hw4 Splay Tree Vs AVL

1. 测试集

- 选择[0,n)中n个整数，乱序插入。
- 测试集选择k个连续的数，每个数重复m/k次，共搜索m次，顺序打乱

```
int key = rand() % n;//搜索的起始值
vector<int> *test =  new vector<int>;

for (int i = 0; i < k; ++i) {
    for(int j = 0; j < m/k ; ++j) {
        test->push_back((i + key) % n);
    }
}
random_shuffle(test->begin(), test->end());
```

参数选择 n=1000，m=100000时，取不同k/n值，输出结果如下：

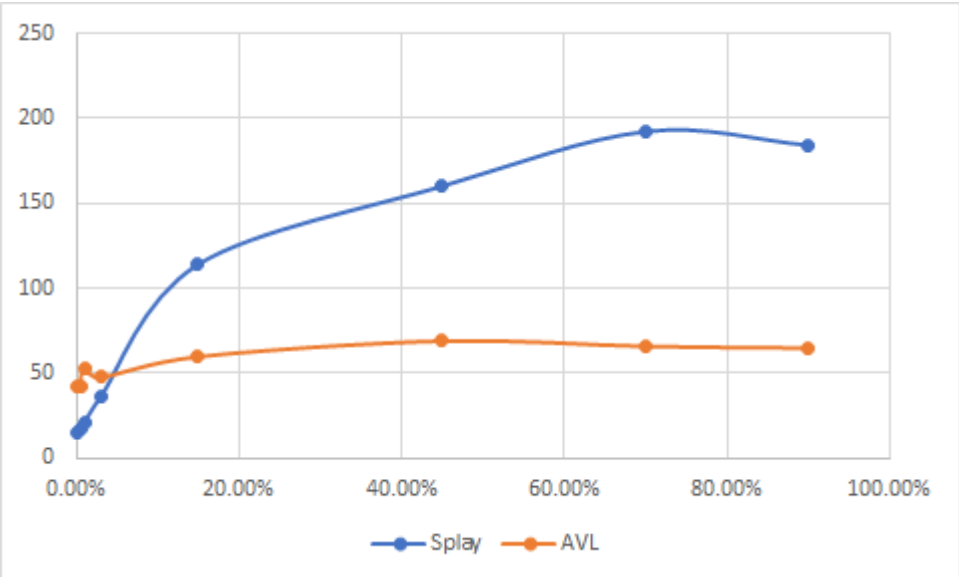
```
Test:: k/n = 0.1%
Search time of Splay Tree : 15
Search time of AVL Tree : 42

Test:: k/n = 0.5%
Search time of Splay Tree : 17
Search time of AVL Tree : 42

Test:: k/n = 1%
Search time of Splay Tree : 21
Search time of AVL Tree : 53

Test:: k/n = 3%
Search time of Splay Tree : 36
Search time of AVL Tree : 48

Test:: k/n = 15%
Search time of Splay Tree : 114
Search time of AVL Tree : 60

Test:: k/n = 45%
Search time of Splay Tree : 160
Search time of AVL Tree : 69

Test:: k/n = 70%
Search time of Splay Tree : 192
Search time of AVL Tree : 66

```

```
Test:: k/n = 90%
Search time of Splay Tree : 184
Search time of AVL Tree : 65
```

2. 实验数据记录

- 当n = 1000, m = 1000000, n/m = 0.1%，记录用时

k/n	Splay	AVL
0.1%	15	42
0.5%	17	42
1%	21	53
3%	36	48
15%	114	60
45%	160	69
70%	192	66
90%	184	65



3. 结果分析

- AVL基本不受测试集内容变化所影响。
- Splay Tree受测试集的局部性影响较大，当搜索范围广度增加时，性能明显下降
- 在测试集k/n较小时，即搜索得较集中时，Splay Tree体现出较大优势（耗时为AVL三分之一）
- 应用场景：实际使用搜索引擎时，同一段时间内，人们搜索时常常用类似的语言来描述同一样事物，每次搜索的关键词相似度较高，搜索对象集中，Splay Tree会有较大优势。

Reference

Splay Tree：https://github.com/BigWheel92/Splay-Tree

AVL：HW1给的样例