

Элементы криптографического анализа

Автор курса: Тимонина Елена Евгеньевна
Составитель: Смирнов Дмитрий Константинович

Версия от 22:12, 2 апреля 2022 г.

Оглавление

1	Домашние задания	1
1.1	Введение	1
1.2	Определение шифра. Простейшие примеры.	1
1.3	Стойкость шифров. Метод полного перебора.	3
1.4	Аналитический метод криптоанализа.	6
1.5	Перекрытия гаммы. Криптоанализ при неравновероятной гамме.	10
1.6	Методы "встреча посередине" и "разделяй и властвуй". . .	11
2	Контрольные работы	13
2.1	Шифры перестановки.	13
2.2	Корреляционный анализ.	20

Часть 1

Домашние задания

1.1 Введение

1.2 Определение шифра. Простейшие примеры.

Задача 2.1 Что такое подстановка?

Решение. Подстановка — это взаимно однозначная функция, которая переводит буквы алфавита в буквы того же самого алфавита.

Задача 2.2 Что такое группа, и почему множество S_m из примера 2.1 образует группу?

Решение. Множество $G \neq \emptyset$ с бинарной операцией " \circ ", называется *группой*, если выполнены условия:

1. $\forall a, b \in G \quad a \circ b \in G$;
2. $\forall a, b, c \in G \quad a \circ (b \circ c) = (a \circ b) \circ c$;
3. $\exists e \in G: \forall a \in G \quad e \circ a = a \circ e = a$;
4. $\forall a \in G \quad \exists b \in G: a \circ b = b \circ a = e$

Множество S_m вводится как множество всех подстановок на конечном алфавите $A = \{a_1, \dots, a_m\}$. Проверим выполнение аксиом группы:

1. Подстановка $k \in S_m$ — отображение $k: A \rightarrow A$. $\forall k_1, k_2 \in S_m$ рассмотрим суперпозицию $k_1 \circ k_2$. Так как $k_1 \circ k_2: A \rightarrow A \rightarrow A$, то $k_1 \circ k_2 \in S_m$ и первая аксиома верна.

2. $\forall k_1, k_2, k_3 \in S_m \quad k_1 \circ (k_2 \circ k_3) = k_1 \circ k_2(k_3(a)) = k_1(k_2(k_3(a))) = k_1(k_2(a)) \circ k_3(a) = (k_1 \circ k_2) \circ k_3$.

3. Поскольку S_m — множество всех подстановок, то найдётся тождественная подстановка: $\exists e \in S_m: \forall a \in A \quad e(a) = a$. Тогда $\forall k \in S_m$ верно

$$e \circ k = e(k(a)) = k(a) = k(e(a)) = k \circ e.$$

4. Так как подстановка – взаимно однозначная функция, то $\forall k \in S_m$ существует обратная функция: $\exists k^{-1}: A \rightarrow A \Rightarrow k^{-1} \in S_m$, для которой будет выполнено равенство $k \circ k^{-1} = k(k^{-1}(a)) = k^{-1}(k(a)) = k^{-1} \circ k$. При этом, $\forall a \in A \quad k^{-1}(k(a)) = a = e(a)$.

Выполнены все аксиомы группы, следовательно S_m – группа.

Задача 2.3 Почему группа S_n из примера 2.2 является симметрической?

Решение. Симметрической группой n -го порядка называется множество $S(X)$ всех биективных отображений $f: X \rightarrow X$, где X – конечное множество из n элементов. Группа S_n в примере 2.2 определяется как группа подстановок на множестве $X = \{1, \dots, n\}$. Подстановка – это биективное отображение, X – конечное множество из n элементов. Следовательно, по определению, группа S_n является симметрической.

Задача 2.4 Что такое кольцо? Что такое кольцо вычетов по модулю m ?

Решение. Множество K называется *кольцом*, если в K определены две операции “+” (сложение) и “·” (умножение) и выполняются следующие условия $\forall a, b, c \in K$:

1. $a + b \in K, a \cdot b \in K$;
2. $a + (b + c) = (a + b) + c, a(bc) = (ab)c$;
3. $a + b = b + a$;
4. $(a + b)c = ac + bc$;
5. $\exists 0 \in K: a + 0 = a$.

Кольцом вычетов по модулю m называется такое кольцо

$\mathbb{Z}/m = \{C_0, C_1, \dots, C_{m-1}\}$ (C_r – смежный класс вычетов по модулю m), в котором операции сложения и умножения определяются следующими правилами:

1. $C_a + C_b = C_r$, где $r \equiv (a + b) \pmod{m}$;
2. $C_a C_b = C_r$, где $r \equiv ab \pmod{m}$

То есть, $C_a + C_b$ – это класс, в который входит число $a + b$, а $C_a C_b$ – класс, в который входит число ab .

Задача 2.5 Какую алгебраическую структуру представляет собой кольцо \mathbb{Z}/m при $m = 2$?

Решение.

Теорема 2.1 Если p – простое число и $p \geq 2$, то \mathbb{Z}/p – поле характеристики p .

По приведённой выше теореме кольцо \mathbb{Z}_2 является полем характеристики 2.

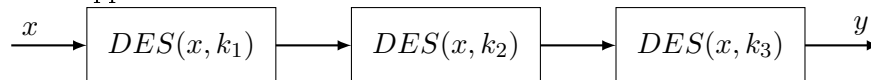
1.3 Стойкость шифров. Метод полного перебора.

Задача 3.1 Дан алфавит $A = \{1, 2, \dots, n\}$, x – открытый текст в алфавите A . Ключ шифрования (T_1, T_2, T_3) , где T_i – случайные подстановки. Алгоритм шифрования: $T_3(T_2(T_1(x))) = y$. Какова формула для расшифрования? Мощность пространства различных ключей? Сложность МПП?

