中国科学技术大学 University of Science and Technology of China | 网络空间安全学院 School of Cyber Science and Technology

作品类别： ☑软件设计 ☐硬件制作 ☐工程实践

《密码学导论》课程大作业作品设计报告

作品题目：半自动单表代换解密工具

团队名称：单人没有队

团队人员：董睿 PB23081522

2025 年 6 月 8 日

基本信息表

作品题目：半自动单表代换解密工具

作品内容摘要：

本作品是以Unity引擎为基础，使用C#语言编写的一款单表代换解密工具。

本作品界面友好、易用，通过嵌入式命令行工具和多按钮实现交互，且各模块数据实时更新，保证使用便捷。

本作品引入了秘钥猜解矩阵的设计，摆脱了明文密文一一对应猜解带来的局限，通过一明文对应多密文的概率分布和一密文对应多明文的概率分布避免一一对应猜解带来的过度自信问题，同时引入撤回重做功能方便纠错。

本作品支持自动推理秘钥，根据当前秘钥和密文生成猜解概率矩阵，通过单字母频率分析、多字母模式分析、字母连接分析和首尾前后缀分析自动进行最佳猜解。

关键词（五个）：

单表代换、自动猜解、秘钥猜解矩阵、易于交互、撤回重做

团队成员（按在作品中的贡献大小排序）：

序号	姓名	学号	任务分工
1	董睿	PB23081522	代码编写、报告撰写、UI 绘制
2			
3			

1.作品功能与性能说明

作品功能：

（1）手动秘钥调整：

界面左侧的秘钥调整区支持手动进行秘钥猜解设置。将A~Z编号为1~26，则第i行第j个按钮表示i是否有可能被加密为j，如果有可能，则显示为绿色；否则显示为红色。点击按钮可以改变其颜色，将猜测置反。当第i行只有一个按钮j为绿时，表示i被确定加密为j，此时会自动排除其他字母加密为j的情况。

（2）实时解密：

根据密文和秘钥调整区的展示情况，实时将解密后的明文结果更新在明文显示区。还没有对应明文的密文会实时解密为'?'字符。

（3）内置命令行：

① help：帮助文档

② guess：进行秘钥猜解区的便捷批量调整

③ suggest：根据当前秘钥猜解区和密文，通过单字母频率分析，2/3/4字母模式分析，字母连接规律分析和单词首尾的前后缀分析等进行猜解，给出最可能的5个猜解和最不可能的5个猜解

④ undo：撤销上一步对秘钥猜解区的更改

⑤ redo：重做上一步的撤销（撤销后再进行其他操作将无法重做）

⑥ auto：自动根据suggest修改秘钥调整区。

性能说明：

1. 整体时间性能：运行帧率取决于加密文本长度L，每帧解密时间复杂度O(L)，详细数值见性能测试。

2. suggest指令和auto指令时间性能：一次指令时间复杂度O(LlogL)

3. 空间性能：额外持续增加的空间开销集中在撤回重做历史记录，详细数值见性能测试。

2.设计与实现方案

2.1 实现原理

（1）Unity Scene 搭建:

Main Camera:提供场景视野

Canvas:UI 附着的父物体，包括 Key（秘钥猜解区）、CipherText（密文输入区）、PlainText（明文显示区）、Command（指令输入区）、Feedback（指令回显区）、Error（报错区）、Submit（指令提交）、Clear（指令回显区和报错区清屏）、Reset（程序重置）、Quit（退出）。

Key: 可滚动展示，包含 26 个可横向滚动展示行（单个猜解区），每个行包括 26 个 GuessKey 按钮，与全局唯一变量中的 guessKey 变量中的成员绑定并添加监听事件。

UIDrawer:组件 DrawUI 在程序第一帧开始前调用一次 Start()，批量绘制 UI 组件并与全局唯一变量的需要值一一绑定，为按钮添加监听事件。

Decoder:组件 Decode 每帧解密一次密文得到明文，通过全局唯一变量中的 plainText 更新到明文显示区。

CommandExecutor: 组件 Command 的方法 Execute 接收 Submit 按钮监听事件传递的参数 command，对 command 解析后传递给各自方法。

ErrorCollector: 组件 ErrorCollect 收集上述流程中产生的所有报错并回显在报错区。

（2）自定义类型:

Guess{　　明文猜解情况

bool sure 是否确定密文，true 表示可能

int sureChar 如果已确定，是哪个密文

bool[] guesskey 明文是否有可能加密为密文 i，true 表示不可能

}

NGramFreq{　n 字母分布

string ngram n 字母串

double frequency 出现频率

}

(3) 全局唯一变量:

