14.2 红黑树的删除

将红黑树内的某一个节点删除。需要执行的操作依次是：

a. 将红黑树当作一颗二叉查找树，将该节点从二叉查找树中删除；

b. 通过"旋转和重新着色"等一系列来修正该树，使之重新成为一棵红黑树。

详细描述如下：

假设y是要删除的节点，x是用来替换的节点（当y是叶节点时，x是NULL节点，当做黑色节点处理）

- 1. 执行标准的BST删除操作

  在标准的 BST 删除操作中，我们最终都会以删除一个叶子结点或者只有一个孩子的结点而结束（对于内部节点，就是要删除结点左右孩子都存在的情况，最终都会退化到删除结点是叶子结点或者是只有一个孩子的情况）。所以我们仅需要处理被删除结点是叶结点或者仅有一个孩子的情况。

- 2. 简单情况：y或者x有一个是红色节点

  如果 x 或者 y 是红色，我们将替换结点 y 的结点 x 标记为黑色结点（这样黑高就不会变化）。注意这里是 x 或者 y 是红色结点，因为在一棵红黑树中，是不允许有两个相邻的红色结点的，而结点 y 是结点 x 的父结点，因此只能是 x 或者 y 是红色结点。

- 3. 复杂情况：y和x都是黑色节点

  双黑节点的定义：

  当要删除结点 y 和孩子结点 x 都是黑色结点，删除结点 y ，导致结点 x 变为双黑结点。当 x 变成双黑结点时，我们的主要任务将变成将该双黑结点 x 变成普通的单黑结点。一定要特别注意，NULL结点为黑色结点 ， 所以删除黑色的叶子结点就会产生一个双黑结点。

