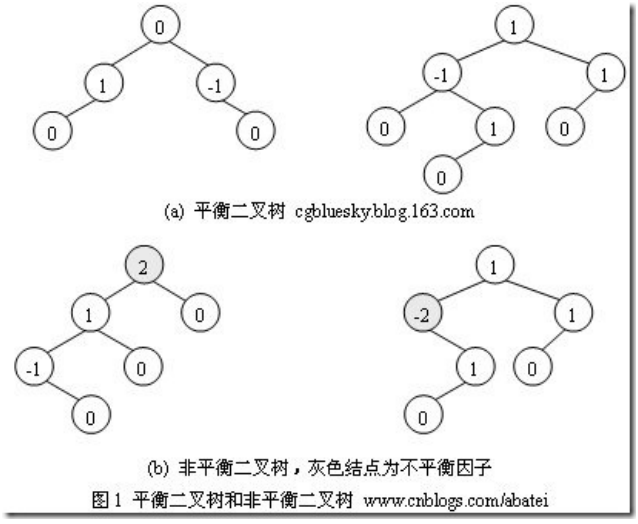
⑴ 每当插入一个新结点，从该结点开始向上计算各结点的平衡因子，即计算该结点的祖先结点的平衡因子，若该结点的祖先结点的平衡因子的绝对值均不超过1，则平衡二叉树没有失去平衡，继续插入结点；

⑵ 若插入结点的某祖先结点的平衡因子的绝对值大于1，则找出其中最小不平衡子树的根结点；

⑶ 判断新插入的结点与最小不平衡子树的根结点的关系，确定是哪种类型的调整；

⑷ 如果是LL型或RR型，只需应用扁担原理旋转一次，在旋转过程中，如果出现冲突，应用旋转优先原则调整冲突；如果是LR型或LR型，则需应用扁担原理旋转两次，第一次最小不平衡子树的根结点先不动，调整插入结点所在子树，第二次再调整最小不平衡子树，在旋转过程中，如果出现冲突，应用旋转优先原则调整冲突；

⑸ 计算调整后的平衡二叉树中各结点的平衡因子，检验是否因为旋转而破坏其他结点的平衡因子，以及调整后的平衡二叉树中是否存在平衡因子大于1的结点。



（b）左边的图 左子数的高度为3，右子树的高度为1，相差超过1

（b）右边的图 -2的左子树高度为0　右子树的高度为2，相差超过1

完全二叉树(Complete Binary Tree)

完全二叉树

若设二叉树的高度为h，除第 h 层外，其它各层 (1～h-1) 的结点数都达到最大个数，第 h 层从右向左连续缺若干结点，这就是完全二叉树。

完全二叉树特点

一、叶子结点只可能在最大的两层上出现,对任意结点，若其右分支下的子孙最大层次为L，则其左分支下的子孙的最大层次必为L 或 L+1;

二、出于简便起见,完全二叉树通常采用数组而不是链表存储,其存储结构如下:

```
var tree:array[1..n]of longint;{n:integer;n>=1}
```

对于tree[i]，有如下特点：

（1）若i为奇数且i>1，那么tree[i]的左兄弟为tree[i-1]；

（2）若i为偶数且i

（3）若i>1，tree[i]的双亲为tree[i div 2]；

（4）若2*i< ntreeitreeii=″n，那么tree[i]的右孩子为tree[2*i+1]；

（5）若i>n div 2,那么tree[i]为叶子结点（对应于（3））；

（6）若i< n- div treeip>

特别地：满二叉树一定是完全二叉树，完全二叉树不一定是满二叉树

完全二叉树叶子节点的算法

如果一棵具有n个结点的深度为k的二叉树，它的每一个结点都与深度为k的满二叉树中编号为1~n的结点一一对应，这棵二叉树称为完全二叉树。

可以根据公式进行推导，假设n0是度为0的结点总数（即叶子结点数），n1是度为1的结点总数，n2是度为2的结点总数，由二叉树的性质可知：n0＝n2＋1，则n= n0＋n1＋n2（其中n为完全二叉树的结点总数），由上述公式把n2消去得：n= 2n0+n1－1，由于完全二叉树中度为1的结点数只有两种可能0或1，由此得到n0＝（n＋1）/2或n0＝n/2，合并成一个公式：n0＝（n＋1）/2 ，就可根据完全二叉树的结点总数计算出叶子结点数。

满二叉树

一棵深度为k，且有2的(k)次方－1个节点的二叉树 特点：每一层上的结点数都是最大结点数

完全二叉树的定义：深度为k，有n个结点的二叉树当且仅当其每一个结点都与深度为k的满二叉树中编号从1至n的结点一一对应时，称为完全二叉树。 特点：叶子结点只可能在层次最大的两层上出现；对任一结点，若其右分支下子孙的最大层次为l，则其左分支下子孙的最大层次必为l 或l+1 满二叉树：一棵深度为k，且有2的(k)次方－1个节点的二叉树 特点：每一层上的结点数都是最大结点数 满二叉树肯定是完全二叉树完全二叉树不一定是满二叉树