字符串匹配

1. 编辑距离算法

所谓编辑距离，就是用来计算从原串（s）转换到目标串(t)所需要的最少的插入，删除和替换的数目。

这里，我们举一个简单的例子，计算字符串“abc”“abe”的相似度。首先，需要构建一个二维数组，用来算最大编辑距离。如图：

	A	B	C	D	E
1		abc	a	b	c
2	abe	0	1	2	3
3	a	1	0	1	2
4	b	2	1	0	1
5	e	3	2	2	1

初始化二维数组，从 B2 到 E2 和从 B2 到 B5 顺次为 0，1，2... 下面我们需要填写这个二维数组的每一格数据，最终得到我们从 abc 到 abe 所需的最小编辑距离。

这里，每一个格子是这样的：

每一个格子由他的左边的格子上边的格子和左上角的格子决定。

具体来说，每一个方框的数字可以这样得来：

- 左边的数字加一 ；
- 上边的数字加一 ；
- 如果这一格对应的行和列字母不同的话， 左上角的数字加一，否则加零；
- 选择三个数中最小的一个。

以 C3 为例计算：

1. 左边数+1=2;
2. 上边数+1=2；
3. 这一格对应的行字母和列字母相同，左上角的数字加 0，即 0+0=0；
4. 选择最小数 0。

以 C4 为例计算：

1. 左边数+1=3;
2. 上边数+1=1；
3. 这一格对应的行字母和列字母不相同，左上角的数字加 1，即 1+1=2；
4. 选择最小的数 1.

算法理解：首先，第一行和第一列的所有数是由 0 写成指定位置字符串的步数；左边数和上边数加 1 的意思是先把自己删除，再写到目的字符串，这样办法也是最蠢的办法；对角数加 0 或加 1 很好理解，相同就不用做出改变，不同就在指定的加 1 个字母就可以了。

最后的计算公式：选择字符串 1->字符串 2 最小的步数为 a，两字符串长度最大值为 b，得 1-a/b;如上例即 1-1/3=0.66。

对应代码实现：(StrProc.cpp/GetStrMatchDegree)

https://github.com/andy-zha/trunk/tree/master/calculation/source/src