由期中考试最后一题引发的思考

**原题大意：**一只猴子随机地敲击键盘上“A”“B”“C”“D”四个字母，敲击N次，形成一个长度为N的字符串，请问这个字符串中“ABAB”子串个数的期望是多少？（“ABABAB”这样的字符串记2个“ABAB”子串）

笔者认为，对于此类问题，如能找到一种计数算法数出任一字符串中“ABAB”子串的个数，便一定能得出问题要求的期望。**算法与概率问题之间似乎有这样一种“默契”。**

对于这个问题，笔者给出了三种算法（逐四位统计法、归并法、折减法），并进行了编程实现（编译环境Visual Studio 2013 Community，语言C++）。程序中，N设定为1024，要求用户输入猴子敲击字符串的数量num（样本容量），然后调用其中一种算法统计所有这些字符串中“ABAB”子串的数量，求出平均值，与理论的期望值（(N-3)/256 = 3.98828）相比较；并给出计数算法在num个字符串上运行的总时间。对于折减法，额外给出字符串中首个“AB”出现位置的分布情况。程序运行示例如下：

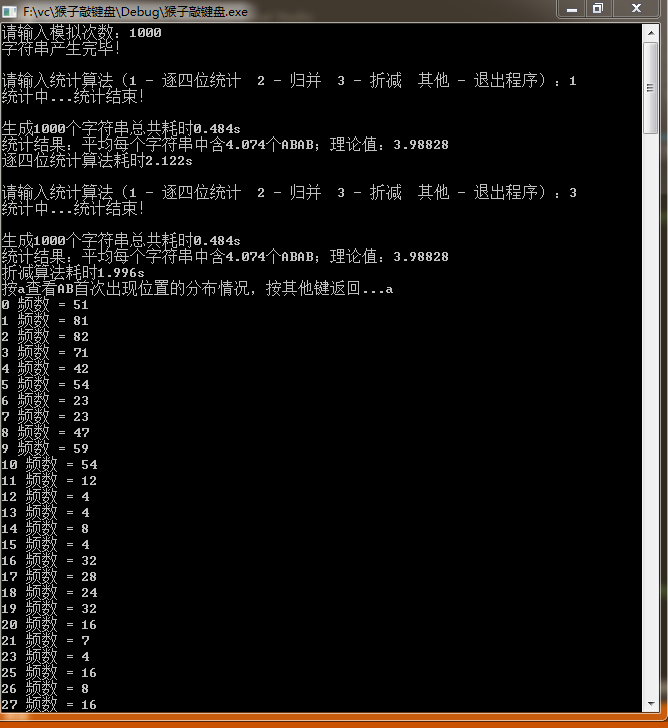