Решение.

1. Формула для расшифрования – $x = T_1^{-1}(T_2^{-1}(T_3^{-1}(y)))$.
2. В каждой подстановке на первое место можно поставить n различных букв, на второе – $n - 1$, и т.д. В итоге получаем $n!$ вариантов на каждую подстановку, следовательно, $|K| = (n!)^3$ для трёх подстановок.
3. Пусть в тексте a букв. Тогда необходимо провести $3a$ операций подстановки, чтобы проверить один ключ. В среднем нужно проверить количество ключей, равное средней трудоёмкости МПП: $E_T = \frac{|K|+1}{2} = \frac{(n!)^3+1}{2}$. Следовательно, сложность МПП равна $\frac{3}{2}a[(n!)^3 + 1]$.

Задача 3.2 Найти минимальную среднюю трудоёмкость в следующей схеме шифрования:



Решение.

В предложенной схеме используется три блока DES с разными ключами. Для одного блока DES $|K| = 2^{56}$, тогда для всей схемы: $|K| = (2^{56})^3 = 2^{168}$. Окончательно, $E_T = \frac{|K|+1}{2} = \frac{2^{168}+1}{2} \approx 2^{167}$.

Задача 3.3 В сообщении каждая буква записывается два раза. Для шифрования используется шифр перестановки длины $2n$. Сложность МПП?

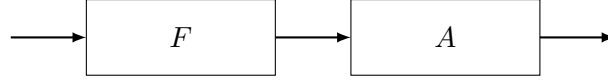
Решение.

В данной схеме используется две подстановки, причём для каждой нечётной буквы применяется первая подстановка, а для каждой чётной – вторая: $T(x) = T(x_1, x_2, \dots, x_{2l-1}, x_{2l}) = (T_1(x_1), T_2(x_2), \dots, T_1(x_{2l-1}), T_2(x_{2l}))$, где l – половина длины сообщения. Тогда длина ключа для каждой из

подстановок будет равна n , а мощность пространства различных ключей для всей системы будет равна $|K| = (n!)^2$.

Для проверки одного ключа (T_1, T_2) требуется $2l$ операций подстановки. Тогда сложность МПП равна $2lE\tau = 2l\frac{|K|+1}{2} = l[(2n)! + 1]$.

Задача 3.4



В данной схеме байт ОТ $x = x_1x_2\dots x_8$ шифруется с помощью функции F следующим образом:

$$x'_1 = x_1;$$

$$x'_2 = x_2 + f_1(x_1);$$

...

$$x'_8 = x_8 + f_8(x_1, x_2, \dots, x_7),$$

где f_1, \dots, f_7 – случайные булевы функции, A – невырожденная матрица. Ключом являются F и A . Оценить сложность нахождения ключа с помощью МПП.

Решение.

Определим мощность пространства ключей для F . Так как количество функций, зависящих от n переменных, равно 2^{2^n} , то

$$|K_F| = \prod_{i=1}^7 2^{2^i} = 2^{\sum_{i=1}^7 2^i} = 2^{\frac{2(2^7-1)}{2-1}} = 2^{2^8-2} = 2^{254}.$$

Теперь рассмотрим матрицу A . Оценим мощность пространства ключей индуктивно по строкам. Для первой строки подходит $2^n - 1$ вариантов (все, кроме нулевой строки). Для следующей строки не подойдёт предыдущий вариант заполнения (иначе будет линейная зависимость, следовательно, вырожденность матрицы) и нулевое заполнение, то есть, $2^n - 2$ вариантов. Теперь, для третьей строки нужно не допустить линейной комбинации первых двух: $\alpha a_1 + \beta a_2 \neq a_3$. Вариантов выбрать коэффициенты α и $\beta - 2^2$ (при этом, тут уже считается и нулевой случай). Далее, для четвёртой строки, аналогично, 2^3 . Таким образом, получаем формулу:

$$|K| = \prod_{i=0}^{n-1} 2^n - 2^i$$

На матрицу A мы умножаем вектор длины 8 и на выходе тоже получаем вектор длины 8. Следовательно, $n = 8$, $|K_A| \approx 2^{62.21}$

Таким образом,

$$|K| = |K_F| \cdot |K_A| \approx 2^{254} \cdot 2^{62.21} = 2^{316.21}$$

Если бы нам были известны функции f_1, \dots, f_7 , то можно было бы рассчитать количество операций на каждый ключ точно. Но нам они неизвестны, поэтому примем за общее число операций для проверки одного ключа за p . Тогда сложность МПП равна $\frac{|K|+1}{2}p \approx 2^{315.21}p$.

Комментарий к задачам о многочлене Жегалкина.

В полином Жегалкина степени не выше m от функции n переменных входит C_n^k различных мономов степени k . При этом перед каждым из них стоит коэффициент, следовательно, $2^{C_n^k}$ – количество различных вариантов выбрать 0 или 1 перед мономами.

Если полином степени ровно m , то хотя бы при одном мономе этой степени стоит коэффициент 1. Это означает, что число различных вариантов выбрать 0 или 1 перед мономами степени m в таком полиноме равно $2^{C_n^m-1}$.

Используя полином Жегалкина степени не выше m , будем считать, что $n = m$.

Задача 3.5 Ключ шифрования k – многочлен Жегалкина степени 2. Мощность пространства различных ключей? Сложность МПП?

