S-DES 加密解密程序开发手册

1. 开发环境

- 操作系统：Windows 10
- 编译器：IntelliJ IDEA 2023.1.6
- Java 版本：Java SE 1.8

2. 算法实现

主要算法实现是在 src/function 目录下.

2.1 SDES_Encrypt——加密

S-DES 加密包含两个重要部分：生成子密钥和 f 函数；本程序将生成子密钥的代码封装在 2.3 KeyGeneration 中。

- 初始化了一些置换表
- 定义了 `public static int[] initialPermute(int IP[],int plainText[])`进行初始 IP 置换；
- 定义了 `private static int[] SBoxSubstitute(int[] temp)`进行 S 盒替换；
- 定义了 `static int[] roundFunction(int[] roundKey, int[] inputBlock)`轮函数：负责进行扩展、置换、S 盒转换等操作；
- 定义了 `public static int[] finalPermute(int[] result, int[] IP_inv)`进行最终 IP 逆置换；

2.2 SDES_Decrypt——解密

- 定义了 `public static int[] decrypt(int[] ciphertext, String key)`负责解密；

- 定义了 `public static String decryptASCII(String ciphertextASCII, String key)` 负责对 ASCII 类型进行解码。

2.3 KeyGeneration——生成子密钥

- 定义了 `private static int[] permutation(String input, int[] permutationTable)` 作为初始置换函数；
- 定义了 `private static int[] leftMove(int[] array)` 作为左循环移位函数；
- 定义了 `private static int[] compressPermute(int[] input, int[] permutation)` 进行压缩置换；
- 定义了 `public static int[][] generateKey(int[] P10, int[] P8, String key)` 通过调用以上函数，生成子密钥。

2.4 encryptionASCII——第三关：对 ASII 类型进行加密

- 定义了 `public static String EncryptASCIIfunction(String PlainTextASCII, String key)` 对 ASCII 进行加密操作。

2.5 BruteForceAttack——第四关：暴力破解

我们输入三对密钥已知的明密文对，并依次对其进行暴力破解，得到的密钥与已知的不同，这一现象在第五关有详细测试。同时，我们统计每个解密用时，并计算了平均解密时间。

- **输入数据**
  - **明文列表**(plaintextList)：一系列已知的明文，每个明文由 8 位二进制数表示。

- **密文列表**(ciphertextList)：与明文列表对应的一系列密文，每个密文由 8 位二进制数表示。

➢ **输出数据**

- **找到的密钥**：能够将密文解密成明文的 10 位二进制密钥。
- **解密时间**：找到密钥所需的时间。
- **平均解密时间**：所有找到的密钥的平均解密时间。

➢ **算法描述**

- 初始化计时器和找到的密钥计数器。
- 遍历所有可能的 1024 个密钥。
- 对每个密钥，将其转换为 10 位二进制字符串。
- 使用当前密钥解密给定的密文。
- 将解密结果与明文进行比较。
- 如果匹配，记录当前密钥和解密时间。
- 计算并输出所有找到的密钥的平均解密时间。

```java
// 明密文对
//key=1111100000
plaintextList.add(new int[]{0, 1, 0, 1,0, 1, 0, 1});
ciphertextList.add(new int[]{0, 0, 0, 1, 0, 1, 1, 1});

//key=1010101010
plaintextList.add(new int[]{0,0, 0, 0, 0,0, 0, 0});
ciphertextList.add(new int[]{1, 1, 1, 1, 1, 0, 1, 0});

//key=1011010010
plaintextList.add(new int[]{ 0, 0, 1, 0, 1,1,1,1});
ciphertextList.add(new int[]{1, 1, 0,0, 0, 0, 0, 1});

for (int num = 0; num < plaintextList.size(); num++) {
    int[] plaintext = plaintextList.get(num);
    int[] ciphertext = ciphertextList.get(num);
    long startTime = System.nanoTime(); // 记录开始时间（纳秒）

    for (int i = 0; i < 1024; i++) { // 2^10 = 1024
        // 将i转换为10位二进制形式作为密钥
        String binaryKey = Integer.toBinaryString(i);
        while (binaryKey.length() < 10) {
            // 补充前导0，确保密钥长度为10位
            binaryKey = "0" + binaryKey;
        }

        // 将测试秘钥与秘文进行输入解密函数中进行解密，得到测试明文
        int[] decrypted = SDES_Decrypt.decrypt(ciphertext, binaryKey);
        // 将测试明文与实际明文进行比较
        if (Arrays.equals(plaintext, decrypted)) {
            long endTime = System.nanoTime();
            long decryptionTime = endTime - startTime;
            totalTime += decryptionTime; // 累计解密时间
            KeysNumber++;
            System.out.println("Possible key is:  " + binaryKey);
            System.out.println("Crack Time: " + decryptionTime / 1_000 + " microsecond");
            break;
        }
    }
}
```

