Отчет по лабораторной работе №2

Шифры перестановки

Бурдина Ксения Павловна

23 сентября 2023

Содержание

List of Figures

1 Цель работы

Целью данной работы является освоение шифров перестановки, таких как маршрутное шифрование, шифрование с помощью решеток и таблица Виженера, а также их программная реализация.

2 Задание

1. Реализовать маршрутное шифрование.
2. Реализовать шифрование с помощью решеток.
3. Реализовать таблицу Виженера.

3 Теоретическое введение

Шифры перестановки преобразуют открытый текст в криптограмму путем перестановки его символов. Способ, каким при шифровании переставляются буквы открытого текста, и является ключом шифра. Важным требованием является равенство длин ключа и исходного текста.

В данной работе рассмотрим такие шифры перестановки, как маршрутное шифрование, шифрование с помощью решеток и таблица Виженера [1].

3.1 Маршрутное шифрование

Данный способ шифрования разработал французский математик Франсуа Виет. Открытый текст записывают в некоторую геометрическую фигуру по некоторому пути, а затем, выписывая символы по другому пути, получают шифртекст. Пусть m и n - целые положительные числа, большие единицы. Открытый текст разбивается на блоки равной длины, состоящие из числа символов, равному произведению mn. Если последний блок получится меньше остальных, то в него следует дописать требуемое количество произвольных символов. Составляется таблица размерности mn. Блоки вписываются построчно в таблицу. Криптограмма получается выписыванием букв из таблицы в соответствии с некоторым маршрутом. Ключом такой криптограммы является маршрут и числа m и n. Обычно буквы выписывают по столбцам, которые упорядочивают согласно паролю: внизу таблицы приписывается слово из n неповторяющихся букв и столбцы нумеруются по алфавитному порядку букв пароля.

Например, для шифрования текста *нельзя недооценивать противника*, разобьем его на блоки длины $n = 6$. Блоков получится $m = 5$. К последнему блоку припишем букву a. В качестве пароля выберем слово *пароль*. Теперь будем выписывать буквы по столбцам в соответствии с алфавитным порядком букв пароля и получим следующую криптограмму: ЕЕНПНЗОАТАЬОВОКННЕЬВЛДИРИЯЦТИА:

н	*е*	*л*	*ь*	*з*	*я*
н	*е*	*д*	*о*	*о*	*ц*
е	*н*	*и*	*в*	*а*	*т*
ь	*п*	*р*	*о*	*т*	*и*
в	*н*	*и*	*к*	*а*	*а*
п	*а*	*р*	*о*	*л*	*ь*

Figure 3.1: Маршрутное шифрование

3.2 Шифрование с помощью решеток

Данный способ шифрования предложил австрийский криптограф Эдуард Флейснер в 1881 году. Суть этого способа заключается в следующем: выбирается натуральное число $k > 1$, строится квадрат размерности k и построчно заполняется числами $1, 2, ..., k^2$. В качестве примера рассмотрим квадрат размерности $k = 2$. Повернем его по часовой стрелке на 90° b присоединим к исходному квадрату справа. Проделаем еще дважды такую процедуру и припишем получившиеся квдраты снизу. Получился большой квадрат размерности $2k$.

Дальше из большого квадрата вырезаются клетки, содержащие числа от 1 до k^2. В каждой клетке должно быть только одно число. Получается своего рода решето. Шифрование осуществляется следующим образом. Решето накладыва-

ется на чистый квадрат $2k*2k$ и в прорези вписываются буквы исходного текста по порядку их следования. Когда заполнятся все прорези, решето поворачивается на 90° и вписывание букв продолжается. После третьего поворота все клетки большого квадрата окажутся заполненными. Подобрав подходящий пароль, выпишем буквы по столбцам. Очередность столбцов определяется алфавитным порядком букв пароля [2].

