作业三：AVL Tree

李沁霞
统计学 3210300363

2022 年 10 月 28 日

1 项目简介

AVL Tree 为 BST 的平衡树，平衡因素有左右子树之间的差距影响。若差距不超过 1 是平衡树，反之依然。可以使用转换解决树的平衡性。转换程序设计可以在 AvlTree.h 的 heading 文件寻找。

2 设计思路

设计 vector 函数的列表，输入元素为 [k1,k2] 之间的树，之后测试程序的运行时间。根据树的高度有三种不同的时间复杂形势。我们将三种时间复杂性测试程序的运行时间。

3 理论分析

由于树的不同高度情况，导致程序有不同的时间复杂性。以下是根据 AVL Tree 的高度而产生的间复杂性：

1. 最佳案例：当 $k <= log(N)$ 时，程序的时间复杂性为 $O(log(N))$.
2. 最差案例：当 $k = N$ 时，程序的时间复杂性为 $O(N)$.
3. 平均案例：当 $k = log_n(N)$ 时，程序的时间复杂性为 $O(k + log(N))$.

4 数据结果分析

- 测试结果

```
当 k <= log(N):
-1 829 -200 500
-200 -1 500 829
当 k = N :
The run time of 2000    in test 1 cost: 0.002016 s.
The run time of 20000   in test 1 cost: 7e-06 s.
The run time of 200000  in test 1 cost: 8e-06 s.
The run time of 2000000 in test 1 cost: 3.1e-05 s.
当 k = logn(N) :
The run time of 65536   in test 2 cost: 0.000437 s.
The run time of 262144  in test 2 cost: 1e-05 s.
The run time of 1048576 in test 2 cost: 1.3-05 s.
The run time of 4194304 in test 2 cost: 1.4e-05 s.
```

$O(log(N))$		$O(N)$			$O(k+log(N))$		
insert	output	k	n	time(s)	$k1$	$k2$	time(s)
-1	-200	2000	2000	0.002016	2^{16}	$log_2(2^{16})$	0.000437
829	-1	20000	20000	7e-06	2^{18}	$log_2(2^{18})$	1e-05
-200	500	200000	200000	8e-06	2^{20}	$log_2(2^{20})$	1.3-05
500	829	2000000	2000000	3.1e-05	2^{22}	$log_2(2^{22})$	1.4e-05

从表上可知，$O(log(N))$ 的运行时间是最优的，很快就可以输出列表中的元素。而 $O(k+log(N))$ 的运行时间也好，虽然输入的元素比 O(N) 比较大，但是运行结果更快。这就说明 $O(N)$ 的时间复杂度是最差的。

5 结论

通过头文件理解 AvlTree 的程序设计思路，另外设计了 main.cpp 文件测试 AvlTree 的时间复杂性运算。由头文件和 main.cpp 可知树的高度影响到程序的运行时间。