图 1 运行示例第一部分

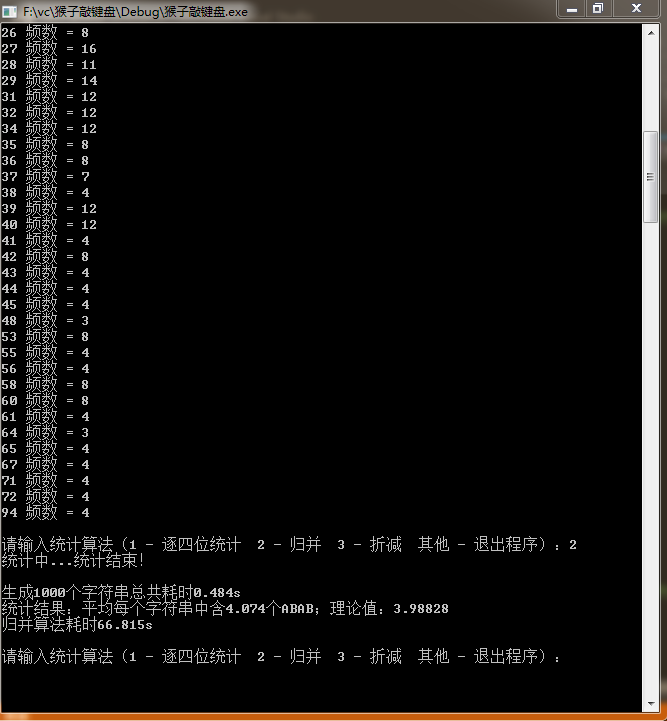


图 2 运行示例第二部分

在进入三种算法的介绍之前，先介绍程序代码的框架及算法之外的部分，必要的说明见代码注释。

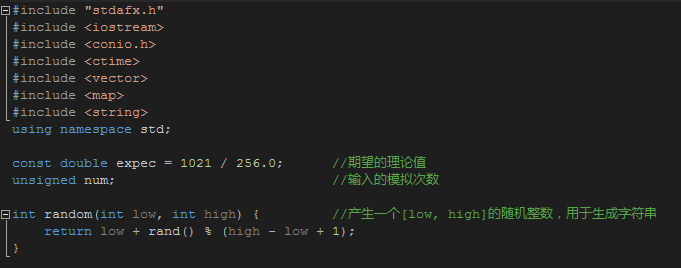


图 3 程序首部

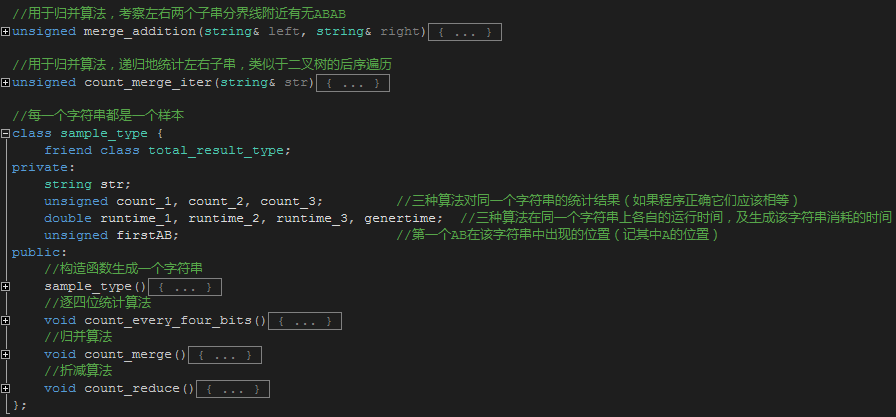


图 4 sample\_type类总览

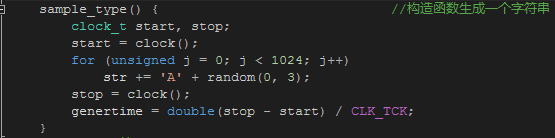


图 5 sample\_type构造函数

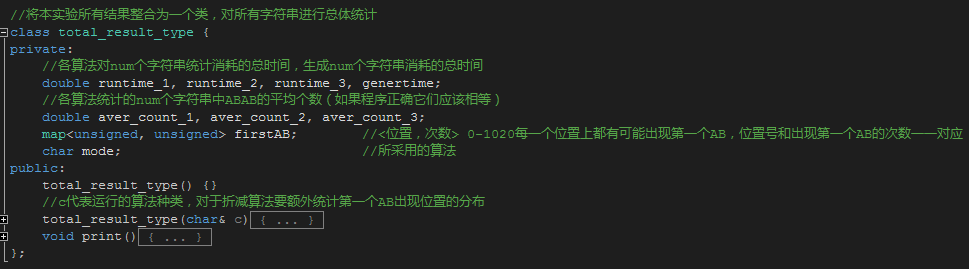


图 6 total\_result\_type类总览

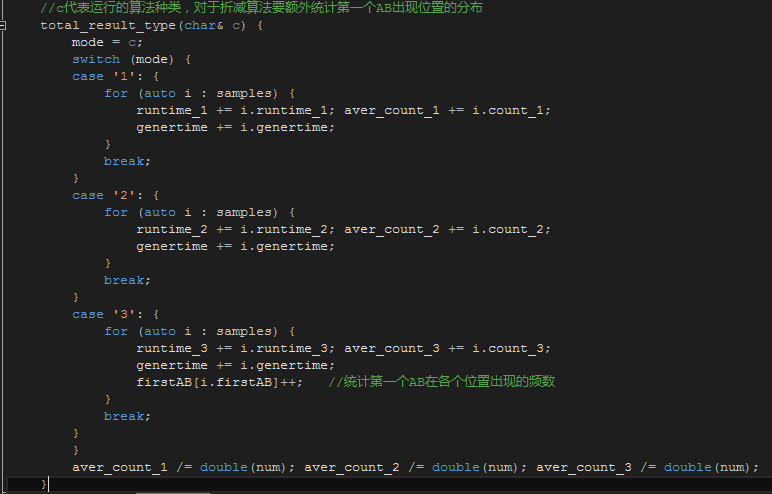


图 7 total\_result\_type构造函数

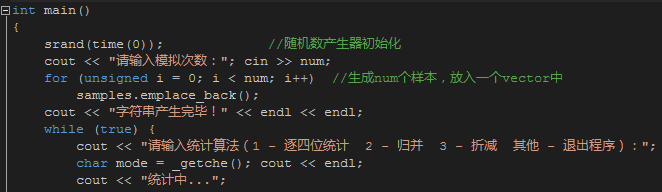


图 8 main函数第一部分

