Undecodable Codes题解

【题意概述】

给定一个字符编码表，求出最短的一个编码，使得它是至少两种字符串的编码。长度相同时输出字典序最小的。（字符集<=20，每个字符编码长度<=20）

【算法分析】

考虑我们如何构造两个字符串的编码相同：每次向编码较短的那个字符串添加字符使得能够与较长的那个字符串匹配。那么，影响我们匹配的状态只有较长一个字符串的最后一个字符，以及当前匹配的位置。

00110010101110000011000011

00110010101110000011000

这个状态应该是0011对应字符编号和已经匹配的长度1

将这两个值作为DP的状态，DP值为最短长度，我们尝试对它进行求解，但是我们很快发现，这个DP其实是可能循环的，因为我们很有可能在中间插入了一些无用的串，最简单的，如下图

001100110011001100110011

0011001100110011001100

显然，这是没有用的，但我们递归处理的时候会进入死循环。考虑到绕圈无意义，解决这种问题的方法是将每个DP状态看做一个点然后进行最短路的处理。枚举第一个串的第一个字符，然后进行最短路，直至恰好完全匹配即可。

下面我们来分析一下复杂度。DP状态共计20\*20=400个，即400个点的最短路，每个状态最多和20个字符分别匹配，边数为8000，而建边时暴力的比较算法可以做到20\*边数的复杂度内建边，更好的做法是KMP，可以做到点数^2\*长度。

这题到这里好像已经解决了，但是我们重新审视题目，发现要求字典序最小。而显然朴素的最短路是不能做到这一点的，需要加第二关键字，但是这个关键字是一个字符串，这就大大增加了比较和复制的难度，原复杂度\*=答案长度。事实上，由最短路构图可知答案长度<=8000，总复杂度8000\*8000\*log400，这是很难通过多组测试数据的。

然而，这个算法在现场只需要0.003s即可通过全部测试数据。究其原因，我们发现：很难用20个长度为20的串构造出长达8000的答案，事实上，超过200的答案都很少见。另外，这400个点并不是每个点都有20条边，而绝大多数情况下每个点的期望出度仅为3，这也大大提升了程序效率。最后即比较。考虑两个状态长度相同末尾也相同，如果字典序不同，则中间相当长的部分必须被等效替代，这也使得字符串比较的短路判断提升了速度。

综上，该算法在本题效果较为良好。

【附注】

出数据时我意外的发现，这题现场的标准算法似乎是搜索。其实，如果答案大概只有10个字符组成，那么搜索算法效果其实并不差，但需要实现的较为精细，如预处理搜索路径和匹配情况，最优性剪枝等。实际上，将搜索路径详细描述后即为本文算法的状态，所以如果加入map判重很可能会使搜索算法复杂度变为多项式级别，甚至比本文算法更加快速。

目前已知的较大答案例子：

考虑循环串1111010000101100，它的16个长度为4的子串互不相同。以每个位置开头长度为15的子串共16个，加上任意一个长度为11的子串，即可构造出答案251。

数据大部分由随机+构造+退火保证有解。