Решение.

$$|K| = 2^{C_n^0 + C_n^1 + C_n^2 - 1} = 2^{n + \frac{(n-1)n}{2}} = 2^{\frac{n^2+n}{2}}.$$

$$\text{Количество операций } p = C_n^1(1+1) + C_n^2(1+2) = 2n + 3\frac{(n-1)n}{2} = \frac{3}{2}n^2 + \frac{1}{2}n$$

$$\text{Сложность: } pE\tau = (\frac{3}{2}n^2 + \frac{1}{2}n)2^{\frac{n^2+n}{2}+1} \approx (3n^2 + n)2^{\frac{n^2+n-4}{2}}$$

$$\text{С учётом последнего комментария получим } |K| = 8, pE\tau = 31.5.$$

Задача 3.6 Ключ шифрования k – многочлен Жегалкина степени не выше m . Мощность пространства различных ключей? Сложность МПП?

Решение.

$$|K| = 2^{\sum_{i=0}^m C_n^i}.$$

$$\text{Количество операций } p = \sum_{i=1}^m C_n^i(i+1)$$

$$\text{Сложность: } pE\tau = [\sum_{i=1}^m C_n^i(i+1)]2^{\frac{\sum_{i=0}^m C_n^i + 1}{2}} \approx [\sum_{i=1}^m C_n^i(i+1)]2^{\sum_{i=1}^m C_n^i}$$

Задача 3.7 Ключ шифрования k – многочлен вида:

$$\sum_{1 \leq i < j \leq n} a_{ij} x_i x_j, a_{ij} \in \{0, 1\}.$$

Мощность пространства различных ключей? Сложность МПП?

Решение.

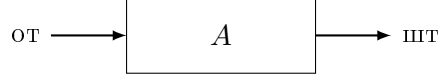
Множество a_{ij} образует верхнетреугольную матрицу без главной диагонали. Следовательно, $|K| = 2^{(n-1)+(n-2)+\dots+1+0} = 2^{\frac{(n-1)n}{2}}$.

Количество операций $p = \frac{(n-1)n}{2}(1+2) - 1 = \frac{3}{2}n^2 - \frac{3}{2}n - 1$

Сложность: $pE\tau = (\frac{3}{2}n^2 - \frac{3}{2}n - 1)^{\frac{(n-1)n}{2}+1} \approx (3n^2 - 3n - 2)2^{\frac{n^2-n-4}{2}}$

1.4 Аналитический метод криптоанализа.

Задача 4.1 Найти минимальную сложность нахождения ключа в схеме



Ключом является невырожденная двоичная матрица A размером $n \cdot n$. Сравнить со сложностью МПП.

Решение.

При решении СЛАУ методом Гаусса сложность оценивается в $\frac{n^3}{3}$ операций. Количество операций, необходимое для проверки одного ключа, равно $p = (n + (n-1)) \cdot n = 2n^2 - n$ – такое количество операций сложения и умножения нужно проделать для умножения вектора на квадратную матрицу. Было установлено, что:

$$|K| = \prod_{i=0}^{n-1} 2^n - 2^i = (2^n)^n + \dots = O(2^{n^2})$$

Следовательно, сложность МПП:

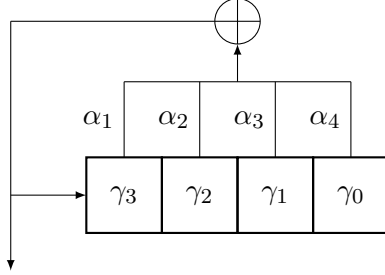
$$E\tau = p \frac{|K| + 1}{2} = (2n^2 - n) \frac{2^{n^2} + \dots}{2} = O(n^2 \cdot 2^{n^2})$$

Пусть $n = 10$, тогда для МПП потребуется порядка $10^2 \cdot 2^{10^2} \approx 10^{32.10}$ операций, тогда как для аналитического метода получится $\frac{10^3}{3} \approx 3 \cdot 10^2$ операций.

Задача 4.2 Для ЛРП, задаваемой с помощью характеристического многочлена

$F(x) = x^4 \oplus x^2 \oplus x \oplus 1$, построить ЛРС, определить матрицу A , и для выходной (после 4-х тактов работы ЛРС) последовательности $\gamma = (1, 0, 1, 0)$ найти начальное заполнение регистра.

Решение.



Из характеристической функции следует, что $\alpha_1 = 1, \alpha_2 = 1, \alpha_3 = 0, \alpha_4 = 1$.

Тогда $\gamma_4 = 1 \cdot \gamma_0 + 0 \cdot \gamma_1 + 1 \cdot \gamma_2 + 1 \cdot \gamma_3$. Значит, матрица $A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$.

Решим следующее уравнение: $A^4 \gamma^T(0) = \gamma^T$.

$$A^4 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}^2 = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 2 \\ 2 & 1 & 3 & 3 \\ 3 & 2 & 4 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\left[\begin{array}{cccc|c} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{array} \right] \sim \left[\begin{array}{cccc|c} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{array} \right] \sim \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{array} \right]$$

Следовательно, $\gamma(0) = (0, 0, 0, 1)$.

Задача 4.3 Объяснить равенства (4.11) и (4.12).

Решение.

Пусть f имеет следующую структуру:

$$f(\gamma_n, \gamma_{n+1}, \dots, \gamma_{n+r-1}) = \gamma_n \oplus g(\gamma_{n+1}, \gamma_{n+1}, \dots, \gamma_{n+r-1}).$$

Тогда:

$$f(0, x_2, \dots, x_r) \oplus f(1, x_2, \dots, x_r) = 0 \oplus g(x_2, \dots, x_r) \oplus 1 \oplus g(x_2, \dots, x_r) = 1$$

Следовательно, $f(0, x_2, \dots, x_r) = 1 \oplus f(1, x_2, \dots, x_r)$.

