16. 平衡二叉查找树--红黑树

1. 概念与理解

概念：
红黑树属于平衡二叉查找树一种，红黑树的英文是"Red-Black Tree"，简称 R-B Tree。它是一种不严格的平衡二叉查找树

平衡二叉查找树：
二叉树中任意一个节点的左右子树的高度相差不能大于 1。完全二叉树、满二叉树其实都是平衡二叉树，但是非完全二叉树也有可能是平衡二叉树，AVL树严格符合平衡二叉查找树的定义

平衡二叉查找树由来：
解决普通二叉查找树在频繁的插入、删除等动态更新的情况下，出现时间复杂度退化的问题

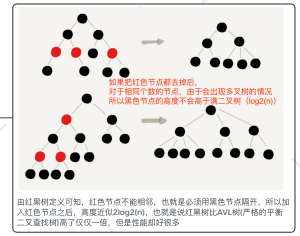
平衡的理解：
平衡二叉查找树中"平衡"的意思，其实就是让整棵树左右看起来比较"对称"、比较"平衡"，不要出现左子树很高、右子树很矮的情况，这样就能让整棵树的高度相对来说低一些，相应的插入、删除、查找等操作的效率高一些

红黑树需满足的要求：
*根节点是黑色的；
*每个叶子节点都是黑色的空节点（NIL），也就是说，叶子节点不存储数据；
*任何相邻的节点都不能同时为红色，也就是说，红色节点是被黑色节点隔开的；
*每个节点，从该节点到达其可达叶子节点的所有路径，都包含相同数目的黑色节点

2.红黑树近似平衡的理解

近似平衡的理解：
平衡二叉查找树的初衷，是为了解决二叉查找树因为动态更新导致的性能退化问题。所以，"平衡"的意思可以等价为性能不退化。"近似平衡"就等价为性能不会退化的太严重

性能与高度的联系
二叉查找树很多操作的性能都跟树的高度成正比。一棵极其平衡的二叉树（满二叉树或完全二叉树）的高度大约是 log2(n)。



由红黑树定义可知，红色节点不能相邻，也就是必须用黑色节点隔开，所以加入红色节点之后，高度近似2log2(n)，也就是说红黑树比AVL树严格的平衡二叉查找树高了仅仅一倍，但是性能却好很多

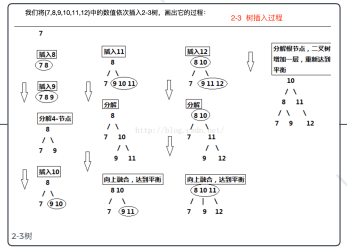
3.平衡调整

平衡调整背后是2-3树的调整可以参考如下链接
https://blog.csdn.net/disiwei1012/article/details/78900594

练习

1. 为什么工程中都喜欢用红黑树，而不是其他平衡二叉查找树呢？
2.红黑树与的2-3树联系？

2.如图：

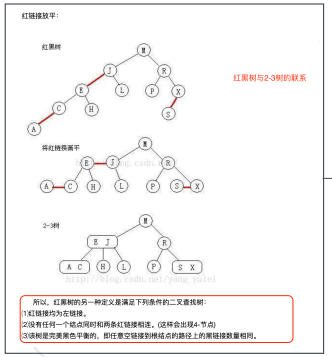


1.思路：

Treap、Splay Tree也是平衡二叉查找树，绝大部分情况下，它们操作的效率都很高，但是也无法避免极端情况下时间复杂度的退化，尽管这种情况出现的概率不大，但是对于单次操作时间非常敏感的场景来说，它们并不适用。

AVL 树是一种高度平衡的二叉树，所以查找的效率非常高，但是，有利就有弊，AVL 树为了维持这种高度的平衡，就要付出更多的代价，每次插入、删除都要做调整，就比较复杂、耗时。所以，对于有频繁的插入、删除操作的数据集合，使用 AVL 树的代价就有点高了。

红黑树只是做到了近似平衡，并不是严格的平衡，所以在维护平衡的成本上，要比 AVL 树要低。所以，红黑树的插入、删除、查找各种操作性能都比较稳定，对于工程应用来说，要面对各种异常情况，为了支撑这种工业级的应用，我们更倾向于这种性能稳定的平衡二叉查找树



红黑树与2-3树的联系