报告

一下规定所有字符串区间表示为左闭右开区间。

题目分析

基因的操作总共有一下几种操作：

1. DUP S a b 表示将 S 中 [a, b] 段的字符串复制一遍，插入到原字符串 [a, b] 后面。
2. INS S a b 表示在 S 的 a 位置插入一端长度为 b - a 的字符串。
3. DEL S a b 表示将 S 中 [a, b] 段的字符串从 S 中删除。
4. INV S a b 表示将 S 中 [a, b] 段的字符串用它的反向互补串代替。
5. TRA S1 a1 b1 S2 a2 b2 表示将 S1 中 [a1, b1] 段的字符串和 S2 中 [a2, b2] 段的字符串交换。

算法描述

对于给定的 DNA 字符串序列 S1，S2，S1 为原始的 DNA 序列，S2 为经过变化后的 DNA 序列。维护 i，j 分别表示当前匹配到的位置，对于 i 和 j 之后的字符串，根据不同情况进行判断出造成当前段不同的操作。

0. 如果 S1[i] 和 S2[j] 相同，检查 i + 1，j + 1。
1. 如果 i + k，j + k 之后的字符串相同，根据 S1[i ... i + k] 和 S2[j ... j + k] 的关系，可以分别推测出 INV，(可能的) TRA 操作。
   1.1 如果 S1[i ... i + k] 是 S2[j ... j + k] 的反向互补串，那么发生的是 INV 操作。
   1.2 否则，发生的是 TRA 操作。
2. 如果 i，j + k 之后的字符串相同，则是 INS 或者 DUP 操作。

   1.1 如果 S1[i - k ... i] 和 S2[j ... j + k] 相同，那么发生的是 DUP 操作。
   1.2 否则，发生的是 INS 操作。
3. 如果 i + k，j 之后的字符串相同，则是 DEL 操作。

细节说明

接下来需要探讨的是解法中描述的细节，如“之后的字符串相同”的定义。

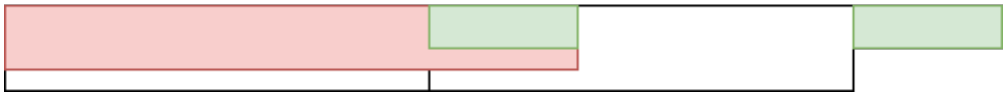
之后字符串相同

由于 DNA 序列的长度比较长，而 SV 操作的次数比较有限，所以可以认为相邻两个 SV 的操作的序列之间会有比较大的未被修改的序列，但是考虑到整个程序的效率问题，将 20 个字符作为上界，如果有 20 个字符相同，那么可以认定这两个位置之后的字符串相同。由于对于两个随机的 DNA 序列来说，每个字符相同的概率为 $\frac{1}{4}$，20 个字符相同的概率已经非常小了。

可能出现的位置指针移动过头

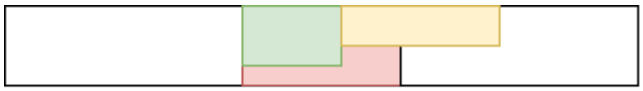
位置指针移动过头指原本 SV 的位置是 [a, b]，但是得到的字符串和原字符串的 [a...a+k] (0 < k < b - a) 相同，那么就会导致指针移过 a 但是没有检测到 SV。接下来讨论这种情况是否对 SV 的检测产生影响。

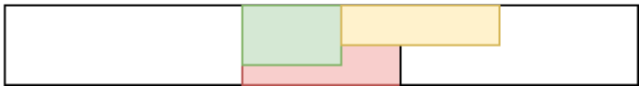
对于 DUP 操作，假定对 [a...a+k] 进行 DUP 操作，那么得到 [a...a+2k]，正常情况下会在 i，j 指向 a + k 时检测到不同，如果发生了位置指针的移动过头，说明 [a+k...a+2k] 和 [a...a+k] 的一个前缀相同，如下图所示，白色区域表示进行 DUP 的区间，绿色区域表示[a+k...a+2k] 和 [a...a+k] 相同的前缀，红色区域表示匹配出相同的区域。



那么此时会将 DUP 操作错认为时 INS 操作，插入的字符串为第二条 DNA 串中第一个绿色区域的结尾到第二个绿色区域的结尾。因此需要在检测 DUP 操作的时候枚举绿色区域的长度。出于对效率和成功率的权衡，一样选择了 20 作为枚举的上限。

对于 INS 和 DEL 操作，均会使插入/删除的区间进行一定的偏移，但是长度没有发生变化。





红色区域表示进行 SV 操作的区间，绿色区域表示位置指针移动过头的区域，黄色区域表示检测出来的 SV 操作的区间。

对于 INV 操作，这种情况发生在操作区间的前缀正好时后缀的反向互补串时，此时检测出来的 INV 操作长度会变短。

对于 TRA 操作，发生于两个区间的前缀或后缀相同的时候，此时发生 TRA 的两个区间的长度也会变短。

TRA 操作的检测

对于 TRA 操作，并非是将两个串放在一起查找，而是在发现 DUP 和 INV 操作均不是可能的操作的情况下，检查 (i, j + k)，(i + k, j)，(i + k, j + k) 哪一组是符合要求，来区分 INS，DEL，TRA 的一部分（暂时记作TRA0）。

在两个串都完成扫描后，对两个串的 TRA0 之间进行匹配，合并成真正的 TRA。

几个操作的优先判断程度

由于 INS，DEL，TRA 三个操作的通用性，所以这三个操作的优先度比较低，而 DUP，INV 两个操作的条件比较严苛。因此，首先判断是否是 DUP，INV 操作，然后再判断 INS，DEL，TRA 三个操作。

可能的位置指针越界情况

为了防止位置指针越界，在两个字符串的结尾分别加上一个不属于 DNA 序列的字符，这样能够有效防止越界，同时也简化了代码。

检测到非 TRA 的 TRA0

也就是在另外一个 DNA 序列中没有对应的 TRA0 变化，那么将这个变化视作对同一位置的 INS 和 DEL 操作各一次，虽然两次操作重叠是不满足题设的，但是至少是能够得到目标 DNA 序列的一个操作。实际运行过程中没有出现这样的情况。

算法效率分析

由于进行 SV 的次数为 50 次，所以检测到 S1[i] 和 S2[j] 不同的次数也是相同数量级的。

枚举 SV 的长度 1~1000，并进行不同的检测，每次检测循环 20 次。

最终程序循环次数为 $50 \times 1000 \times 20 = 10^6$ 量级，是一个效率比较高的方法。