Например, при исходном тексте *договор подписали* и пароле *шифр* с применением вышеуказанной решетки за пять шагов получим следующую криптограмму:

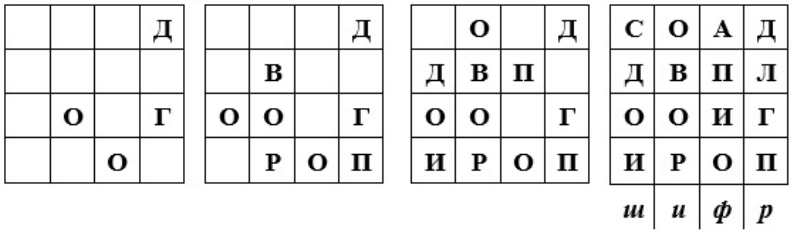

Figure 3.2: Шифрование с помощью решеток

Получившаяся криптограмма: ОВОРДЛГПАПИОСДОИ. Важно отметить, что число k подбирается в соответствии с количеством букв N исходного текста. В идеальном случае $k^2 = N$. Если такого равенства достичь невозможно, то можно либо дописать произвольную букву к последнему слову открытого текста, либо убрать ее.

3.3 Таблица Виженера

В 1585 году французский криптограф Блез Виженер опубликовал свой метод шифрования в “Трактате о шифрах”. Шифр считался нераскрываемым до 1863 года, когда австриец Фридрих Казиски взломал его.

Открытый текст разбивается на блоки длины n. Ключ представляет собой последовательность из n натуральных чисел: $a_1, a_2, ..., a_n$. Далее в каждом блоке

первая буква циклически сдвигается вправо по алфавиту на a_1 позиций, вторая буква - на a_2 позиций, последняя - на a_n позиций. Для лучшего запоминания в качестве ключа можно взять осмысленное слово, а алфавитные номера входящих в него букв использовать для осуществления сдвигов. Рассмотрим еще одну схему построения шифра Виженера. В таблицу в строки записываются буквы русского алфавита. При переходе от одной строке к другой происходит циклический сдвиг на одну позицию. Исходный текст: *криптография серьезная наука*; пароль - *математика*. Пароль записывается с повторениями над буквами сообщения:

м	а	т	е	м	а	т	и	к	а	м	а	т	е	м	а	т	и	к	а	м	а	т	е	м	а
к	р	и	п	т	о	г	р	а	ф	и	я	с	е	р	ь	е	з	н	а	я	н	а	у	к	а

Figure 3.3: Таблица Виженера

В горизонтальном алфавите в таблице находится буква *к*, а в вертикальном - буква *м*. На пересечении столбца и строки в таблице расположена буква *ц* [3]. Далее переходим к буквам *р* и *а* соответственно. В итоге получается следующая криптограмма: ЦРЬФЯОХШКФФЯДКЭЬЧПЧАЛНТШЦА.

4 Ход выполнения лабораторной работы

Для реализации шифров перестановки будем использовать среду JupyterLab. Выполним необходимую задачу.

1. Реализация маршрутного шифрования.

1.1. Зададим алфавит для дальнейшей работы с шифрами перестановки:

```
rus = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'
```

Figure 4.1: Алфавит для реализации шифров

1.2. Пропишем функцию, в которой запишем принцип формирования и работы метода маршрутного шифра, а также произведем вывод полученного зашифрованного текста по примеру на экран:

```python
# маршрутное шифрование
def mar(text, key, m, n):
    global rus
    textws = text.replace(' ', '')
    if len(textws)<m*n:
        textws += rus[:m*n-len(textws)]
    t = iter(textws)
    matrix = [[next(t) for y in range(m)] for x in range(n)]
    ps = [rus.index(x) for x in key]
    pss = sorted(ps)
    output = ''
    for letter in pss:
        for x in range(n):
            output += matrix[x][ps.index(letter)]
    return output
```

```python
print(mar('нельзя недооценивать противника', 'пароль', 6, 5))
```

```
еенпнзоатаьовокннеьвлдирияцтиа
```

Figure 4.2: Реализация маршрутного шифрования

Здесь мы подаем на ввод исходный текст, пароль, а также размерность матрицы, с которой работаем. Далее формируем текст в виде матрицы, а также записываем пароль под ней. После этого сортируем столбцы по алфавиту букв пароля, после чего выводим зашифрованный текст по полученным столбцам.

