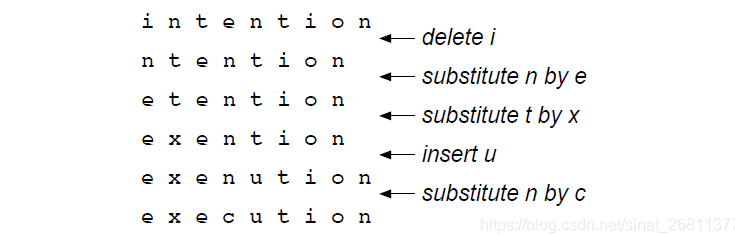
Through the detection of modern scientific and technological means, the genetic information of RNA is usually converted into a series of base sequence coding for representation and recording. Because RNA is not easy to preserve and extract, scientists often reverse transcrib it into DNA, and routinely use DNA instead when talking about genetic coding. So for any code representing RNA's genetic information, the only possible letters are A, C, G, and T, which stand for the four nucleotides that make up DNA -- adenine, cytosine, Guanine, thymine. Each letter represents a base, and they are arranged in unspaced rows to form a string such as "AAAGTCTGAC". Taking the above information into consideration, if the three mutations produced in the genetic process of RNA are converted into coding performance, the three mutations can be described as three kinds of editing for string insertion, replacement or deletion respectively, and the number of mutation points can be converted into the number of character editing times of string and the distance between two sequences. Is the minimum number of edits required to change a string into another string through the "insert, delete, and replace" editing operations, that is, the editing distance between the two strings. In the figure below, the original string is converted to the target string in 5 steps, and the editing distance between the two strings is 5.



Assuming that the string encoded by the source sequence has n characters and the string encoded by the target sequence has m characters, we need to find the minimum number of edits required to convert n characters from the source string to m characters from the target string.

Let's assume that and is the source string and the target string, respectively.

If the i’th character of X is the same as the j’th character of Y, just calculate the distance between and

If they are different, do the following:

1) Replace the I’th character of X, and then calculate the distance between

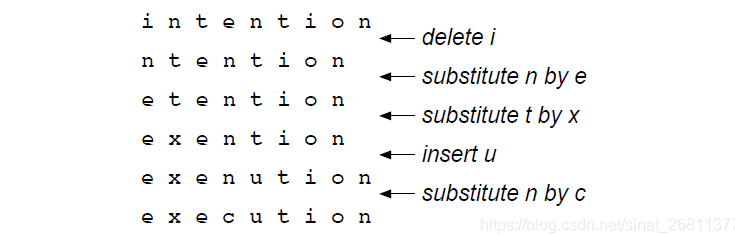
and

2) Insert the j’th character of Y before the i’th character of X, and then calculate the distance between and

3) Delete the ith character of X, and then calculate the distance between and

The above formula is not difficult to find using recursive functions. But direct recursion is a lot of double computation, and it's much more efficient to loop from the bottom up. In order to adopt the idea of circular solution, we use a matrix to save the ，that has been calculated at present, so that the data can be directly read from the matrix when it is needed for later calculation. In addition, can be obtained from or or which is equivalent to moving to from one of the three directions in the matrix. Therefore, there are three different moving directions in the matrix: To the left, up and up left, we can assign values to the three directions respectively, and then use another matrix to store the direction of movement, recording the shortest path.

通过现代科学技术手段检测，RNA的基因信息通常都转化为一串碱基序列编码进行表示与记录。因为RNA很不容易保存和提取，通常科学家们会将RNA反转录成DNA，在讨论基因编码时也习惯性地用DNA来代替，所以对于任意一条表示RNA基因信息的编码，其可能的字母只有A，C，G和T，分别代表组成DNA的四种[核苷酸](https://baike.baidu.com/item/%E6%A0%B8%E8%8B%B7%E9%85%B8?fromModule=lemma_inlink)——[腺嘌呤](https://baike.baidu.com/item/%E8%85%BA%E5%98%8C%E5%91%A4?fromModule=lemma_inlink)，[胞嘧啶](https://baike.baidu.com/item/%E8%83%9E%E5%98%A7%E5%95%B6?fromModule=lemma_inlink)，[鸟嘌呤](https://baike.baidu.com/item/%E9%B8%9F%E5%98%8C%E5%91%A4?fromModule=lemma_inlink)，[胸腺嘧啶](https://baike.baidu.com/item/%E8%83%B8%E8%85%BA%E5%98%A7%E5%95%B6?fromModule=lemma_inlink)。每个字母代表一种碱基，它们无间隔的排列在一起，形成一串如“AAAGTCTGAC”的字符串。综合以上信息考虑，如果将RNA的遗传过程中产生的三种突变转化为编码上的表现，三种突变就可以分别被描述为对字符串进行插入、替换或者删除三种编辑，而突变点的数量就转化为了字符串的字符编辑次数，两个序列之间的距离，就是把一个字符串通过“插入、删除和替换”这样的编辑操作变成另外一个字符串所需要的最少编辑次数，也就是两个字符串之间的编辑距离。如下图中通过5步将原字符串转化为目标字符串，两个字符串之间的编辑距离就是5。



假设源序列编码的字符串有n个字符，目标序列编码的字符串有m个字符，我们需要求解将源字符串的n个字符转换为目标字符串的m个字符所需要的最少编辑次数。

我们假设和分别为源字符串和目标字符串。

如果X的第i个字符与Y的第j个字符相同，只要计算和的距离即可

如果不相同，那么进行如下操作：

1）替换X的第i个字符，然后计算和的距离；

2）插入Y的第j个字符到X的第i个字符之前，然后计算和的距离；

3）删除X的第i个字符，然后计算和的距离；

上面的公式用递归函数不难求得。但直接递归会有很多重复计算，我们用从底向上循环求解的思路效率更高。为了能够采用循环求解的思路，我们用一个矩阵保存下来当前已经计算好了的，当后面的计算需要这些数据时就可以直接从矩阵读取。另外，由于求取可以从、或者三个方向计算得到，相当于在矩阵中是从三个中的某一个移动到，因此在矩阵中有三种不同的移动方向：向左、向上和向左上方，我们可以分别对三种方向赋值，然后用另外一个矩阵保存移动的方向，记录下最短的路径即可。