  3.1 当前节点x是双黑节点且不是根节点

  （a） x的兄弟节点w是黑色且w的孩子节点至少有一个是红色的

  （b）x的兄弟节点w是黑色且它的两个孩子都是黑色的

  （c）x的兄弟节点w是红色

  3.2 当前节点x是双黑节点且是根节点

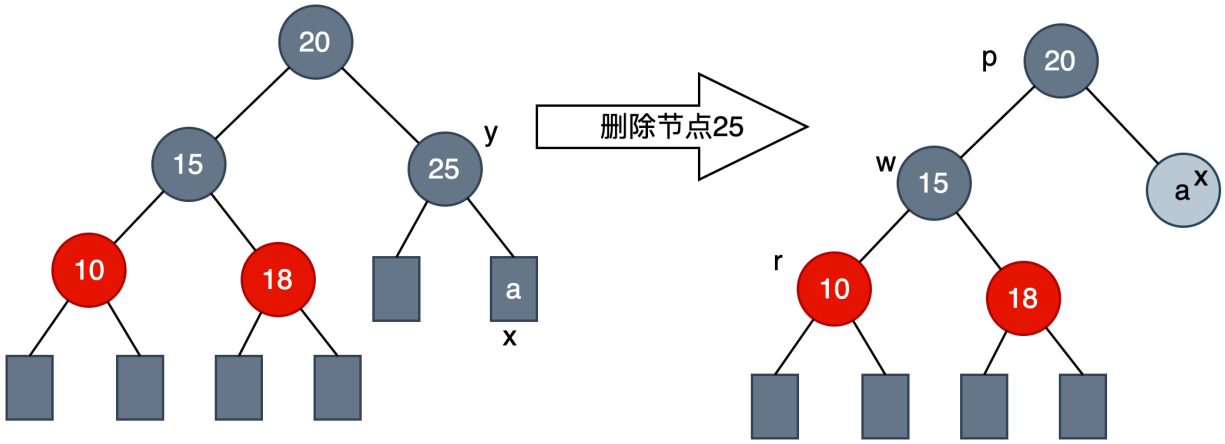
当前节点x是双黑节点且不是根节点

a. x的兄弟节点w是黑色且w的孩子节点至少有一个红色

对于这种情况，需要对x的兄弟节点w进行旋转操作，将w的一个红色节点用r表示，x和w的父节点用p表示。