2. Реализация шифрования с помощью решеток.

2.1. Подключим для работы программы необходимые библиотеки. Зададим размерность используемой решетки, создадим матрицу для работы с зашифровкой вводимого текста, а также укажем параметры, по которым происходит заполнение решетки символами:

```
import numpy as np

# шифрование с помощью решеток
k = 2
k2 = [x+1 for x in range(k**2)]
matrix = [[0 for x in range(2*k)] for y in range(2*k)]
matrix = np.array(matrix)
for x in range(k**2):
    c = 0
    for x in range(k):
        for y in range(k):
            matrix[x][y] = k2[c]
            c += 1
    matrix = np.rot90(matrix)
ds = {k: 0 for k in k2}
dss = {1:2, 2:4, 3:3, 4:3}
for x in range(k**2):
    for y in range(k**2):
        ds[matrix[x][y]] += 1
        if ds[matrix[x][y]] != dss[matrix[x][y]]:
            matrix[x][y] = -1
        else:
            matrix[x][y] = 0
```

Figure 4.3: Реализация шифрования с помощью решеток

2.2. Введем данные, с которыми будем работать. Пропишем метод вывода зашированного сообщения с помощью использования разворота матрицы с символами и пароля, который задается для шифровки. После этого выводим получившийся шифр на экран:

```
text = 'договорподписали'
key = 'шифр'
ct = 0
t = iter(text)
matrixt = [['0' for y in range(k**2)] for x in range(k**2)]
for d in range(4):
    for x in range(k**2):
        for y in range(k**2):
            if matrix[x][y] == 0:
                matrixt[x][y] = text[ct]
                ct += 1
    matrix = np.rot90(matrix, -1)
ps = [rus.index(x) for x in key]
pss = sorted(ps)
output = ''
for letter in pss:
    for x in range(k**2):
        output += matrixt[x][ps.index(letter)]
print(output)
```

```
оворддгпаписдои
```

Figure 4.4: Вывод зашифрованного сообщения по шифрованию с помощью решеток

3. Реализация таблицы Виженера.

3.1. Пропишем функции для реализации данного метода. В первой будет описан метод задания пароля для работы с текстом, то есть каким образом пароль накладывается на исходное сообщение. Во второй функции пропишем способ нахождения зашифрованного текста по исходному с помощью пересечения букв алфавита по таблице Виженера:

```python
# таблица Виженера
def key_k(m, key):
    key.replace(' ', '')
    m.replace(' ', '')
    key = list(key)
    if len(m) == len(key):
        return(key)
    else:
        for i in range(len(m) - len(key)):
            key.append(key[i%len(key)])
    return(''.join(key))

def vig(m, key):
    ct = []
    m.replace(' ', '')
    for i in range(len(m)):
        x = (ord(m[i]) + ord(key[i])) % 32
        x += ord('A')
        ct.append(chr(x))
    return(''.join(ct))
```

Figure 4.5: Функции для реализации таблицы Виженера

По итогу при вызове функции получим зашифрованное сообщение:

```python
m = 'криптографиясерьезнаянаука'
key = 'математика'
print(vig(m, key_k(m, key)))
```

```
ЦРЪФЮОХШКФФЯГКЬЬЧПЧАЛНТШЦА
```

Figure 4.6: Вывод шифра по таблице Виженера

3.2. Далее определим функцию, которая будет обратно собирать исходное сообщение. Для этого так же через таблицу алфавита находим исходные символы по пересечению. В результате при вызове функции получаем наше исходное сообщение:

```python
def unvig(ct, key):
    ot = []
    for i in range(len(ct)):
        x = (ord(ct[i]) - ord(key[i]) + 32) % 32
        x += ord('а')
        ot.append(chr(x))
    return(''.join(ot))
```

```python
ct = 'ЦРЪФЮОХШКФФЯГКЬЬЧПЧАЛНТШЦА'
key = 'математика'
print(unvig(ct, key_k(ct, key)))
```

```
криптографиясерьезнаянаука
```

