

项目说明文档

算法设计与分析

——assignment2

作者姓名： 沈星宇

学号： 1951576

指导教师： 罗烨

学院/专业： 软件学院 软件工程

同济大学

Tongji University

一、 算法思想

本题采用了动态规划的思想。

将锁的每个状态考虑成节点，拨动连续的 1-3 位数字考虑成边，则本题就是一张无向连通图，要求解的就是给定的两个字符串之间的最短路径。如果从起始状态出发，枚举生成图结构，再用广度优先搜索寻找最短路径，那么由于拨动数字的操作本质上是模 10 的加减，则会重复产生许多同一子问题的冗余（即重叠子问题），且最短路径一定是基于 1-3 位数字拨动这样的子问题的最优解（即最优子结构），因此考虑选用动态规划来解决。

设前 i 位都已经完成最优匹配，用一个多维数组 $dp[i][j][k]$ 来存储当前位置最短路径的值，第 $i+1$ 位已经加了 j ，第 $i+2$ 位已经加了 k 。进行初始化操作时，将每一个元素置为正无穷，将最初的状态 $dp[0][0][0]$ 置为 0，将找到的最小值作为元素存入数组中，则 $dp[len][0][0]$ 就是所求。

本题因为数字能向上或向下两个方向旋转，因此在两个字符串对应位之间的差值存在两个，即 $differ1$ 与 $differ2$ ，由于模 10 的关系，它们之间存在 $differ1+differ2=10$ 的关系。考虑只拨动第 $i+1$ 位的时候，在前 i 位匹配的状态下， $newStep$ 相较增加的步数即是 $differ1$ 或 $differ2$ ，但因为考虑拨动连续 2 位和 3 位的情况，里面嵌入两层循环，搜索最短路径存入

二、 复杂度分析

```
void run() {
    for (int i = 0; i < _len; i++) {
        for (int j = 0; j < 10; j++) {
            for (int k = 0; k < 10; k++) {
                if (dp[i][j][k] != INF) {
                    //for the digits can be rotated up or down, so there are two direct differences between _cur[i] and _des[i]
                    //因为数字能被向上或向下旋转，因此在_cur[i]和_des[i]之间直接的差值有两种
                    int differ1 = (_des[i] - _cur[i] - j + 20) % 10;
                    int differ2 = 10 - differ1;

                    _newStep = dp[i][j][k] + differ1;
                    for (int m = 0; m <= differ1; m++) {
                        for (int n = 0; n <= m; n++) {
                            if (_newStep < dp[i + 1][(k + m) % 10][n]) {
                                dp[i + 1][(k + m) % 10][n] = _newStep;
                            }
                        }
                    }

                    _newStep = dp[i][j][k] + differ2;
                    for (int m = 0; m <= differ2; m++) {
                        for (int n = 10; n >= 10 - m; n--) {
                            if (_newStep < dp[i + 1][(k - m + 10) % 10][n % 10]) {
                                dp[i + 1][(k - m + 10) % 10][n % 10] = _newStep;
                            }
                        }
                    }
                }
            }
        }
    }
}
```

(图 1 核心代码截图)

basic operation 是赋值语句

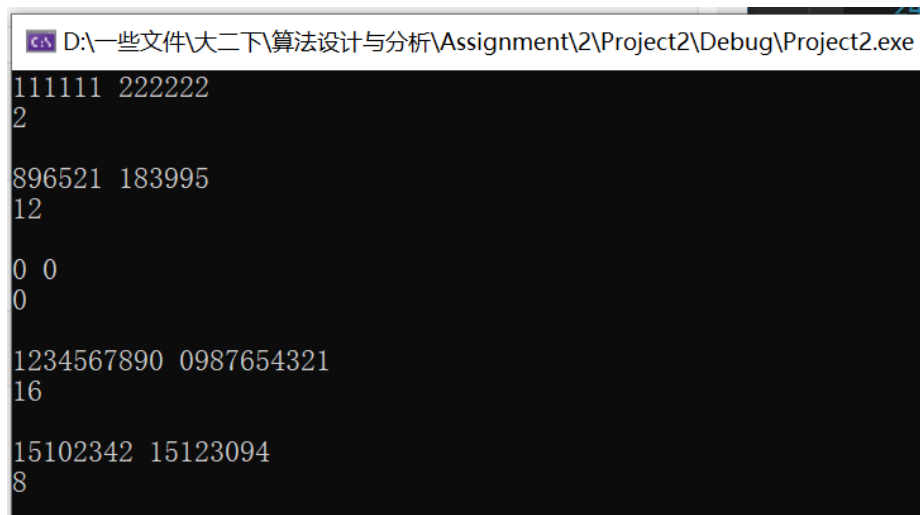
$$C(n) = 100n * ((1+2+3+...+differ1) + (1+2+...differ2))$$
$$= 100n * (2differ1^2 - 20differ1 + 110)$$

考虑最好的情况， $differ1==5$ 时， $C(n)=6000n$

考虑最坏的情况， $differ1==0$ 或 $differ1==10$ 时， $C(n)=11000n$

则总的时间复杂度为 $O(n)$ ，空间复杂度为 $O(n)$

三、 运行结果



A screenshot of a Windows command prompt window. The title bar at the top reads "D:\一些文件\大二下\算法设计与分析\Assignment\2\Project2\Debug\Project2.exe". The command prompt area has a black background with white text. The output consists of several lines of numbers, some on single lines and some on pairs of lines. The text is as follows:

```
111111 222222
2

896521 183995
12

0 0
0

1234567890 0987654321
16

15102342 15123094
8
```

(图 2 运行结果截图)