Отчёт по лабораторной работе №2

Шифры перестановки

Артур Арменович Давтян

Содержание

Список таблиц

Список иллюстраций

1 Цель работы

Ознакомиться с шифрами перестановки и обучиться их программной реализации.

2 Задание

- Реализовать маршрутное шифрование;
- Реализовать шифрование с помощью решёток;
- Реализовать таблицу виженера.

3 Теоретическое введение

При подготовке использовалась методичка со страницы курса в ТУИС.[1]

Шифрование перестановкой заключается в том, что текста переставляются по определенному правилу. Простейшим примеров перестановочного шифра являются так называемые «маршрутные перестановки», использующие некоторую геометрическую фигуру (плоскую или объемную). Шифрование заключается в том, что текст записывается в такую фигуру по некоторой траектории, а выписывается по другой траектории. Пример — маршрутные шифры перестановки, основанные на прямоугольниках (таблицах). Шифруемое сообщение в этом случае записывается в прямоугольную таблицу по маршруту: по горизонтали, начиная с верхнего левого угла, поочередно слева направо.

Представим, что у нас есть зашифрованное сообщение (рис. 3.1)

П	Р	И	М	Е	Р	М
А	Р	Ш	Р	У	Т	Н
О	Й	П	Е	Р	Е	С
Т	А	Н	О	В	К	И

Рис. 3.1: Пример шифрования перестановкой

Тогда наше зашифрованное сообщение: ПАОТРРЙАИШПНМРЕОЕУРВРТЕКМНСИ.

Шифрование с помощью решёток применяется для защиты информации, представляющую ценность в течение ограниченного времени (несколько часов). Этот шифр также является перестановочным, т.е. криптограммы этого шифра представляют собой анаграммы открытого текста. Данный метод шифрования активно применялся во время второй мировой войны, и до сих пор используется в качестве армейского шифра.

Алгоритм шифрования следующий (рис. 3.2):

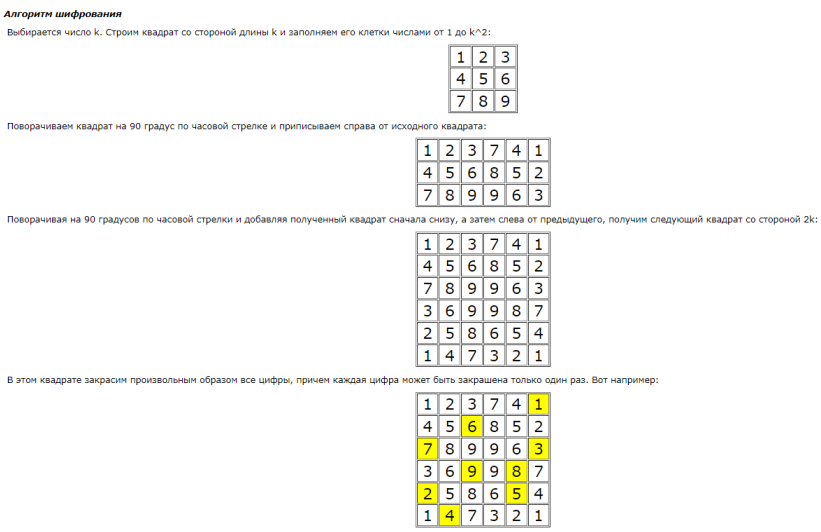


Рис. 3.2: Алгоритм шифрования с помощью решеток

Шифр Виженера — метод полиалфавитного шифрования буквенного текста с использованием ключевого слова. Этот метод является простой формой многоалфавитной замены. Метод прост для понимания и реализации, он является недоступным для простых методов криптоанализа.

Алгоритм следующий: у нас есть сообщение, которое мы хотим зашифровать, и кодовое слово. Кодовое слово надо повторять до тех пор, пока количество букв не станет равным количеству букв нашего сообщения. Тогда, если буквы A-Z соответствуют числам 0-25, то шифрование Виженера можно записать в виде формулы:

$$C_i = (P_i + K_i) mod 26$$

А расшифровку:

$$P_i = (C_i - K_i + 26) mod 26$$

4 Выполнение лабораторной работы

Работа была выполнена на языке программирования Python.