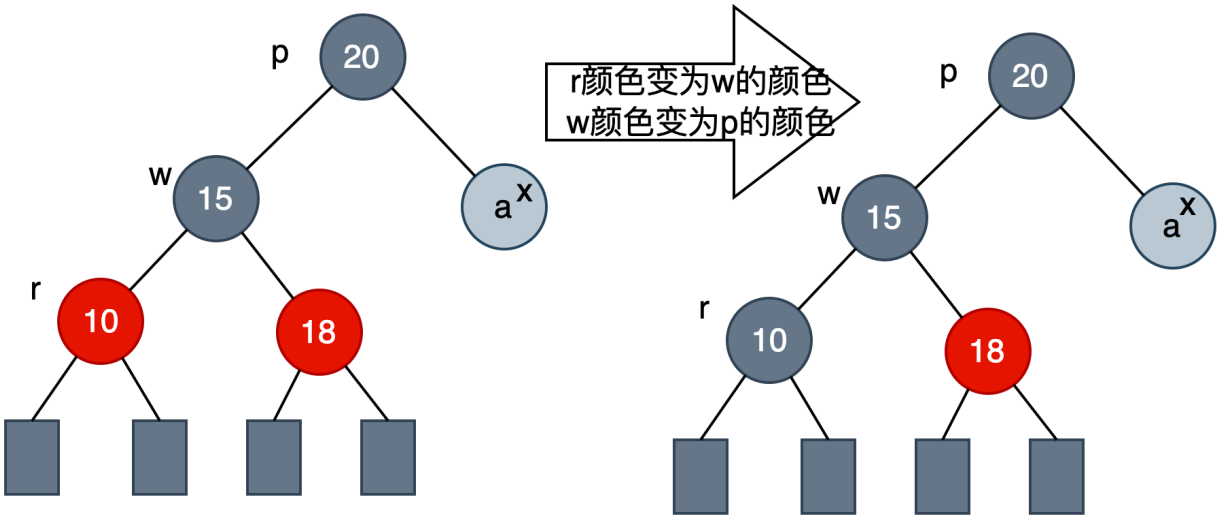
那么p，w和r将出现四种可能性（LL、LR、RR、RL）

- LL（w是p的左孩子，r是w的左孩子，或者w的两个孩子都是红色）

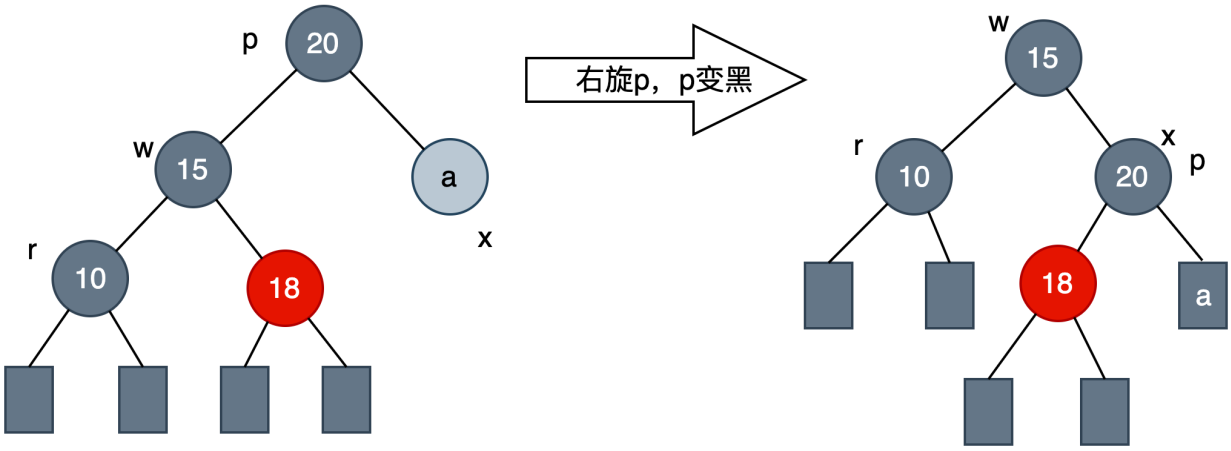


删除节点25，用节点25的NULL节点替换节点25，产生一个双黑节点x，双黑节点x的兄弟节点w为15，节点w是其父节点20(p)的左孩子，其左孩子10(r)也是红色，就为LL情况。