Равенство $f(x_1, x_2, \dots, x_r) = x_1 f(1, x_2, \dots, x_r) \oplus (1 \oplus x_1) f(0, x_2, \dots, x_r)$ проверяется непосредственной подстановкой x_1 . В самом деле, при $x_1 = 0$ первое слагаемое обращается в ноль, и имеем $f(0, x_2, \dots, x_r) = f(0, x_2, \dots, x_r)$. А при $x_1 = 1$ – второе: $f(1, x_2, \dots, x_r) = f(1, x_2, \dots, x_r)$

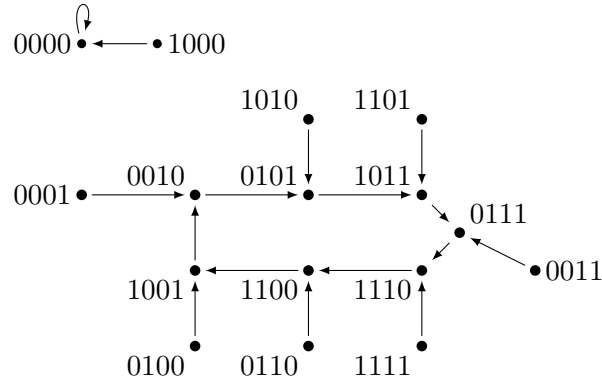
Задача 4.4 Построить графы отображений для РС, обратные связи которых задаются функциями от 4 переменных:

$$f_1 = x_2 \oplus x_3, f_2 = x_1 \oplus x_2 \oplus x_3, f_3 = x_3 \oplus x_2 * x_4, f_4 = x_1 \oplus x_3 * x_4, f_5 = x_1 * x_3 \oplus x_2 * x_4.$$

Прокомментировать результаты.

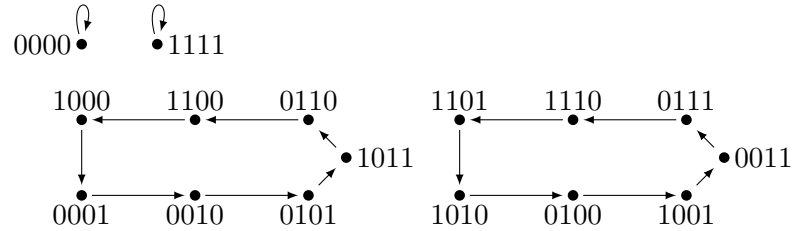
Решение.

$$\text{ЛРС } F_1 : (x_1, x_2, x_3, x_4) \rightarrow (x_2, x_3, x_4, x_2 \oplus x_3)$$



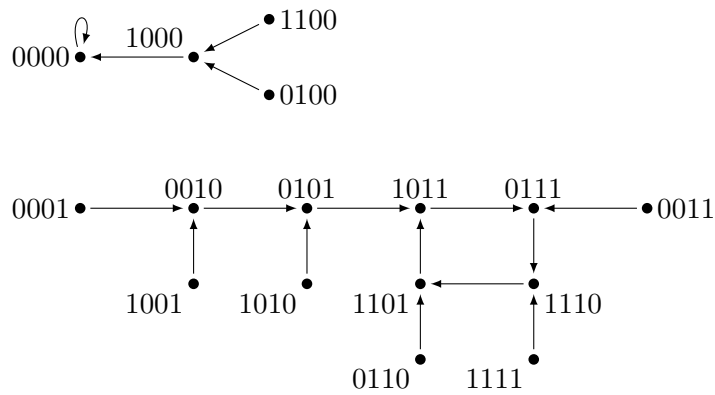
Данный граф имеет структуру "циклы с подходами". Длины циклов: 1, 7. Это отображение не является взаимно однозначным.

$$\text{ЛРС } F_2 : (x_1, x_2, x_3, x_4) \rightarrow (x_2, x_3, x_4, x_1 \oplus x_2 \oplus x_3)$$



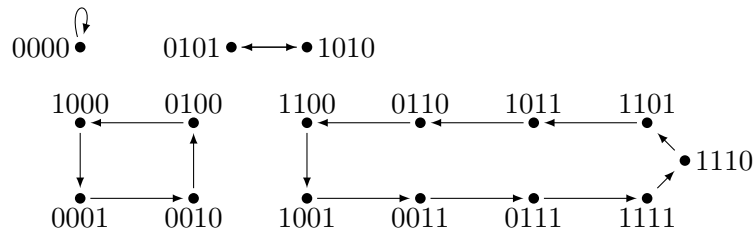
У этого графа полностью цикловая структура. Длины циклов: 1, 1, 7 и 7. Это отображение является взаимно однозначным.

$$\text{ЛРС } F_3 : (x_1, x_2, x_3, x_4) \rightarrow (x_2, x_3, x_4, x_3 \oplus x_2 * x_4)$$



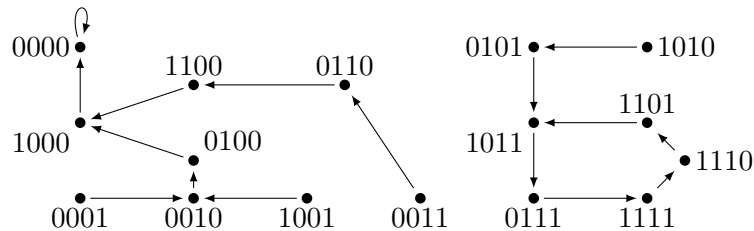
Данный граф имеет структуру "циклы с подходами". Длины циклов: 1, 4. Это отображение не является взаимно однозначным.

$$\text{ЛРС } F_4 : (x_1, x_2, x_3, x_4) \rightarrow (x_2, x_3, x_4, x_1 \oplus x_3 * x_4)$$



Граф имеет полностью цикловую структуру. Длины циклов: 1, 2, 4 и 9. Это отображение является взаимно однозначным.

$$\text{ЛРС } F_5 : (x_1, x_2, x_3, x_4) \rightarrow (x_2, x_3, x_4, x_1 * x_3 \oplus x_2 * x_4)$$



Данный граф имеет структуру "циклы с подходами". Длины циклов: 1, 5. Это отображение не является взаимно однозначным.

