

算法基础上机实验二

PB17000285

雷洋

一．实验内容

- 1.实现求最优二叉搜索树算法。关键字 n 数目为 5, 9, 13, 17, 21。输入在相应的 txt 文件中, 输出要求打印二叉树, 并给出期望搜索代价。统计算法运行所需时间, 画出时间曲线, 并进行性能分析。
- 2.实现求最长公共子序列的算法。序列 X 的长为 m , 序列 Y 的长为 n , 序列 X 和 Y 的元素从 26 个大写字母中随机生成, 给出算法运行所需的时间, 画出时间曲线, 进行性能分析。

二．实验要求

1. 实验文件严格按照要求创建。
2. 最优二叉搜索树: 从输入文件读入数据; 最长公共子序列随机生成数据。
3. 每个实验建立子文件夹, result.txt 输出结果, time.txt 对应时间输出。

三．实验设备及环境

实验设备和环境为 windows10 个人计算机操作系统下的
devC++ IDE, TDM-GCC 4.9.2 64-bit。

四．实验方法

最优二叉搜索树:

1. 通过文件读入函数 set_data 来输入数据, 存放在全局数组

中。

2. 调用 `optimal` 函数对不同规模的输入进行计算。通过递归形式的 `print_tree` 函数，将 `root` 数组包含的信息转化成前序遍历输入到 `result.txt` 文件。
3. 通过 `windows` 库里的函数获取 `cpu` 频率，以及开始和结束的周期数差，来计算运行时间（单位秒），文件写入 `time.txt` 中。
4. 重复统计 50 次实验的各个规模时间，以其平均数来作为最后分析的数据，绘图观察分析得出结论。

最长公共子序列：

1. 通过 `createdata` 函数，随机生成 0-25 的整数，转化为 a-z 字母，输出到 `inputA`, `inputB` 中。
2. `Setdata` 函数将文件中的数据导入到数组中。
3. 调用 `lcs` 和 `print` 函数，将对应的数据计算并输出。
4. 通过 `windows` 库里的函数获取 `cpu` 频率，以及开始和结束的周期数差，来计算运行时间（单位秒），文件写入 `time.txt` 中。
5. 重复统计 50 次实验的各个规模时间，以其平均数来作为最后分析的数据，绘图观察分析得出结论。

五．实验步骤

最优二叉搜索树：

1. 创建头文件 `head.h`, 写入必要的宏定义，声明和定义所用到的全

局变量和函数。

2. 创建 main.cpp，进行统计和输出。

最长公共子序列：

1 . 创建头文件 head.h,写入必要的宏定义，声明和定义所用到全局变量和函数。

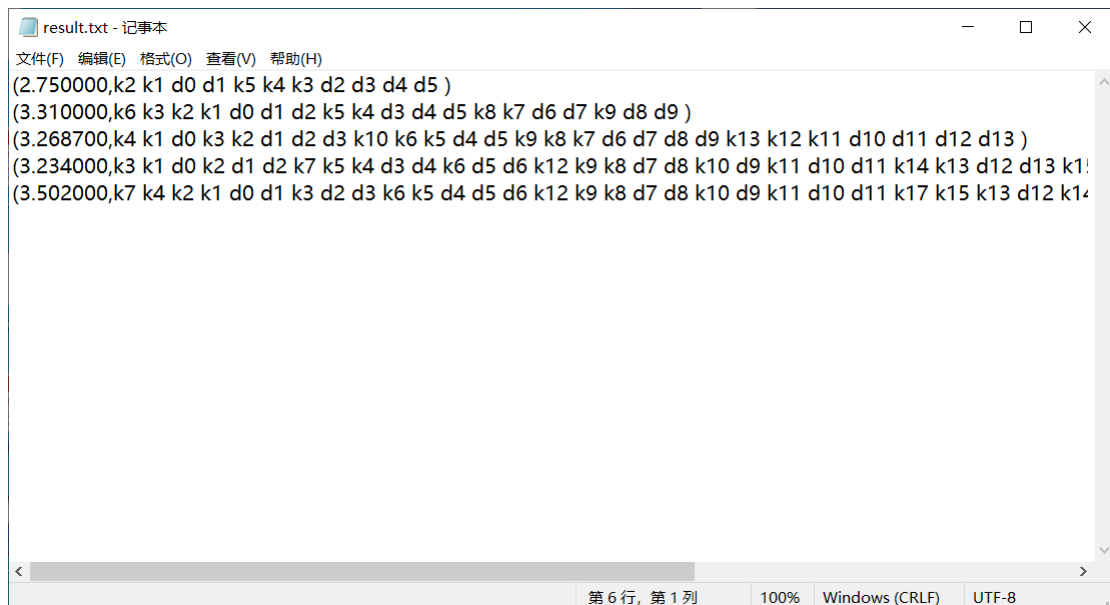
2 . 创建 createdata.cpp，随机生成字符串，导入到 input 文件。