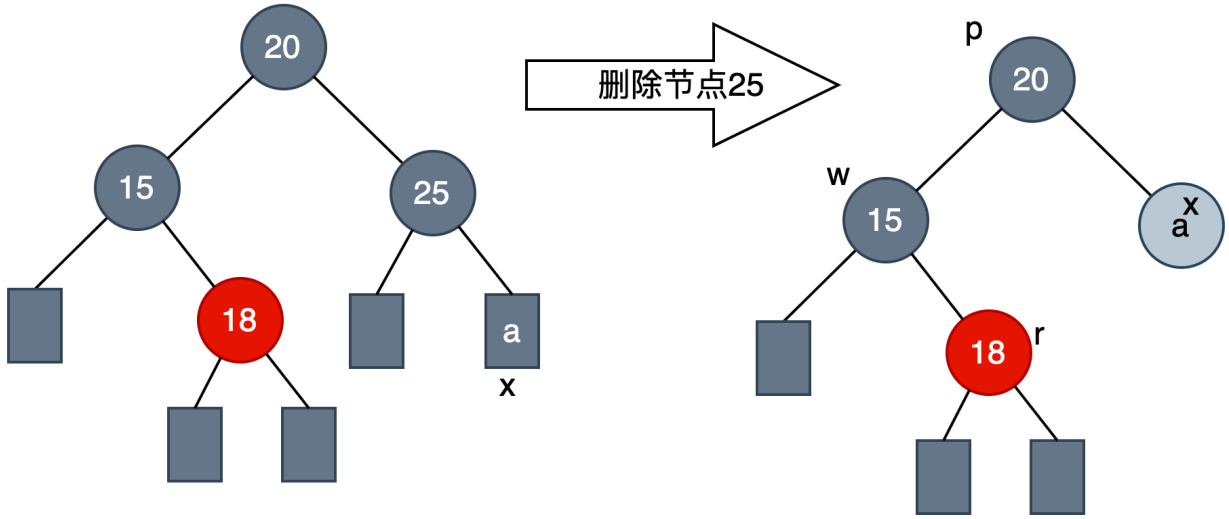
r的颜色设置为w的颜色，w的颜色设置为父节点p的颜色，如下图：



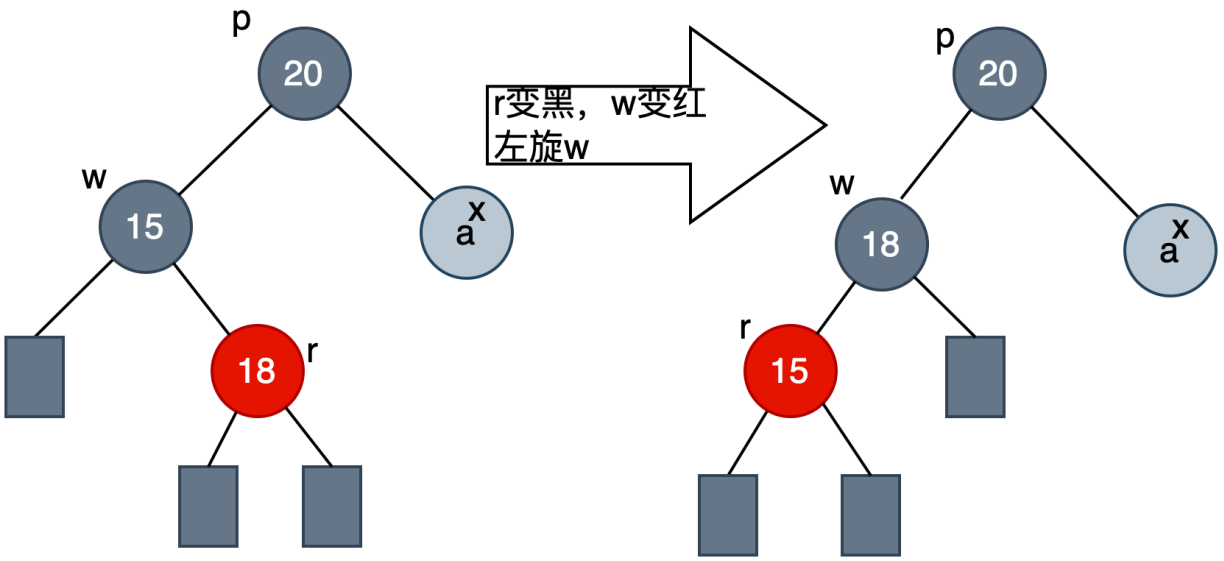
右旋p节点后，将p的颜色设置为黑色，这样a的双黑节点变为单黑节点：



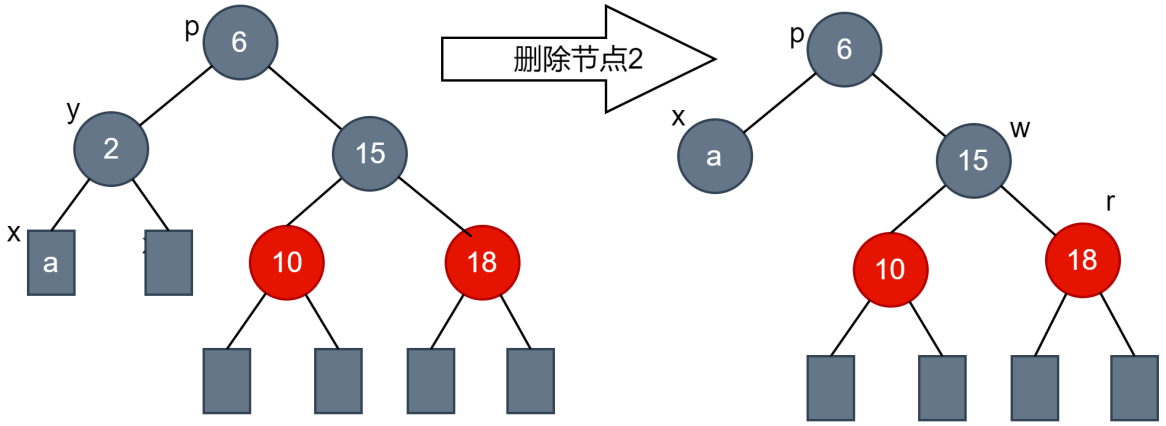
- LR（w是p的左孩子，r是w的右孩子）