Реализуем маршрутное шифрование (рис. 4.1):

```python
# Encryption
def encryptMessage(msg, key):
    cipher = ""

    # убираем пробелы, потому что мы их ненавидим
    msg = msg.replace(' ', '')

    # берём длину текста
    msg_len = int(len(msg))

    # создаём список букв этого текста
    msg_lst = list(msg)

    # сортируем буквы ключа по алфавиту
    key_lst = sorted(list(key))

    # считаем количество стобцов
    col = len(key)

    # считаем количество строк
    if msg_len % col == 0:
        row = int(msg_len / col)
    else:
        row = int(msg_len // col) + 1

    # добавляем английскую a, если наша таблица не полная
    # не знаю, почему именно её. так в примере, а я повторюшка
    fill = int((row * col) - msg_len)
    msg_lst.extend('a' * fill)

    # создадим матрицу нужного размера для шифрования
    matrix = [msg_lst[i: i + col] for i in range(0, len(msg_lst), col)]

    # читаем получившуюся матрицу постолбцово (?)

    for i in range(col):
        # так как до этого мы сортировали в алфавитном порядке,
        # теперь нам надо найти эти буквы в исходном ключе
        # и взять их порядковый номер на рукаве
        curr_idx = key.index(key_lst[i])
        # и соединить это всё в одну строку
        cipher += ''.join([row[curr_idx] for row in matrix])

    return cipher
```

Рис. 4.1: Программная реализация маршрутного шифрования

И расшифровку (рис. 4.2):

```python
def decryptMessage(cipher, key):
    msg = ""

    # берём длину текста
    msg_len = int(len(cipher))

    # создаём список букв этого текста
    msg_lst = list(cipher)

    # считаем количество стобцов
    col = len(key)

    # считаем количество столбцов
    # по логике этого типа шифрования, у нас будет всегда получаться целое число
    row = int(msg_len / col)

    # сортируем буквы ключа по алфавиту или возрастанию
    key_lst = sorted(list(key))

    # создаём пустую матрицу размера, который мы высчитали
    dec_cipher = []
    for _ in range(row):
        dec_cipher += [[None] * col]

    # Arrange the matrix column wise according
    # to permutation order by adding into new matrix

    # счётчик для номера элемента в списке букв нашего текста
    # не придумал, как сунуть это в цикл, поэтому решил проблему так
    msg_indx = 0

    for i in range(col):
        # так как до этого мы сортировали в алфавитном порядке,
        # теперь нам надо найти эти буквы в исходном ключе
        # и взять их порядковый номер на рукаве
        curr_idx = key.index(key_lst[i])

        for j in range(row):
            dec_cipher[j][curr_idx] = msg_lst[msg_indx]
            msg_indx += 1

    # объединяем воедино
    msg = ''.join(sum(dec_cipher, []))

    return msg
```

Рис. 4.2: Программная реализация расшифровки маршрутного шифрования

Сам процесс шифрования и расшифровки (рис. 4.3):

Шифрование

In [4]:
```
while True:
    msg = input(bold + "What message do you want to encrypt?\n" + end + "Note that only russian and english characters and space
    if (False in [x in a for x in msg]):
        continue
    else:
        break

while True:
    key = input(bold + "\nEnter the key\n" + end + "Note that repeated characters are prohibited:\n")
    if len(set(key)) != len(key):
        continue
    else:
        break

print("\nYour encrypted message is: " + bold + ul + encryptMessage(msg, key))
```

```
What message do you want to encrypt?
Note that only russian and english characters and space are allowed:
&
What message do you want to encrypt?
Note that only russian and english characters and space are allowed:
?
What message do you want to encrypt?
Note that only russian and english characters and space are allowed:
привет

Enter the key
Note that repeated characters are prohibited:
нор

Your encrypted message is: пвреит
```

Расшифровка

In [5]:
```
msg = input("What message do you want to decrypt? ")
key = input("\nEnter the key: ")

print("\nYour decrypted message is: " + bold + ul + decryptMessage(msg,key))
```

```
What message do you want to decrypt? пвреит

Enter the key: нор

Your decrypted message is: привет
```

Рис. 4.3: Шифрование и расшифровка методом маршрутного шифрования

Шифрование методом таблицы Виженера (рис. 4.4):

```python
# повторяем (убираем?) буквы ключа до тех пор, пока не станет
# столько же, сколько у сообщения
def genKey(msg, key):
    key = list(key)
    if len(msg) == len(key):
        return(key)
    else:
        for i in range(len(msg) -
                       len(key)):
            key.append(key[i % len(key)])
    return("" . join(key))
```