1.5 Перекрытия гаммы. Криптоанализ при неравновероятной гамме.

Задача 5.1 Два текста x и x' на русском языке зашифрованы шифром гаммирования по $\bmod 30$ с помощью одной и той же гаммы γ . Использована следующая таблица соответствия букв числами (здесь – означает пробел):

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	Э	Ю	Я	–
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

Получено два шифротекста $y = \text{КЛОВБЛЖЗФ}$ и $y' = \text{ВУПЗЕРСВЖ}$, известна тематика x и x' : 'времена года'. Применяя 'протяжку вероятного слова' найти x, x', γ .

Решение.

Переведём векторы y и y' в числа и найдём их разность:
 $y - y' = x + \gamma - x' - \gamma = x - x' = (9 - 2, 10 - 18, 13 - 14, 2 - 7, 1 - 5, 10 - 15, 6 - 16, 7 - 2, 19 - 6) = (7, 22, 29, 25, 26, 25, 20, 5, 13) = \text{ЗЧ-ЫЭЫЧЕО}.$

Попробуем подставить в начало x' слово 'ЗИМА-':
 $x = (x - x') + x' = \text{АСНЕГ * * * *}$

Видно, что получается осмысленное предложение. Посмотрим, какая гамма:

$$\gamma = y' - x' = \text{ВВВВВ * * * *}$$

Предположим, что гамма состоит только из этих букв, продлим и получим окончательный ответ:

$$x = \text{ЗИМА – ИДЕТ}$$

$$x' = \text{АСНЕГОПАД}$$

$$\gamma = \text{ВВВВВВВВВ}$$

Задача 5.2 Пусть в шифре гаммирования по $\bmod 30$ используется только 6 знаков гаммы $\{17, 05, 02, 15, 08, 14\}$ (соответствие букв и чисел в таблице):

А	Б	В	Г	Д	Е	Ж	З	И	К	Л	М	Н	О	П
00	01	02	03	04	05	06	07	08	09	10	11	12	13	14
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	Ъ	Э	Ю	Я
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

Получен шифртекст $y = \text{ШАССЧАТАИЦОС}$. Используя "зигзагообразное" чтение дешифровать открытый текст и восстановить гамму.

Решение.

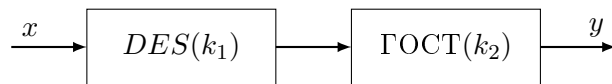
Составим таблицу из возможных результатов гаммирования:

17	Ж	О	Я	Я	Е	О	А	О	Ц	Д	Ъ	Я
05	У	Ы	М	М	Т	Ы	Н	Ы	Г	С	И	М
02	Ц	Ю	П	П	Х	Ю	Р	Ю	Ж	Ф	М	П
15	И	Р	Б	Б	З	Р	В	Р	Ш	Ж	Ю	Б
08	Р	Ч	И	И	П	Ч	К	Ч	А	О	Е	И
14	К	С	В	В	И	С	Г	С	Щ	З	Я	В

Легко видеть, $x = \text{КРИПТОГРАФИЯ}$, $\gamma = \text{ПРИВЕТПРИВЕТ}$.

1.6 Методы "встреча посередине" и "разделяй и властвуй".

Задача 6.1 Найти минимальную среднюю трудоёмкость нахождения ключа в следующей схеме шифрования, длина ключа ГОСТ = 256 бит. Сравнить с МПП.



Решение.

Средняя трудоёмкость метода "встречи посередине":

$$(|K_1| + |K_2|)(1 + \ln(|K_1| + |K_2|)) = (2^{56} + 2^{256})(1 + \ln(2^{56} + 2^{256})) \approx 10^{79.47}$$

Средняя трудоёмкость полного перебора: $\frac{|K_1||K_2|}{2} = \frac{2^{56} \cdot 2^{256}}{2} = 2^{311} \approx 10^{93.62}$

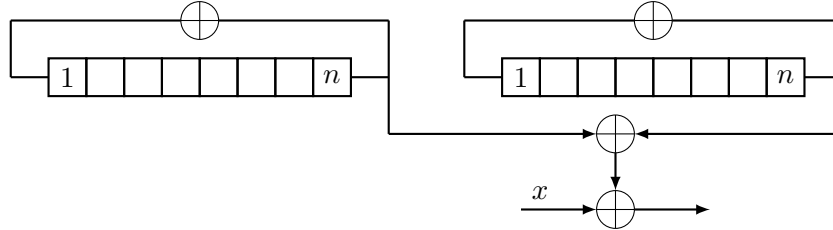
Метод "встречи посередине" оказывается на 14 порядков эффективнее МПП.

Если предположить, что у нас имеется эффективный критерий, отбраковывающий ключи из K_1 , то можно воспользоваться методом "разделяй и властвуй", средняя трудоёмкость которого равна $\frac{|K_1| + |K_1|}{2} = 2^{55} + 2^{255} \approx 10^{76.76}$. Этот метод ещё эффективнее в 1000 раз.

Задача 6.2 Ключом являются начальные заполнения ЛРС в алгоритме получения γ для шифра гаммирования. Предполагается, что имеется

12 1.6 Методы "встреча посередине" и "разделяй и властвуй".