将r的颜色设置为黑色，将w设置为红色，对w进行右旋，右旋后，重新设置w

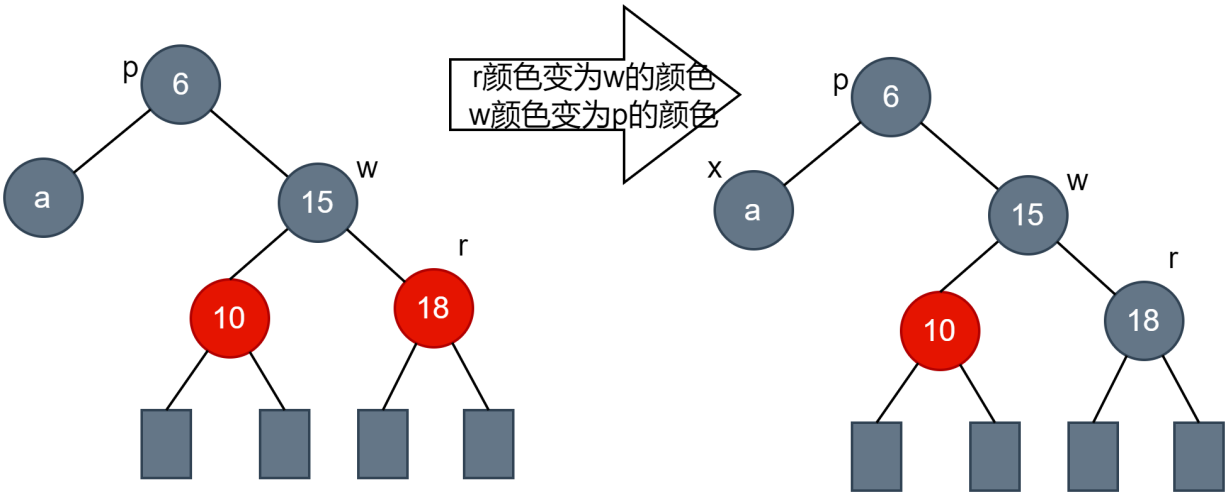


- RR（w是p的右孩子，r是w的右孩子，右孩子为红色）

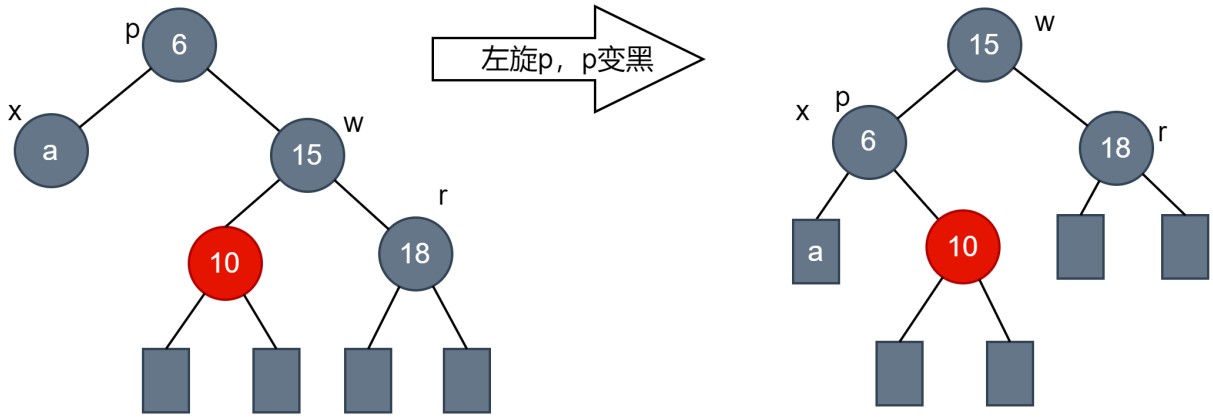


删除节点2，用节点2的NULL节点替换节点2，产生一个双黑节点x，双黑节点x的兄弟节点w为15，节点w是其父节点6（p）的右孩子，其右孩子18（r）正好是红色节点，即为RR。

