Project4串匹配实验报告

徐煜森 PB16110173

1. **实验要求**

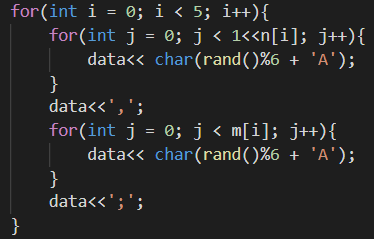
实现字符串匹配算法，文本串T的长度为n，对应的模式串P的长度为m，字符串均是随机生成的字符 (A-F，共6种不同字符)。(n, m)共取五组数据: (2^5，2), (2^8，3), (2^11,4), (2^14,5), (2^17,6)。

其中需要实现的算法有：Rabin-Karp算法；KMP算法；Boyer-Moore-Horspool算法

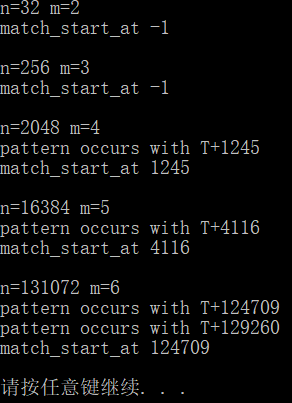
1. **实验环境**
2. Windows10 64位x86，机器内存8G，时钟主频2.59GHz
3. 软件环境：Visual Studio 2017
4. **实验过程**
5. **编译选项**

注意编译选项 –std = c++11

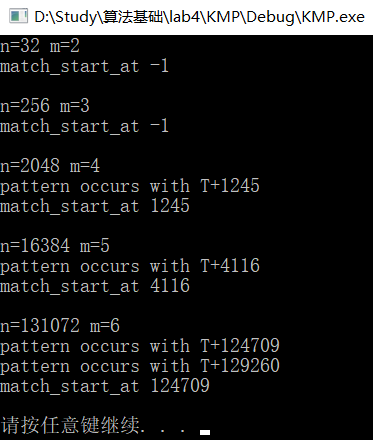
1. **生成数据**



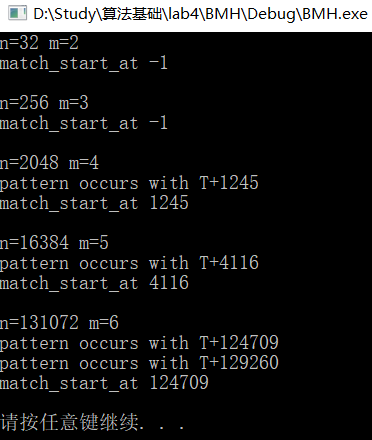
1. **Rabin Karp**
2. 按照书上方法实现算法，注意C++中数组下标从0开始，在之后代码截图部分也将强调这一点。
3. 书上考虑到模式串可能很长，对ts、p进行了mod q计算，而本次实验中模式串最长为6位，故无需mod q，因而也无需考虑伪命中情况，不过为了使算法可拓展、方便阅读，依然考虑了伪命中情况。
4. 以下为运行结果截图



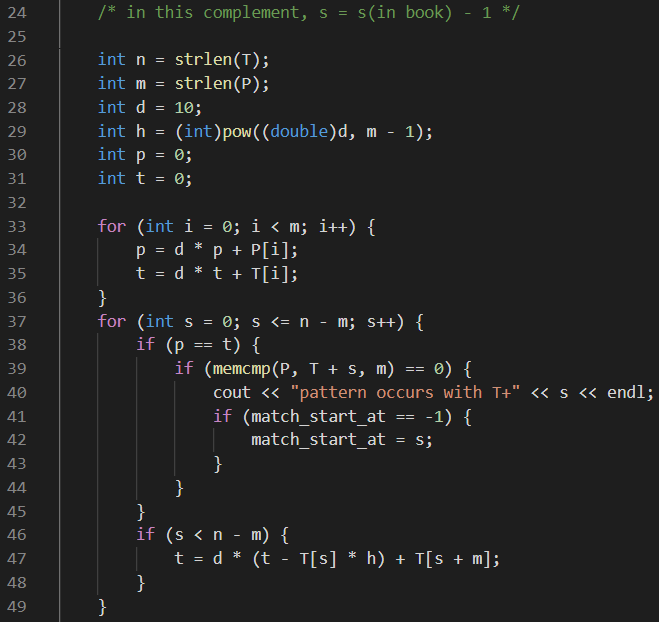
1. **KMP**
2. 按照书上算法实现KMP，同样需要注意数组下标从0开始
3. 以下为运行结果截图