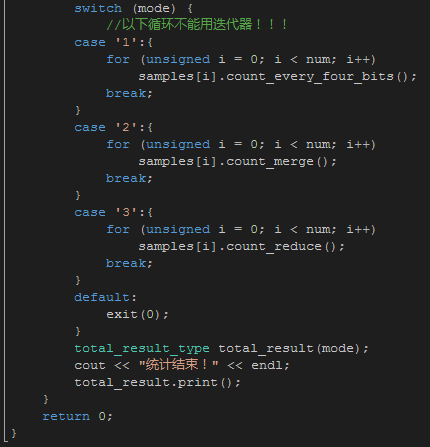


图 9 main函数第二部分

1. **逐四位统计法**

这是最简单的算法。从第一位到第N-3位，每一位和它后面的三位构成一个子串，确认每个子串是否是“ABAB”，如果是就计1。这样，每次读取四个字母。第k位和它后面三位构成“ABAB”的概率是，k从1到(N-3)都是如此。尽管第k ~ (k+3)位是“ABAB”和第(k+1) ~ (k+4)位是“ABAB”这两个事件并不独立（是互斥的），但由于期望的线性性质并不要求各个事件相互独立，因此仍然可以这样计算整个字符串中包含“ABAB”子串数目的期望：。

代码如下：

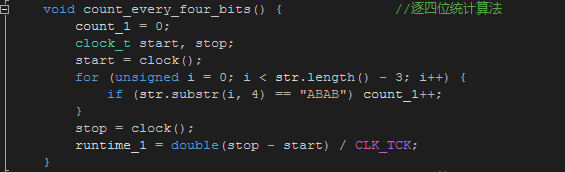


图10 逐四位统计算法

易知其时间复杂度为。

1. **归并法**

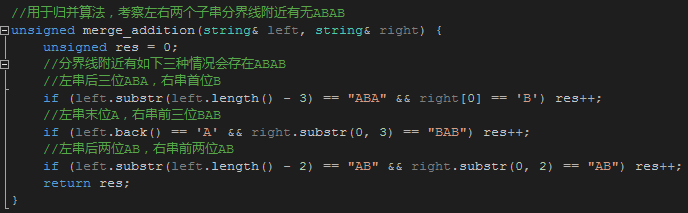
运用“分而治之”的思想。要统计整个字符串中“ABAB”子串的数目（），将整个字符串从中间一分为二，假设左右两边子串均已统计完毕（都是）。现在将两个子串合并，在合并处可能产生额外的“ABAB”子串，有如下三种情况：