节点r的颜色变为w的颜色，w的颜色变为p的颜色。



左旋p，p的颜色变为黑色，双黑变单黑。



- RL（w是p的右孩子，r是w的左孩子）

【总结】：

对于兄弟是黑节点，兄弟节点的子孩子至少有一个红色节点的情况：

（1）其中LR想办法转为LL，RL想办法转为RR的形式。

LR --> LL的方法：w的右孩子变为黑色，w变为红色，以w为节点进行左旋。

RL --> RR的方法：w的左孩子变为黑色，w变为红色，以w为节点进行右旋。

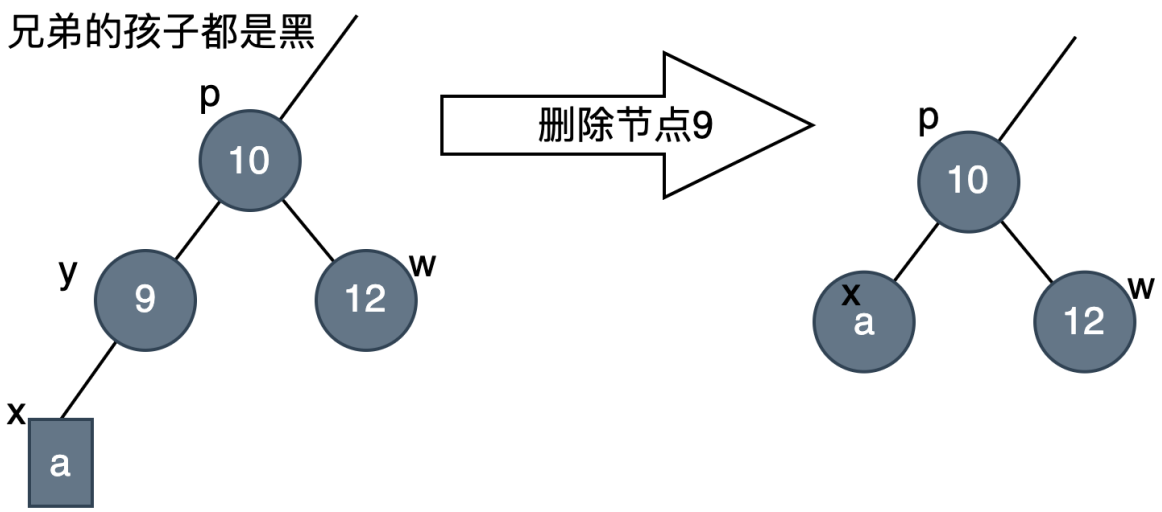
（2）LL的调整方法，r节点(w的左孩子)变为黑色，w节点变为父节点颜色，父节点变为黑色，右旋父节点

（3）RR的调整方法，r节点(w的右孩子)变为黑色，w节点变为父节点颜色，父节点变为黑色，左旋父节点

b. x的兄弟节点w是黑色，且两个孩子节点都是黑色

对于这种情况需要递归地进行处理，如果删除结点后得到的双黑结点的父结点此时为黑色，则结点 u 变单黑，且结点 u 的父结点 p 变双黑，然后对结点 u 的父结点 p 继续进行处理，直到当前处理的双黑结点的父结点为红色结点，此时将双黑结点的父结点设置为黑色，双黑结点变为单黑结点（红色 + 双黑 = 单黑）。

- 兄弟的孩子都是黑色



假设以 10 为根结点的子树为整棵树的左子树，删除结点 9 ，产生双黑结点 a 且其兄弟结点 12（w） 为黑色，兄弟结点的左右孩子均为黑色。

此时双黑结点的兄弟结点 12 变为红色结点，然后将 x 的父结点 10 变为双黑结点，一直向上判断。

这个情况的处理思想：是将“x中多余的一个黑色属性上移(往根方向移动)”。 x是“黑+黑”节点，我们将x由“黑+黑”节点 变成 “黑”节点，多余的一个“黑”属性移到x的父节点中，即x的父节点多出了一个黑属性(若x的父节点原先是“黑”，则此时变成了“黑+黑”；若x的父节点原先时“红”，则此时变成了“红+黑”)。 此时，需要注意的是：所有经过x的分支中黑节点个数没变化；但是，所有经过x的兄弟节点的分支中黑色节点的个数增加了1(因为x的父节点多了一个黑色属性)！为了解决这个问题，我们需要将“所有经过x的兄弟节点的分支中黑色节点的个数减1”即可，那么就可以通过“将x的兄弟节点由黑色变成红色”来实现。

经过上面的步骤(将x的兄弟节点设为红色)，多余的一个颜色属性(黑色)已经跑到x的父节点中。我们需要将x的父节点设为“新的x节点”进行处理。若“新的x节点”是“黑+红”，直接将“新的x节点”设为黑色，即可完全解决该问题；若“新的x节点”是“黑+黑”，则需要对“新的x节点”进行进一步处理。

c. x的兄弟节点w是红色节点

当前 x 的兄弟结点 w 是红色结点时，通过旋转操作将 x 当前的兄弟结点向上移动，并对 x 的父结点和其旋转前的兄弟结点重新着色，接着继续对结点 x 旋转后的兄弟结点 w 进行判断，确定相应的平衡操作。旋转操作将 x 的兄弟结点情况又会转换为前面的（a）和（b）的情况。

> （1）将w设置为黑色
>
> （2）将p设置为红色

(3) 对父进行旋转

(4) 重新设置w