

# 拼音输入法编程作业

## 软件最终效果

```
chenyu@chenyu-CP65S ~/Typewriting/build (master*)
$ ./main input.txt output.txt answer.txt
初始化中...
耗时(初始化):5.56327 s
正在尝试搜寻汉字...
耗时(搜寻汉字):58.6794 s
字正确率: 83.5447 字总数: 7177
语句正确率: 39.4805 语句总数: 770
请输入要查询的拼音(exit 表示退出)
>
```

```
chenyu@chenyu-CP65S ~/Typewriting/build (master*)
$ ./main
初始化中...
耗时(初始化):5.61566 s
请输入要查询的拼音(exit 表示退出)
> qing hua da xue
清华大学
> qing hua da xue ji suan ji xi
清华大学计算机系
> qing hua da xue shi shi jie shang zui hao de xue xiao
清华大学是世界上最的学校
> pu ti ben wu shu
菩提本无树
> ming jing yi fei tai
明镜亦非台
> ben shi wu yi wu
本市无一梧
> he chu re chen ai
何处惹尘埃
> zui hao de yu yan
最好的语言
> zi jing gong yu
紫荆公寓
> xue xi
学习
> wo ai xue xi
我爱学习
> xue xi shi wo kuai le
学习是我快乐
>
```

测试集采用网络学堂上采集的测试集，最终字的正确率为83.5%,句子的正确率为39.5%。

算法运行依赖的文件大小约200M，初始化大约5秒钟，查询一条语句大约0.5秒钟;批量处理的时候使用了多线程进行优化。

## 算法说明

最终算法采用三元概率模型。

首先，整理新闻，分别统计出单个字，两个字和三个字出现的次数;根据贝叶斯公式：

$$P(W_i|W_{i-2}W_{i-1}) = \frac{P(W_{i-2}W_{i-1}W_i)}{P(W_{i-2}W_{i-1})}$$

可以通过频率可以计算出已知前两个字，第三个字出现的概率;

最终，一种方案的得分即使：

$$S = P(W_1) * P(W_2|W_1) * \prod_3^n (P(W_i|W_{i-2}W_{i-1}))$$

然后，根据这个公式，选择一种最优的方案;注意到上面的公式是可以使用动态规划来加速的，所以最终实现的过程中使用了动态规划。

同时，对于建出的索引，由于有很多出现次数非常小的组合，所以在实际的算法中，对索引进行了过滤，一方面可以减少索引的大小，另一方面可以加速搜索的过程。

## 其他算法

在实现输入法算法的过程中，也探索了一些其他算法;

首先实验了二元概率算法，其根据是下面两个公式：

$$P(W_i|W_{i-1}) = \frac{P(W_{i-1} W_i)}{P(W_{i-1})}$$

$$S = P(W_1) * \prod_2^n (P(W_i|W_{i-1}))$$

这两个公式和上面三元的公式非常相似，实际算法实现的过程也很像，最终字的正确率大约76%，语句的正确率大约76%。

其次，尝试使用词语进行优化，但无奈没有找到合适的数学模型，最终最高的语句正确率只有33%。

## 存在的问题

- 没有考虑多音字，实际上在计算概率P的时候，应该考虑上字的发音
- 没有考虑词性

## 实验收获

- 了解了拼音输入法的实现原理
- get了一波C++11和多线程的技能