3 . 创建 main.cpp，进行统计和输出。

六．实验结果与分析

1. 本次实验结果截图：

最优二叉搜索树输出：



```
result.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
(2.750000,k2 k1 d0 d1 k5 k4 k3 d2 d3 d4 d5 )
(3.310000,k6 k3 k2 k1 d0 d1 d2 k5 k4 d3 d4 d5 k8 k7 d6 d7 k9 d8 d9 )
(3.268700,k4 k1 d0 k3 k2 d1 d2 d3 k10 k6 k5 d4 d5 k9 k8 k7 d6 d7 d8 d9 k13 k12 k11 d10 d11 d12 d13 )
(3.234000,k3 k1 d0 k2 d1 d2 k7 k5 k4 d3 d4 k6 d5 d6 k12 k9 k8 d7 d8 k10 d9 k11 d10 d11 k14 k13 d12 d13 k15 )
(3.502000,k7 k4 k2 k1 d0 d1 k3 d2 d3 k6 k5 d4 d5 d6 k12 k9 k8 d7 d8 k10 d9 k11 d10 d11 k17 k15 k13 d12 k14 )
第 6 行, 第 1 列 100% Windows (CRLF) UTF-8
```

最优二叉搜索树时间统计：

```
time.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
0.000032 0.000009 0.000031 0.000043 0.000056
0.000003 0.000009 0.000019 0.000050 0.000055
0.000003 0.000009 0.000018 0.000033 0.000055
0.000003 0.000010 0.000019 0.000034 0.000056
0.000003 0.000010 0.000018 0.000034 0.000054
0.000003 0.000009 0.000019 0.000034 0.000056
0.000003 0.000009 0.000018 0.000034 0.000055
0.000003 0.000010 0.000019 0.000034 0.000056
0.000003 0.000009 0.000018 0.000033 0.000055
0.000003 0.000009 0.000018 0.000034 0.000062
0.000003 0.000010 0.000019 0.000036 0.000056
0.000003 0.000009 0.000018 0.000034 0.000056
0.000003 0.000009 0.000018 0.000034 0.000055
0.000003 0.000010 0.000019 0.000035 0.000055
0.000003 0.000009 0.000019 0.000035 0.000056
0.000003 0.000010 0.000018 0.000034 0.000056
0.000003 0.000009 0.000019 0.000034 0.000075
0.000003 0.000009 0.000018 0.000034 0.000055
第 1 行, 第 1 列 100% Windows (CRLF) UTF-8
```

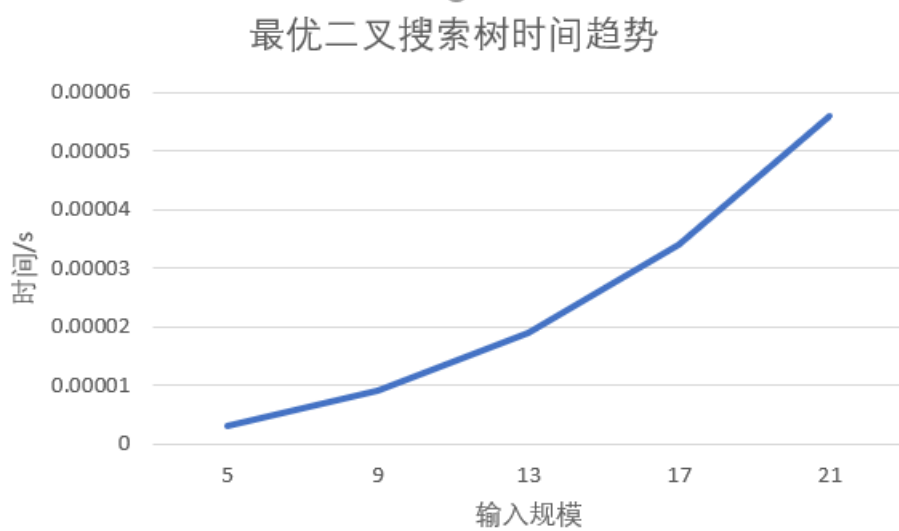
最长公共子序列结果:

```
result.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
(16,10)fx 2
(16,20)fx 2
(16,30)csc 3
(16,40)fx 2
(16,50)vx 2
(16,60)rc 2
(15,26)clvwxx 6
(30,26)gguoxtc 7
(45,26)rdubmoxs 8
(60,26)dvybsmbxhc 10
(75,26)bdybubrxlsm 11
(90,26)dlusmorxthcm 13
第 1 行, 第 1 列 100% Windows (CRLF) UTF-8
```