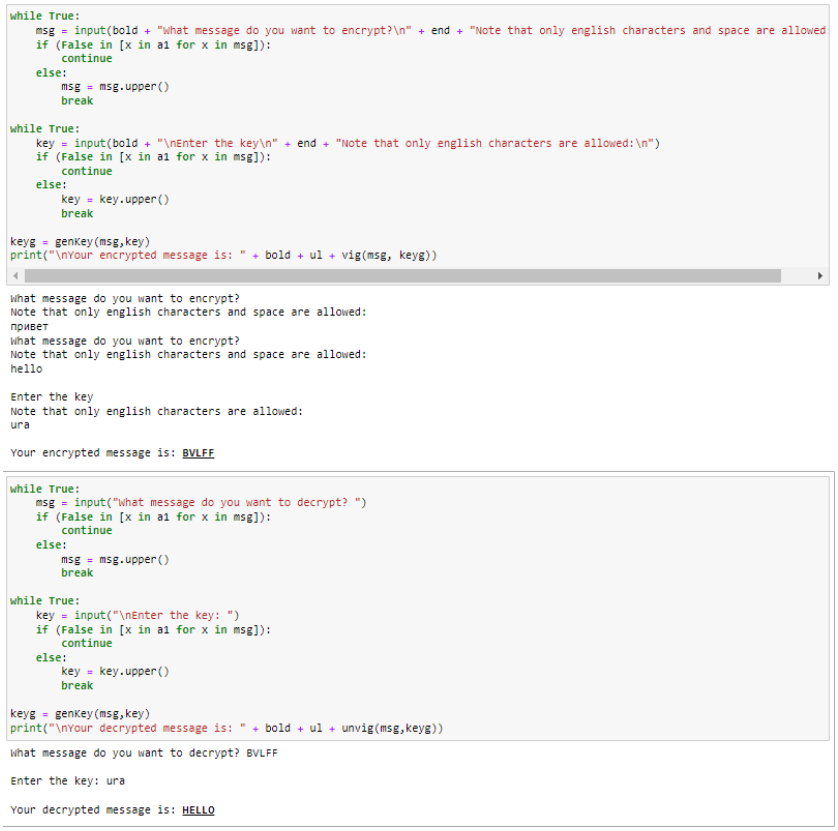
```python
# шифрование
def vig(msg, key):
    cipher_text = []
    for i in range(len(msg)):
        x = (ord(msg[i]) + ord(key[i])) % 26
        x += ord('A')
        cipher_text.append(chr(x))
    return("" . join(cipher_text))
```

```python
def cipherText(string, key):
    cipher_text = []
    for i in range(len(string)):
        x = (ord(string[i]) +
             ord(key[i])) % 26
        x += ord('A')
        cipher_text.append(chr(x))
    return("" . join(cipher_text))
```

```python
# расшифровка
def unvig(cipher_text, key):
    orig_text = []
    #key.replace(' ','')
    for i in range(len(cipher_text)):
        x = (ord(cipher_text[i]) - ord(key[i]) + 26) % 26
        x += ord('A')
        orig_text.append(chr(x))
    return("" . join(orig_text))
```

Рис. 4.4: Программная реализация шифрования методом таблицы Виженера

Сам процесс шифрования и расшифровки (рис. 4.5):



```
while True:
    msg = input(bold + "What message do you want to encrypt?\n" + end + "Note that only english characters and space are allowed
    if (False in [x in a1 for x in msg]):
        continue
    else:
        msg = msg.upper()
        break

while True:
    key = input(bold + "\nEnter the key\n" + end + "Note that only english characters are allowed:\n")
    if (False in [x in a1 for x in msg]):
        continue
    else:
        key = key.upper()
        break

keyg = genKey(msg,key)
print("\nYour encrypted message is: " + bold + ul + vig(msg, keyg))
```

```
What message do you want to encrypt?
Note that only english characters and space are allowed:
привет
What message do you want to encrypt?
Note that only english characters and space are allowed:
hello

Enter the key
Note that only english characters are allowed:
ura

Your encrypted message is: BVLFF
```

```
while True:
    msg = input("What message do you want to decrypt? ")
    if (False in [x in a1 for x in msg]):
        continue
    else:
        msg = msg.upper()
        break

while True:
    key = input("\nEnter the key: ")
    if (False in [x in a1 for x in msg]):
        continue
    else:
        key = key.upper()
        break

keyg = genKey(msg,key)
print("\nYour decrypted message is: " + bold + ul + unvig(msg,keyg))
```

```
What message do you want to decrypt? BVLFF

Enter the key: ura

Your decrypted message is: HELLO
```

Рис. 4.5: Шифрование и расшифровка методом таблицы Виженера

5 Выводы

Ознакомился с шифрами перестановки и обучился их программной реализации.

Список литературы

1. ТУИС: Математические основы защиты информации и информационной безопасности (02.04.02) [Электронный ресурс]. РУДН, 2022. URL: https://esystem.rudn.ru/course/view.php?id=2084.