1. **BMH**
2. 按照课件上算法实现BMH，同样需要注意数组下标从0开始会导致偏移量不相同。
3. 以下为运行结果截图

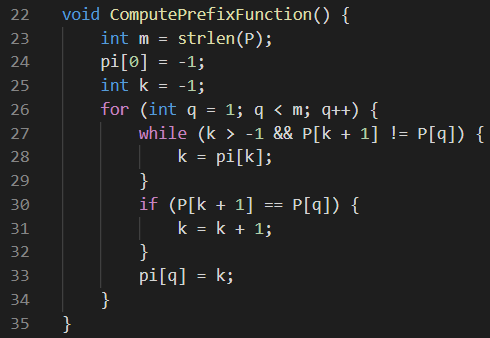


1. **关键代码截图**
2. **Rabin Karp**

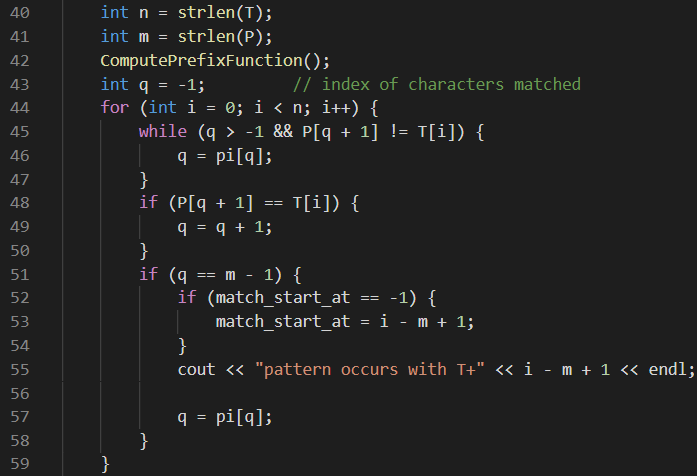


从图中可以看到，计算p和t时没有像书上进行mod q操作，因为实验中模式串最长为6位，可以使用一个int变量储存，但考虑到算法的可拓展性、可阅读性，依然加入伪命中判断。另外值得注意的 是本代码中的s = 书上代码中的s – 1。同时，记录下最早匹配到的字符串的起始下标。

1. **KMP**
2. 以下为计算KMP辅助数组pi的函数，应注意这里数组中存的值表示已经匹配成功的字符的下标，-1则表示没有字符成功匹配。

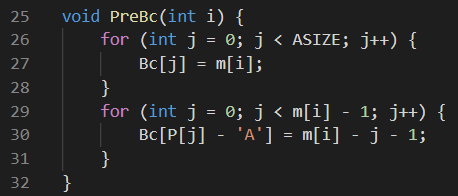


1. 以下为KMP算法实现：

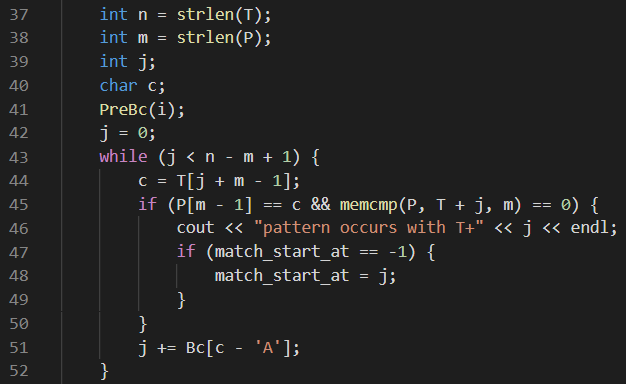


可以发现本代码实现的KMP算法中q的含义变化为已匹配成功的字符的下标，-1则表示没有字符成功匹配。另外应注意下标的换算。

1. **BMH**
2. 以下为BMH辅助函数坏字符Bc数组的计算函数，其中ASIZE为字母表的大小，本实验中为6。另外应注意偏移量的计算。

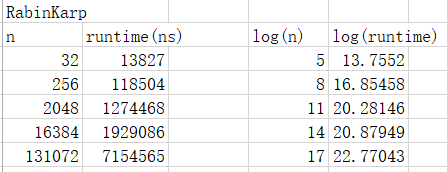


1. 以下为BMH算法

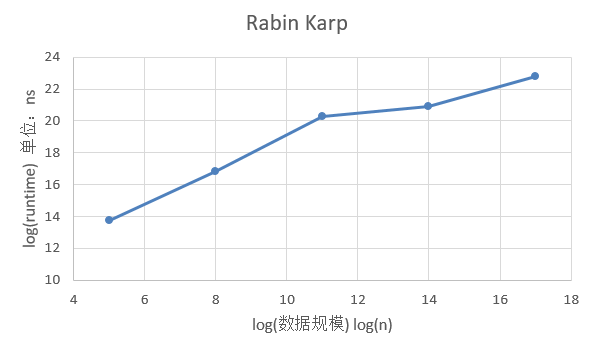


应注意的是下标计算与课件上有些区别。

1. **实验结果及分析**
2. **Rabin Karp**
3. 以下为Rabin Krap算法运行时间原数据，因为数据间差值太大，将对数据以2为底取log画图（KMP、BMH同理）。