необходимое количество пар (x, y) . Оценить сложность нахождения ключа с помощью метода "встречи посередине" и сравнить с МПП.



Решение.

Для каждого ЛРС оценим мощность множеств ключей: $N = |K_1| = |K_2| = 2^n$. Тогда средняя трудоёмкость метода "встречи посередине":

$$\sqrt{N} \ln N = 2^{\frac{n}{2}} \ln 2^n$$

Средняя трудоёмкость полного перебора:

$$\frac{|K_1||K_2|}{2} = \frac{2^n \cdot 2^n}{2} = 2^{2n-1}$$

При $n = 8$ метод "встречи посередине" эффективнее, чем МПП, в 369 раз, а при $n = 256$ – примерно в 10^{113} раз.

Задача 6.3 В задаче 3.4 найти минимальную среднюю трудоёмкость нахождения ключа и сравнить с МПП. Предполагается, что имеется необходимое количество пар (x, y) .

Решение.

Было установлено, что $|K_F| = 2^{254}$ и $|K_A| \approx 2^{62.21}$.

Метод "разделяй и властвуй": $\frac{|K_F| + |K_A|}{2} \approx 10^{76.16}$.

Метод "встречи посередине": $(|K_A| + |K_F|)(1 + \ln(|K_A| + |K_F|)) \approx 10^{78.71}$.

МПП: $\frac{|K_F||K_A|}{2} \approx 10^{95.19}$.

Если предположить, что у нас есть эффективный критерий, отбраковывающий ключи из K_F , то минимальная средняя трудоёмкость достигается первым методом, иначе – вторым. Разница по эффективности с МПП от $10^{16.48}$ до $10^{19.03}$ раз.

Часть 2

Контрольные работы

2.1 Шифры перестановки.

Задача 1.1 Раскрыть шифр простой замены:

56 73 31 68 52 88 52 70 16 78 16 90 40 49 16 31 78 56 46 28 88 31 40 88 70
68 52 40 19 56 70 73 88 19 94 00 52 31 49 68 78 88 56 90 73 16 31 49 94 88
88 46 36 49 88 52 88 46 68 74 49 16 78 64 94 88 52 40 68 19 94 16 03 20 49
64 46 88 78 64 13 16 90 40 49 03 16 52 31 78 16 70 88 73 68 78 88 90 40 49
20 94 56 66 46 00 88 49 40 68 78 88 73 31 74 87 88 16 83 16 78 68 94 56 16
16 52 20 90 68 73 56 70 88 73 68 49 64 49 03 87 56 94 16 73 16 31 16 78 56
78 56 31 64 46 00 88 94 56 40 88 40 88 73 88 70 20 16 28 88 73 16 03 94 00
66 94 16 70 88 19 68 90 20 52 16 94 56 82 31 83 16 94 11 56 94 68 52 56 90
40 49 90 94 68 74 90 40 49 03 49 88 31 78 68 73 88 82 70 68 52 31 87 88 28
88 20 28 88 70 94 56 87 68 83 68 87 88 46 74 90 68 94 46 88 74 90 94 56 31
40 68 49 64 73 88 70 56 94 88 03 16 31 49 73 16 90 40 49 68 94 16 40 19 56
19 88 70 94 88 82 88 90 68 46 88 03 16 94 94 88 31 49 56 49 03 87 68 31 94
16 70 68 73 94 56 66 40 88 19 13 20 49 56 73 88 73 31 16 31 49 19 68 13 56
78 31 74 90 68 31 00 40 68 49 64 56 90 90 68 40 88 31 49 88 74 94 94 00 66
87 88 13 52 68 19 88 73 49 03 87 66 88 19 88 13 88 16 11 16 90 40 49 03 49
88 88 94 40 88 94 68 49 20 19 16 03 16 78 88 73 16 87 78 16 28 87 88 52 00
31 78 16 94 94 00 82 56 94 16 31 40 88 31 88 46 94 00 82 90 68 97 56 87 78
56 73 68 49 64 31 74 94 68 03 16 52 78 56 46 88 90 40 49 56 94 68 03 16 52
88 83 94 88 56 20 52 88 52 49 19 88 94 20 49 64 31 74

Решение.

Для более простого воспроизведения описанных действий буду приводить код на языке Python.

Проанализируем частоты монограмм.

```
>>> sorted(zip(*np.unique(cipher, return_counts = True)), key =
          lambda x: x[1], reverse = True)[:10]
[('88', 58), ('16', 37), ('94', 36), ('68', 33), ('49', 31), ('56',
  29), ('31', 26), ('40', 21), ('73', 19), ('90', 19)]
```

Теперь рассмотрим биграммы:

```
>>> bigram = np.array([cipher[i] + ' ' + cipher[i+1] for i in
          range(len(cipher) - 1)])
>>> sorted(zip(*np.unique(bigram, return_counts = True)), key =
          lambda x: x[1], reverse = True)[:10]
[('40 49', 8), ('88 73', 8), ('90 40', 8), ('40 88', 7), ('94 56',
  7), ('03 16', 6), ('16 31', 6), ('31 49', 6), ('49 03', 6),
  ('49 64', 6)]
```

Наиболее частые моно- и биграммы русского языка:

О	Е	А	И	Н	Т	С	Р	В	Л
---	---	---	---	---	---	---	---	---	---

СТ	НО	ЕН	ТО	НА	ОВ	НИ	РА	ВО	КО
----	----	----	----	----	----	----	----	----	----

Предположим, что 88 – это О. В биграммах из текста эта буква встречается дважды: 88 73 и 40 88. В справочной таблице единственное сочетание, в котором О стоит на первом месте – это ОВ. Сравнивая позицию буквы 73 с первой таблицей, можем убедиться, что В действительно подходит.

