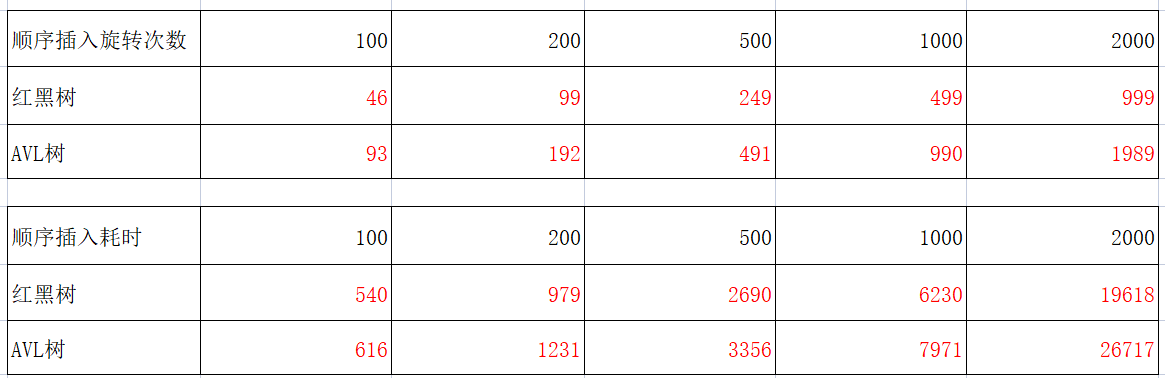
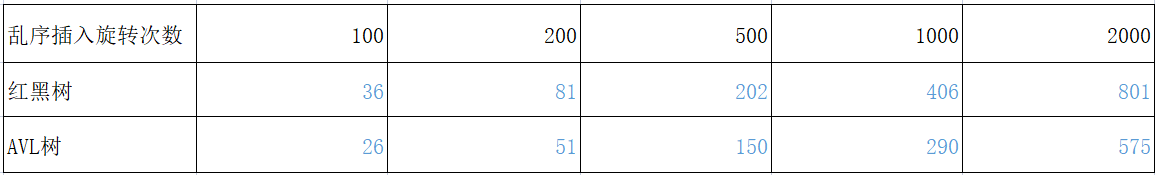
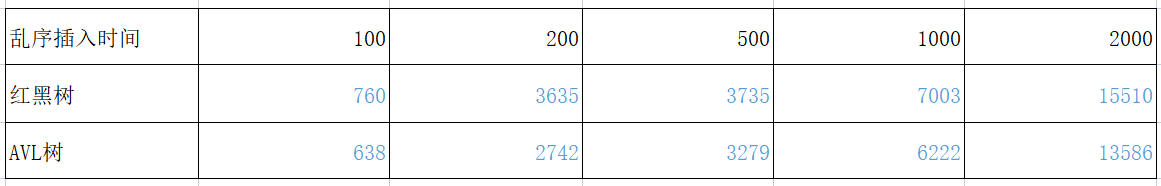
1. **对于插入的耗时以及旋转次数对比：**

**顺序插入：**



**乱序插入：**





从上面的顺序插入以及乱序插入的对比可以看出，对于顺序插入，红黑树的插入时间以及旋转次数均明显优于AVL树，但是乱序插入的时候红黑树的插入消耗的时间高于AVL树，并且旋转次数也明显多于红黑树，这说明乱序插入的时候并没有满足“红黑树对于“平衡”的要求没有 AVL 严格，所以红黑树的插入为了达到“平衡”的旋转次数要更少一些，同时红黑树的高度也会因为不严格的“平衡”而更高，因此查找操作的性能要更低一些”的理论情况。

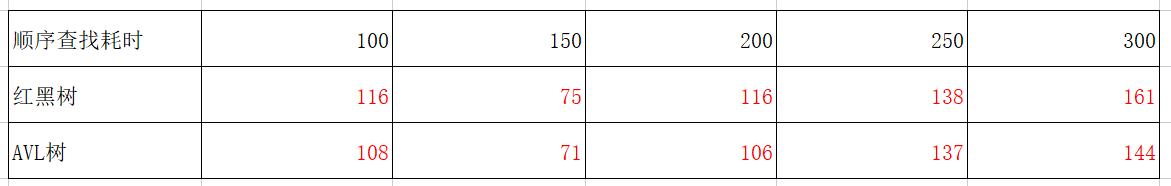
**分析可能的原因：**在本次试验中，我的乱序插入的测试集是通过MATLAB直接生成的在一定范围内的随机数，故用于测试的数据存在一定的随机性；同时，实验次数比较少也可能会使得实际结果与理论数值之间产生较大的偏差。同时，根据个人实践发现，通过top-down

与bottom-up方式生成的结果也存在一定的差异(两种方式的插入都是现在了RBTree.cpp中)，这说明插入顺序的不同也可能对结果差生一定的影响。

1. **对于查询的耗时性能的分析与对比：**

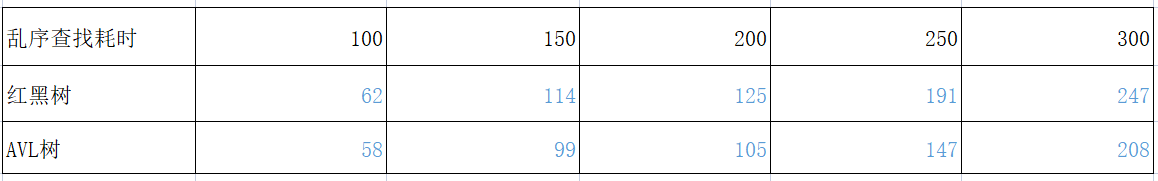
(对于查找的性能分析都是基于插入数据量为1-500来进行的)

**顺序查找：**



从上图可以看出，顺序查找时，avl树的性能在大多数情况下都是优于红黑树的，这是基本符合理论情况的。但是从上图还可以看出，在某些组别下面，avl树与红黑树的查询时间是基本相同的，分析可能的原因：可能是由于实验时的偶然误差(及试验次数多少)引起的一定偏差，但是从总体趋势而言，是符合理论情况的。

**乱序查找：**



从上图可以知道，红黑树与avl的时间性能是avl树的明显优于红黑树的，这是符合“红黑树的高度也会因为不严格的“平衡”而更高，因此查找操作的性能要更低一些”的理论情况。

对AVL树以及红黑树两种数据结构的差异的理解：

1. 对于两者的调节平衡的操作，虽然都分为LL LR RR RL四种情况，但是由于红黑树并没有像AVL树那样要求严格的平衡因子，因而其其平衡性相对于AVL树更差一些。
2. 从时间性能以及空间性能等方面来看，AVL树由于其对平衡的标准更高，因而其高度更加均衡，因而查找时的平均性能较好；但是正是由于其平衡要求过高，其在进行插入后进行调整时需要的时间代价以及空间代价相比于红黑树而言更高；而红黑树虽然对于平衡的要求较低，但是其通过限制节点颜色等方式也能使得整棵树较为平衡，并且由于其平衡开销较小于AVL树，因而其插入性能相较于AVL树更好，并且这一特性也使得其空间开销略小于AVL树。
3. 综合上述分析可以知道，AVL树更适合对已经处理好的数据进行有条件的检索，而相比于AVL树，红黑树更加是用于数据量更大、对数据更新要求更高的情景，如数据库等情景。