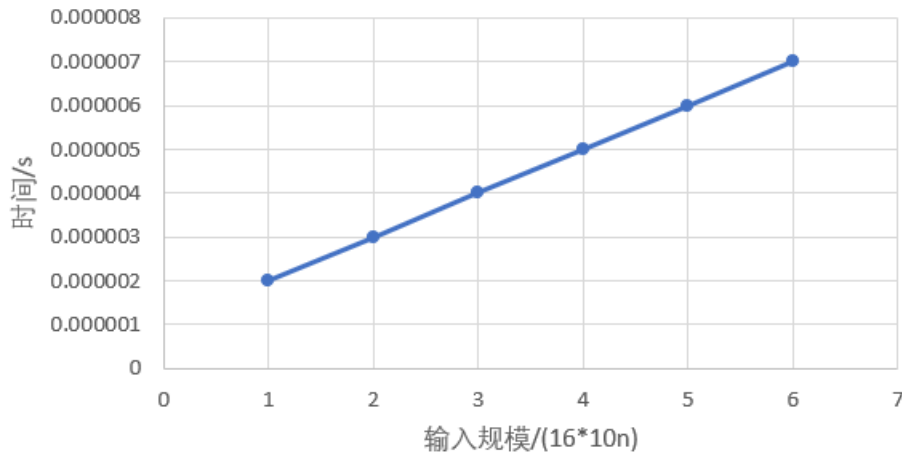
最长公共子序列时间统计:

```
time.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
(16,10) 0.000002
(16,20) 0.000003
(16,30) 0.000004
(16,40) 0.000005
(16,50) 0.000006
(16,60) 0.000007
(15,26) 0.000003
(30,26) 0.000009
(45,26) 0.000012
(60,26) 0.000015
(75,26) 0.000017
(90,26) 0.000018
(16,10) 0.000002
(16,20) 0.000003
(16,30) 0.000004
(16,40) 0.000005
(16,50) 0.000006
(16,60) 0.000007
第 12 行, 第 17 列 100% Windows (CRLF) UTF-8
```

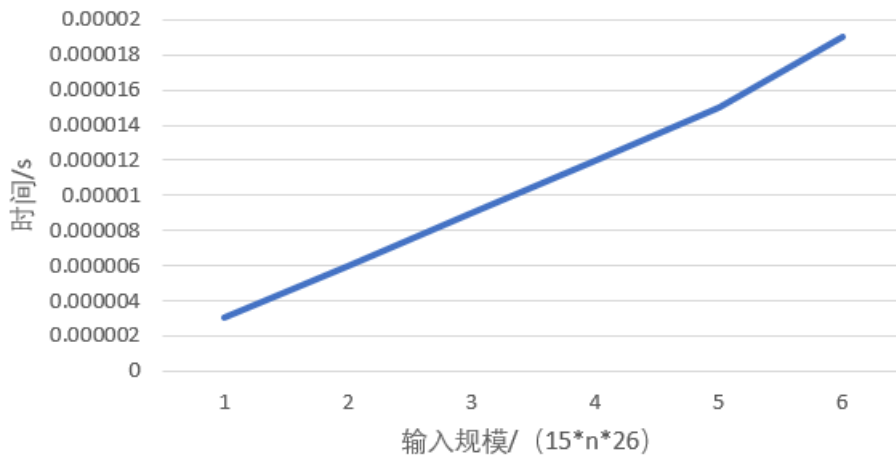
2.结果分析



最长公共子序列 (16, 10n) 时间趋势



最长公共子序列 (15n, 26) 时间趋势



通过观察最优二叉搜索树时间趋势图可以发现，时间复杂度在 $\Theta(n^2)$ ，与书上给出的 $\Theta(n^3)$ 不同，推测原因可能是：最优二叉搜索树原算法中有一层循环中大部分为无效操作，可以简化，使得该无效循环操作得以避免，使其复杂度降低。具体改动为：将 `for r = i to j` 修改为 `for r = root(i,i+l) to root(i+1,i+l-1)`。

通过观察最长公共子序列时间趋势图可以发现，时间复杂度在 $\Theta(mn)$ ，对应 (16, 10n) 和 (15n,26) 规模的输入，时间呈线性增加，比较符合预期。