Допустим также, что 16 – это Е. Поскольку в шифротексте нет явных знаков препинания, предположим, что они записаны в виде ЗПТ и ТЧК. Запятых, скорее всего, больше, чем точек, поэтому рассмотрим триграммы текста и самую частую определим как ЗПТ.

```
>>> trigram = np.array([cipher[i] + ' ' + cipher[i+1] + ' ' +
          cipher[i+2] for i in range(len(cipher) - 2)])
>>> sorted(zip(*np.unique(trigram, return_counts = True)), key =
          lambda x: x[1], reverse = True)[:5]
[('90 40 49', 8), ('16 90 40', 4), ('68 49 64', 4), ('03 16 52',
  3), ('16 31 49', 3)]
```

Тогда 49 – это Т. Попробуем найти среди биграмм наиболее частую – СТ: единственный вариант, заканчивающийся на 49, – это 31 49 (40 49 уже занято – ПТ). Пусть 31 будет С.

Итак, попробуем подставить:

О	В	Е	З	П	Т	С
88	73	16	90	40	49	31

```
>>> key = {'88': 'О', '73': 'В', '16': 'Е', '90': 'З', '40': 'П',
          '49': 'Т', '31': 'С'}
>>> ' '.join([key[x] if x in key else x for x in cipher])
'56 В С 68 52 0 52 70 Е 78 Е З П Т Е С 78 56 46 28 0 С П 0 70 68 52
П 19 56 70 В 0 19 94 00 52 С Т 68 78 0 56 З В Е С Т 94 0 0 46
36 Т 0 52 0 46 68 74 Т Е 78 64 94 0 52 П 68 19 94 Е 03 20 Т 64
46 0 78 64 13 Е З П Т 03 Е 52 С 78 Е 70 0 В 68 78 0 З П Т 20 94
56 66 46 00 0 Т П 68 78 0 В С 74 87 0 Е 83 Е 78 68 94 56 Е Е 52
20 З 68 В 56 70 0 В 68 Т 64 Т 03 87 56 94 Е В Е С Е 78 56 78 56
С 64 46 00 0 94 56 П 0 П 0 В 0 70 20 Е 28 0 В Е 03 94 00 66 94
Е 70 0 19 68 З 20 52 Е 94 56 82 С 83 Е 94 11 56 94 68 52 56 З П
Т З 94 68 74 З П Т 03 Т 0 С 78 68 В 0 82 70 68 52 С 87 0 28 0
20 28 0 70 94 56 87 68 83 68 87 0 46 74 З 68 94 46 0 74 З 94 56
С П 68 Т 64 В 0 70 56 94 0 03 Е С Т В Е З П Т 68 94 Е П 19 56
19 0 70 94 0 82 0 З 68 46 0 03 Е 94 94 0 С Т 56 Т 03 87 68 С 94
Е 70 68 В 94 56 66 П 0 19 13 20 Т 56 В 0 В С Е С Т 19 68 13 56
78 С 74 З 68 С 00 П 68 Т 64 56 З З 68 П 0 С Т 0 74 94 94 00 66
87 0 13 52 68 19 0 В Т 03 87 66 0 19 0 13 0 Е 11 Е З П Т 03 Т 0
0 94 П 0 94 68 Т 20 19 Е 03 Е 78 0 В Е 87 78 Е 28 87 0 52 00 С
78 Е 94 94 00 82 56 94 Е С П 0 С 0 46 94 00 82 З 68 97 56 87 78
56 В 68 Т 64 С 74 94 68 03 Е 52 78 56 46 0 З П Т 56 94 68 03 Е
52 0 83 94 0 56 20 52 0 52 Т 19 0 94 20 Т 64 С 74'
```

Обратим внимание на 'ЗПТЕС 78 56', 'ПОПОВО 70 20', 'ПОСТО 74 94 94 *', 'СПОСО 46'. Всё это похоже на ', если', 'по поводу', 'постоянн*' и 'способ'. Попробуем добавить в ключ следующие замены:

Л	И	Д	У	Я	Н	Б
78	56	70	20	74	94	46

```
>>> key.update({'78': 'Л', '56': 'И', '70': 'Д', '20': 'У', '74':
              'Я', '94': 'Н', '46': 'Б'})
>>> ' '.join([key[x] if x in key else x for x in cipher])
'И В С 68 52 0 52 Д Е Л Е З П Т Е С Л И Б 28 0 С П 0 Д 68 52 П 19 И
Д В 0 19 Н 00 52 С Т 68 Л О И З В Е С Т Н О О Б 36 Т 0 52 О Б
68 Я Т Е Л 64 Н 0 52 П 68 19 Н Е 03 У Т 64 Б О Л 64 13 Е З П Т
03 Е 52 С Л Е Д О В 68 Л О З П Т У Н И 66 Б 00 О Т П 68 Л О В С
Я 87 О Е 83 Е Л 68 Н И Е Е 52 У З 68 В И Д О В 68 Т 64 Т 03 87
И Н Е В Е С Е Л И Л И С 64 Б 00 О Н И П О П О В О Д У Е 28 О В
Е 03 Н 00 66 Н Е Д 0 19 68 З У 52 Е Н И 82 С 83 Е Н 11 И Н 68
52 И З П Т З Н 68 Я З П Т 03 Т О С Л 68 В 0 82 Д 68 52 С 87 0
28 0 У 28 0 Д Н И 87 68 83 68 87 О Б Я З 68 Н Б О Я З Н И С П
```