2.6 ClosedBeta——第五关：封闭测试

我们对 S-DES 算法进行了封闭测试，讨论了 S-DES 算法的安全性。通过输入一组已知密钥的明密文对后解密，得到多个不同密钥。

➢ 算法描述

- 准备一个固定的明文数组 plaintext 和对应的密文数组 ciphertext。
- 遍历所有可能的 10 位二进制密钥。
- 使用每个密钥解密 ciphertext。

- 检查解密结果是否与 plaintext 相同。如果相同，记录该密钥。

```
Yuetong
public static void main(String[] args) {
    // 存储找到的密钥
    List<String> foundKeys = new ArrayList<>();

    //key=1111111000
    int[] plaintext = {1,1,1,1,1,1,1,1};
    int[] ciphertext = {0,1,0,1,1, 1, 1, 1};

    for (int i = 0; i < 1024; i++) {
        // 将i转换为10位二进制形式作为密钥
        String binaryKey = Integer.toBinaryString(i);
        while (binaryKey.length() < 10) {
            // 确保密钥长度为10位
            binaryKey = "0" + binaryKey;
        }
        // 解密
        int[] decrypted = SDES_Decrypt.decrypt(ciphertext, binaryKey);
        // 比较是否一致
        if (Arrays.equals(plaintext, decrypted)) {
            foundKeys.add(binaryKey); // 将找到的密钥添加到列表中
        }
    }
```

3. 页面设计

我们主要基于 Java Swing 进行 GUI 设计，代码位于 src/function 中的 ui.java 中。

➢ 应用程序窗口

- **窗口标题**: "S-DES 加密解密"
- **窗口大小**: 530 x 630 像素
- **窗口风格**: Nimbus
- **窗口设置**:
  - 不可调整大小（setResizable(false)）
  - 默认关闭操作（setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)）

- 显示状态（setVisible(true)）

➢ 页面组件

面板（JPanel）

标签（JLabels）

- jlab1: "请输入加解密使用的 10 位二进制密钥"
- jlab2: "请输入加解密对应的原文或密文"
- jlab3: "请选择编码类型"
- jlab4: "结果"

文本字段（JTextField）

- keytext: 用于输入 10 位二进制密钥
- text: 用于输入原文或密文
- output: 显示结果

按钮（JButtons）

- btn1: "解密"
- btn2: "加密"
- btn3: "全部重置"

单选按钮（JRadioButtons）

- bitButton: "Bit"
- asciiButton: "ASCII"

按钮监听器

- btn1（**解密**）: 检查输入的密钥和数据长度，执行解密操作，并显示结果。

- **btn2（加密）**：检查输入的密钥和数据长度，执行加密操作，并显示结果。
- **btn3（全部重置）**：清空所有文本字段。

布局

- 使用 Box 来创建垂直和水平的布局。
- 组件被添加到 vBox 和 hBox 中，然后这些盒子被添加到 JPanel。
- JPanel 设置为背景颜色，然后添加到 JFrame 的内容面板。