参考文档：

<https://www.jianshu.com/p/bbd5d5b4d1a3>

23查找树

为了保证查找树的平衡性，我们需要一些灵活性，因此在这里我们允许树中的一个结点保存多个键。

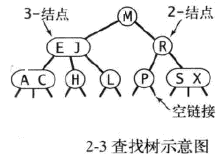
确切地说，我们将一棵标准的二叉查找树中的结点称为**2-结点**(含有一个键和两条链接)。现在我们引入**3-结点**，它含有两个键和三条链接。

2-结点和3-结点中的每条链接都对应着其中保存的键所分隔产生的一个区间。

一棵**2-3查找树**或为一棵空树，或由以下结点组成：

* 2-结点，含有一个键(及其对应的值)和两条链接，左链接指向的2-3树中的键都小于该结点，右链接指向的2-3树中的键都大于该结点。
* 3-结点，含有两个键(及其对应的值)和三条链接，左链接指向的2-3树中的键都小于该结点，中链接指向的2-3树中的键都位于该结点的两个键之间，右链接指向的2-3树中的键都大于该结点。

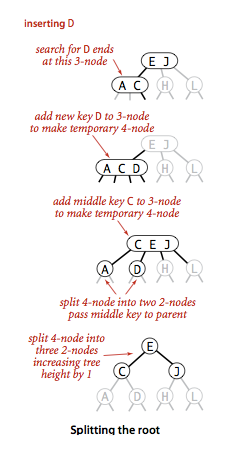
我们将指向一棵空树的链接称为空链接。



注意二叉查找树是向下生长，而2-3树是向上生长。

构建树的过程不同，二叉树一直往下生长。2-3树，如果子节点变成为4节点，会向父节点生长。这样来达到自平衡。

2-3查找树，在子节点插入key，然后在向上生长，来实现平衡。如果根节点达到了4节点，需要分解，那就要往上生长一级。插入节点分很多中情况。下面介绍一种父节点是3节点的3节点插入过程。



二叉树最坏的情况就是链表。2-3树会自动平衡，不会存在这种情况。

2-节点，含有一个键和两条链接。  
3-节点，含有两个键和三条链接  
一棵完美平衡的2-3树中所有空链到根节点的距离都应该是相同的。

红黑树

## 2.1定义

1.红链接均为左链接  
2.没有任何一个节点同时和两条红链接相连  
3.该树是完美黑平衡的，即任意空链接到根节点上的黑链接数量相同  
红链接：将两个2-节点连接起来构成3-节点  
黑链接：2-3树中的普通链接

红黑树其实是2-3树的一种只含2节点的表现形式。还是二叉树节点大于左子节点，小于右子节点。

**红黑树的性质**

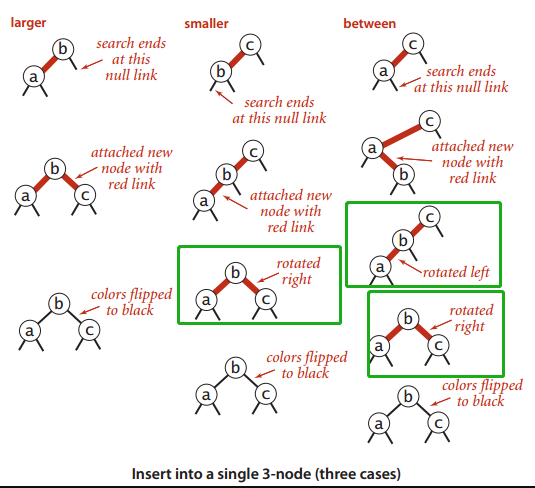
1. 每个节点要么是红色，要么是黑色（2-3树节点要么是2节点要么是3节点）
2. 根节点必须是黑色
3. 红色节点不能连续（红色节点的子和父不能为红）（不能有4节点。注意2-3的4节点是临时的）
4. 对于每个节点，从该点至null（树的尾端）的任何路径，都含有相同个数的黑色节点。

新节点都用红链接连接。

如果新节点是右链接连接的，需要左旋到左边。

如果新节点导致不平衡，需要右旋转来实现平衡。

如果一个节点有2个子红链，需要color flip to black。（父节点变成红色，子节点变成黑色）



旋转的实质就是父子节点变换位置。

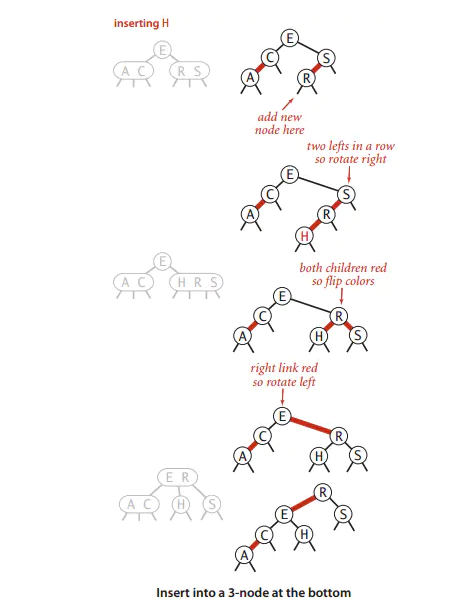
左旋就是父节点变成左节点，有节点变成父节点

右旋就是节点的父节点变成右子节点。

### 2.5.5 根节点永远是黑色的

每次插入都会将根节点设置成黑色，每当根节点由红变黑的时候，黑链接高度就会加1

向树底部的3-节点插入新节点。



红链接向上传递