Figure 4.7: Расшифровка сообщения по таблице Виженера

5 Листинг программы

```
rus = 'абвгдежзийклмнопрстуфхцчшщъыьэюя'
# маршрутное шифрование
def mar(text, key, m, n):
  global rus
  textws = text.replace(' ', '')
  if len(textws)<m*n:
    textws += rus[:m*n-len(textws)]
  t = iter(textws)
  matrix = [[next(t) for y in range(m)] for x in range(n)]
  ps = [rus.index(x) for x in key]
  pss = sorted(ps)
  output = ''
  for letter in pss:
    for x in range(n):
      output += matrix[x][ps.index(letter)]
  return output

print(mar('нельзя недооценивать противника', 'пароль', 6, 5))

import numpy as np

# шифрование с помощью решеток
```

```
k = 2
k2 = [x+1 for x in range(k**2)]
matrix = [[0 for x in range(2*k)] for y in range(2*k)]
matrix = np.array(matrix)
for x in range(k**2):
  c = 0
  for x in range(k):
    for y in range(k):
      matrix[x][y] = k2[c]
      c += 1
  matrix = np.rot90(matrix)
ds = {k: 0 for k in k2}
dss = {1:2, 2:4, 3:3, 4:3}
for x in range(k**2):
  for y in range(k**2):
    ds[matrix[x][y]] += 1
    if ds[matrix[x][y]] != dss[matrix[x][y]]:
      matrix[x][y] = -1
    else:
      matrix[x][y] = 0

text = 'договорподписали'
key = 'шифр'
ct = 0
t = iter(text)
matrixt = [['O' for y in range(k**2)] for x in range(k**2)]
for d in range(4):
  for x in range(k**2):
    for y in range(k**2):
```

```
      if matrix[x][y] == 0:
        matrixt[x][y] = text[ct]
        ct += 1
  matrix = np.rot90(matrix, -1)
ps = [rus.index(x) for x in key]
pss = sorted(ps)
output = ''
for letter in pss:
  for x in range(k**2):
    output += matrixt[x][ps.index(letter)]
print(output)

# таблица Виженера
def key_k(m, key):
  key.replace(' ', '')
  m.replace(' ', '')
  key = list(key)
  if len(m) == len(key):
    return(key)
  else:
    for i in range(len(m) - len(key)):
      key.append(key[i%len(key)])
  return(''.join(key))

def vig(m, key):
  ct = []
  m.replace(' ', '')
  for i in range(len(m)):
    x = (ord(m[i]) + ord(key[i])) % 32
```

```
    x += ord('A')
    ct.append(chr(x))
  return(''.join(ct))

m = 'криптографиясерьезнаянаука'
key = 'математика'
print(vig(m, key_k(m, key)))

def unvig(ct, key):
  ot = []
  for i in range(len(ct)):
    x = (ord(ct[i]) - ord(key[i]) + 32) % 32
    x += ord('a')
    ot.append(chr(x))
  return(''.join(ot))

ct = 'ЦРЪФЮОХШКФФЯГКЬЬЧПЧАЛНТШЦА'
key = 'математика'
print(unvig(ct, key_k(ct, key)))
```

6 Выводы

В ходе работы мы изучили и реализовали шифры перестановки, такие как маршрутное шифрование, шифрование с помощью решеток и таблица Виженера.

7 Список литературы

1. Традиционные шифры с симметричным ключом [1]
2. Фороузан Б. А. Криптография и безопасность сетей. - М.: Интернет-Университет Информационных Технологий : БИНОМ. Лаборатория знаний, 2010. - 784 с. [2]
3. Методические материалы курса [3]