GlobalVariable.instance 作为全局唯一变量实例,绑定所有具有唯一性的元素

(4) 命令行实现:

[1] guess: 根据参数，判断需要更改哪些密钥矩阵的点，将这些点压入一个 List，将 List 传给 InferChain 方法，该方法进行自动推断 List 中点的改变是否引发其他点改变，检查这些改变是否引发逻辑冲突。如果发生冲突则抛出错误，否则将其他被改变点也压入 List，ChangeGuess 方法逐个改变 List 中点对应的全局唯一变量数值，最后由 LockCheck 方法检查是否有密文字母被成功解密并将结果反馈到秘钥猜解区。

[2] suggest: 生成初始概率矩阵和反馈报告，进行 1 字母频率分析，2/3/4 字母模式分析，字母连接分析和单词开头结尾前后缀分析。

① 1/2/3/4 字母分析: 找出最多 10 个密文中的连续 n 字符，满足: 出现频率最高; n 字符不能被现有秘钥矩阵完全解密。对每个这样的 n 字符串 s，在提供的参考 n 字母频率表中找出最多 10 个 n 字母串 t，满足: 出现频率最高; s 和 t 在现有秘钥矩阵约束下不存在加解密冲突。然后，对于 s 和 t 每个对应位置形成的字母对，提高其在概率矩阵中的映射概率。提高幅度由 s 与 t 频率相似度、s 与 t 已知确定对应的位置数目、找到的 t 的个数共同决定。

② 单词开头/结尾/任意位置字母连接分析: 通过提取文本中的单词、与已提供的前缀/后缀/常见连接对应，为符合规律的猜测加权，为违反规律的规则惩罚。

③ 归一化概率矩阵。

④ 去除由当前密钥矩阵已确定的映射，为剩余映射排序，得到最可能的 5 个猜解和最不可能的 5 个猜解。

[3] auto: 自动提取 suggest 中最可能的猜解并实施猜解。

[4] undo/redo: 在 InferChain 方法结束后，将新的 List 压入 Stacl<List> history 栈。undo 时，只需将 history 出栈得到的 List 传给

ChangeGuess 方法，并将该 List 压入 redo 栈。若发生 undo/redo 以外的对密钥矩阵的更改操作，将 redo 栈置空。

2.2 参考文献

密码学导论教材

2.3 运行结果

2.4 技术指标

加密算法：单表代换加密

破解算法：单字母频率分析、多字母模式分析、字母连接分析、单词前缀分析、单词后缀分析

破解算法时间复杂度：O(LlogL)（L 为密文长度）

程序运行实测帧率：400~1000FPS（取决于文本长度）

程序运行实测内存占用：400MB~600MB（取决于文本长度和历史操作记录长度）

程序运行实测 CPU 占用：<1%

安全指标：历史记录超过栈上限会自动舍弃栈底内容，不会导致内存过高。但并未限制密文长度，可能引发内存问题。

跨平台兼容性：借助 Unity 跨平台 Build，通过源代码为不同平台构建后可在各平台(Windows,Mac,Linux 等)直接运行，无须其他环境依赖。

3.系统测试与结果

3.1 测试方案

1.测试对给定密文的自动破译能力：反复使用 auto 指令自动破译以下无标点、空格的文本

```
AHNFCROACOAHNISEFLCIASFVCNWOSEAHNWSRLDWC
AHCEIRNTONDATRCFFOSEOANNLTEDTLUMCEUMFRSM
AHNNUITETDTTEDJTPTEAHNUOWCNLDOCAOATRCFFB
TASETGTCEOACAOARTDCACSETLTLLCNOTAAHNOTMN
ACMNCACODNMTEDCEGAHTATFRCITEISUEARCNOOUI
HTORWTEDTTEDUGTEDTRNMSVNCMPSRAATRCFFOSEO
NISEDHTEDILSAHNOFRSMAHNUOAHTAVCNWSEAHNWS
RLDCOKESWETOTMNRCITFCROATEDCAMNTEOAHTATE
YSENONNETOHTVCEGMSVNDTMNRCITOIHNNONBNCAT
ETLLYSRESACOPUECOHNDBYISEARTOAIHCETONNOC
AORNLTACSEOWCAHAHNWSRLDTOBTONDSEMUAUTLRN
OPNIAFTCRENOOJUOACINTEDWCEWCEISSPNRTACSE
```