图 11 归并算法的合并步

1. 左子串最后三位是ABA，右子串第一位是B。
2. 左子串最后一位是A，右子串前三位是BAB。
3. 左子串最后两位是AB，右子串前两位是AB。

这三种情况出现的概率都是。每种情况一旦出现，为“ABAB”子串的数目贡献1。因此有：



观察下图的递归树，其最后一行应有个，而圆圈所示的叶子共有个，又，因此有。























图12 递归树

代码如下：

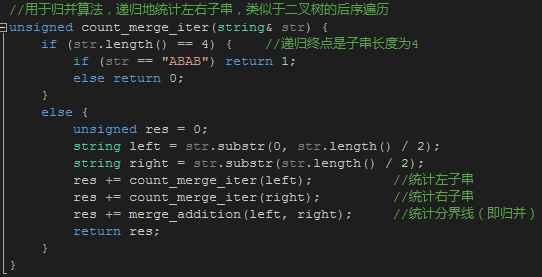


图13 归并法第一部分

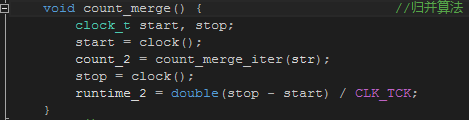


图 14 归并法第二部分

理论上其时间复杂度为。但其实际表现很不尽人意，主要可能是由于substr方法的复杂度是与所读取子串的长度近乎成正比的，这使得这段代码的实际复杂度接近。如果count\_merge\_iter使用指针或字符串迭代器作为参数（参考快速排序，并没有以容器作为函数参数），其时间复杂度可能有所降低。此外，递归与回溯过程需要的额外处理（如创建、销毁临时变量，尤其是字符串变量）也消耗了大量时间。

1. **折减法**

这个算法的思想是，要找到“ABAB”，首先要找到“AB”。在第一个“AB”之前的部分对整个字符串中“ABAB”的数目没有任何贡献，将其“折减”。详细步骤描述如下：

1. 逐位（一二位、二三位、三四位……）读取字符串，寻找“AB”。假设第k ~ (k+1)位出现了第一个“AB”。
2. 向后读取两位（读取方法是(k+2) ~ (k+3)位、(k+4) ~ (k+5)位、(k+6) ~ (k+7)位……），如果是“AB”，计1；直到读取的两位不是“AB”为止。假设第m ~ (m+1)位第一次出现不是“AB”的字符串。
3. 返回1）从第(m+1)位继续读，直到整个字符串读取完毕。

理论上，按这个算法也能够计算出。首先需要计算出首位“AB”出现位置的分布（第k位首次出现“AB”），有。还需要计算出在字符串头两位出现“AB”的条件下，“AB”连续出现次数的分布（这应该类似于几何分布），有。根据这两式及、，便可递归地计算出。只是这已经超出了笔者的能力范围。

代码如下：

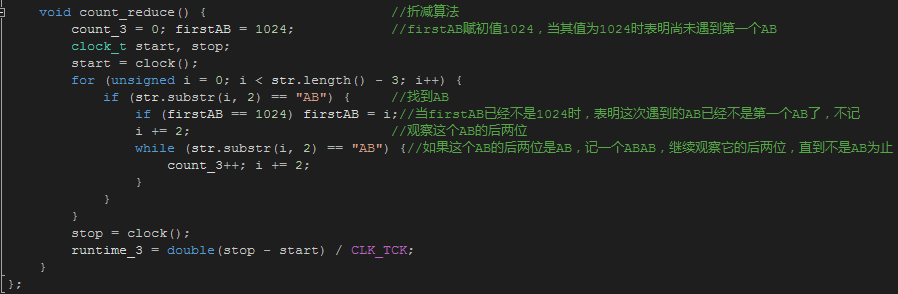


图 15 折减法

易知其时间复杂度为。根据上面的分析，其时间代价应小于逐四位统计法。实验结果是，在相同的样本容量下，它消耗的时间约为逐四位统计法的90%。

综上可以看出，往往越是高效的算法，人理解它的难度越大。由此带来的问题就是通过这样的算法来计算统计量难度也越大。要计算统计量，“简单粗暴”的算法就够了。

因为解决问题的算法是对普遍情况适用的，而期望恰是普遍情况的一个统计量。方差、矩等统计量或许也可以通过这样的统计算法得出，只是更加复杂，笔者遗憾没有更多的时间探索。