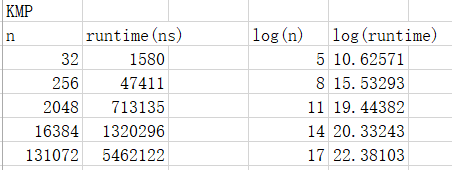


1. 以下为log(n)-log(runtime)图

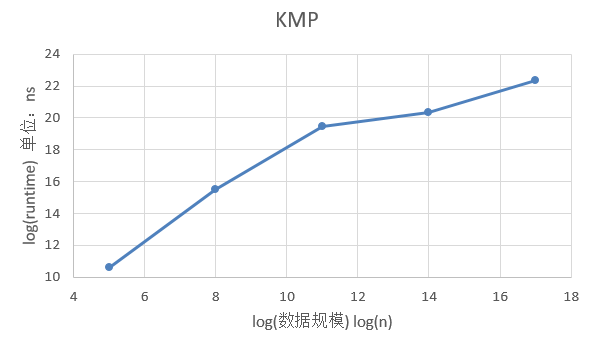


可以观察出，运行时间基本符合期望Ο(n)，因为本次实验中数据量足够大，相较于之前实验更容易得到准确结果。曲线有些波动很有可能是受硬件cache和操作系统调换页的影响，几乎可以忽略不计。

1. **KMP**
2. 以下为KMP运行时间原数据

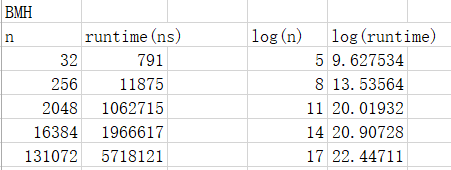


1. 以下为log(n)-log(runtime)图

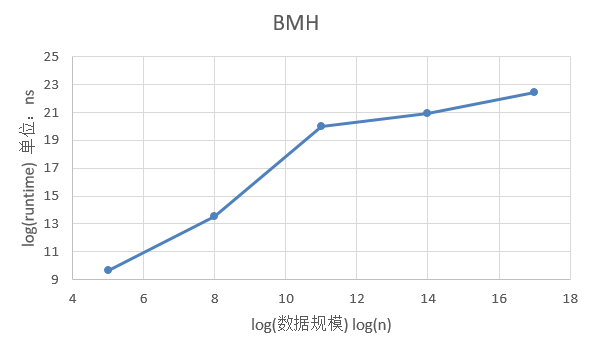


可以看出运行时间基本符合期望Θ(n)，这也是三个算法运行时间中曲线波动最小的，可以认为在本实验中KMP算法相较于RK和BMH（见下面）算法表现更稳定。

1. **BMH**
2. 以下为BMH算法运行时间原数据



1. 以下为log(n)-log(runtime)图

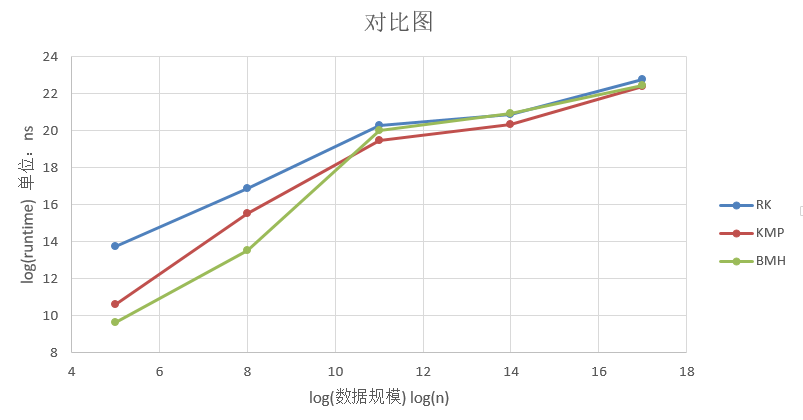


可以看出运行时间基本符合期望Ο(n)，但曲线在中间有较小的向上波动，可能是受到硬件cache策略影响较大，如cache在运行第三组数据时，因其他程序也要占用一定存储空间的原因将BMH中使用到的数组调换出一部分，在之后会引发较多的cache miss。

我针对BMH算法查阅相关资料，发现BM和BMH算法在工业界应用比KMP算法要广泛的多。大家普遍认为BM算法比KMP算法要稳定，而且平均性能更好，尤其是在搜索引擎领域BM算法远好于KMP算法。

1. **对比图**

将三个算法得到的三组数据画在同一张图上进行对比



可以看出三种算法在本次实验中的区分度并不大，渐进时间复杂度也与期望相同，均为Ο(n)。

1. **实验总结**

本次实验中让我对这三个算法有了更进一步的理解，同时也锻炼了我的对下标变换的理解能力。另外也通过查阅资料了解了更多关于字符串匹配算法的知识。