2. 测试反复修改密钥矩阵增加历史记录容量对内存的影响
3. 测试使用超长文本(3000 字符)对帧率和内存的影响

3.2 功能测试

程序的 auto 指令在第 16 个密钥处出现一次猜解错误，误将 U 解密为 M。通过 undo 回到第 15 步猜解后第 16 步猜解前，使用 suggest 指令观察，发现给出 M->U 概率%8.1325，F->F 概率%8.1320。输入 guess 0 F F 使程序按可能性第二大的建议继续猜解，最终顺利完成了全文本猜解。

3.3 性能测试

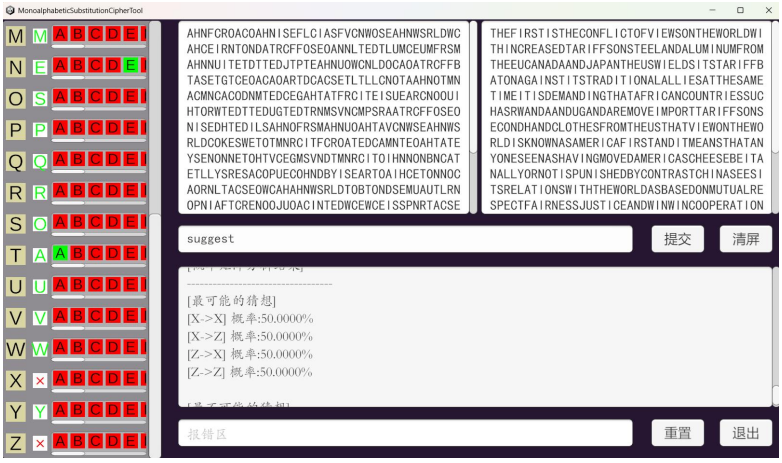
使用 Unity Editor 调试模式，获取 3.2 中密文使用时的帧率和超长密文使用时的帧率。

使用任务管理器查看程序单独运行时，3.2 中密文使用时的内存占用和超长密文使用时的内存占用。

使用任务管理器查看进行 1000 次操作后，超长密文使用时的内层占用。

3.4 测试数据与结果

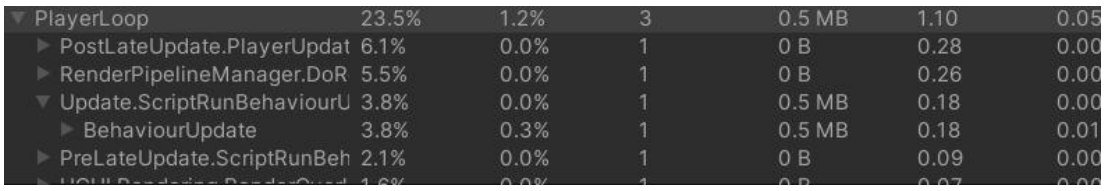
功能测试结果：运行结果良好，auto 功能准确率非常高，即使出现错误也可以通过 undo 选择其他高概率建议路径达成正确解密。



图中 X, Z 无法猜解，因为文本中未出现这两个字符。

性能测试结果：

短文本时

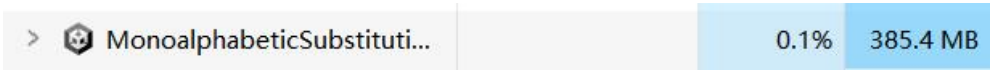


单帧耗时 1.10s，帧率 909FPS

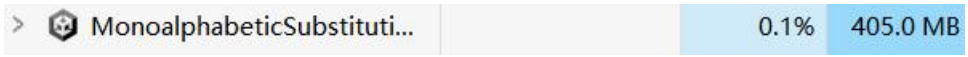
长文本时

单帧耗时 2.49s，帧率 401FPS

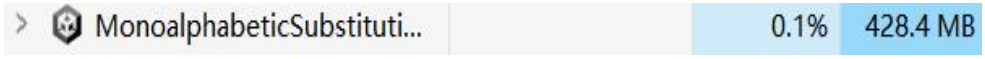
短文本时内存占用



长文本时内存占用



历史操作 1000 次时，长文本时内存占用



4.应用前景

可辅助对单表代换的密码破译，用于传统密码破译工作、传统密码教学工作等。

5.结论

本作品实现了操作便捷的单表代换工具，通过调整概率矩阵和事先约束密钥矩阵的方式自动生成最有可能的猜解，实现了较稳定的单表代换解密。功能测试符合预期，性能测试结果良好，便于部署于 PC 使用。