68 Т 64 В О Д И Н О 03 Е С Т В Е З П Т 68 Н Е П 19 И 19 О Д Н О
 82 О З 68 Б О 03 Е Н Н О С Т И Т 03 87 68 С Н Е Д 68 В Н И 66 П
 О 19 13 У Т И В О В С Е С Т 19 68 13 И Л С Я З 68 С 00 П 68 Т
 64 И З З 68 П О С Т О Я Н Н 00 66 87 О 13 52 68 19 О В Т 03 87
 66 О 19 О 13 О Е 11 Е З П Т 03 Т О О Н П О Н 68 Т У 19 Е 03 Е Л
 О В Е 87 Л Е 28 87 О 52 00 С Л Е Н Н 00 82 И Н Е С П О С О Б Н
 00 82 З 68 97 И 87 Л И В 68 Т 64 С Я Н 68 03 Е 52 Л И Б О З П Т
 И Н 68 03 Е 52 О 83 Н О И У 52 О 52 Т 19 О Н У Т 64 С Я'

Видно, что 'С Т 68 Л О И З В Е С Т Н О О Б 36 Т О 52 О Б 68 Я Т Е
 Л 64 Н О 52 П 68 19 Н Е' похоже на 'стало известно об этом обаятельном
 парне', а 'В Е С Е Л И Л И С 64 Б 00 О Н И П О П О В О Д У Е 28 О'
 – на 'веселились бы они по поводу его', 'В О Д И Н О 03 Е С Т В Е' –
 'в одиночестве'

А	Э	М	Ь	Р	Ы	Г	Ч
68	36	52	64	19	00	28	03

```
>>> key.update(**{'68': 'А', '36': 'Э', '52': 'М', '64': 'Ь', '19':  
    'Р', '00': 'Ы', '28': 'Г', '03': 'Ч'})  
>>> ' '.join([key[x] if x in key else x for x in cipher])  
'И В С А М О М Д Е Л Е З П Т Е С Л И Б Г О С П О Д А М П Р И Д В О  
    Р Н Ы М С Т А Л О И З В Е С Т Н О О Б Э Т О М О Б А Я Т Е Л Ь Н  
    О М П А Р Н Е Ч У Т Ь Б О Л Ь 13 Е З П Т Ч Е М С Л Е Д О В А Л  
    О З П Т У Н И 66 Б Ы О Т П А Л О В С Я 87 О Е 83 Е Л А Н И Е Е  
    М У З А В И Д О В А Т Ь Т Ч 87 И Н Е В Е С Е Л И Л И С Ь Б Ы О  
    Н И П О П О В О Д У Е Г О В Е Ч Н Ы 66 Н Е Д О Р А З У М Е Н И  
    82 С 83 Е Н 11 И Н А М И З П Т З Н А Я З П Т Ч Т О С Л А В О 82  
    Д А М С 87 О Г О У Г О Д Н И 87 А 83 А 87 О Б Я З А Н Б О Я З Н  
    И С П А Т Ь В О Д И Н О Ч Е С Т В Е З П Т А Н Е П Р И Р О Д Н О  
    82 О З А Б О Ч Е Н Н О С Т И Т Ч 87 А С Н Е Д А В Н И 66 П О Р  
    13 У Т И В О В С Е С Т Р А 13 И Л С Я З А С Ы П А Т Ь И З З А П  
    О С Т О Я Н Н Ы 66 87 О 13 М А Р О В Т Ч 87 66 О Р О 13 О Е 11  
    Е З П Т Ч Т О О Н П О Н А Т У Р Е Ч Е Л О В Е 87 Л Е Г 87 О М Ы  
    С Л Е Н Н Ы 82 И Н Е С П О С О Б Н Ы 82 З А 97 И 87 Л И В А Т Ь  
    С Я Н А Ч Е М Л И Б О З П Т И Н А Ч Е М О 83 Н О И У М О М Т Р  
    О Н У Т Ь С Я'
```

'ЧУТЬБОЛЬ 13 ЕЗПТ' – 'чуть больше', 'УНИ 66 БЫ' – 'у них бы',
 'В С Я 87 О Е 83 Е Л А Н И Е' – 'всякое желание', 'Н Е Д О Р А З У
 М Е Н И 82 С 83 Е Н 11 И Н А М И' – 'недоразумений с женщинами',
 'З А 97 И 87 Л И В А Т Ь С Я' – 'зацикливаться'.

Ш	Х	К	Ж	Й	Щ	Ц
13	66	87	83	82	11	97

```
>>> key.update(**{'13': 'Ш', '66': 'Х', '87': 'К', '83': 'Ж', '82':
    'Й', '11': 'Щ', '97': 'Ц'})
>>> ' '.join([key[x] if x in key else x for x in cipher])
'И В С А М О М Д Е Л Е З П Т Е С Л И Б Г О С П О Д А М П Р И Д В О
Р Н Ы М С Т А Л О И З В Е С Т Н О О Б Э Т О М О Б А Я Т Е Л Ь Н
О М П А Р Н Е Ч У Т Ь Б О Л Ь Ш Е З П Т Ч Е М С Л Е Д О В А Л О
З П Т У Н И Х Б Ы О Т П А Л О В С Я К О Е Ж Е Л А Н И Е Е М У З
А В И Д О В А Т Ь Т Ч К И Н Е В Е С Е Л И Л И С Ь Б Ы О Н И П О
П О В О Д У Е Г О В Е Ч Н Ы Х Н Е Д О Р А З У М Е Н И Й С Ж Е Н
Щ И Н А М И З П Т З Н А Я З П Т Ч Т О С Л А В О Й Д А М С К О Г
О У Г О Д Н И К А Ж А К О Б Я З А Н Б О Я З Н И С П А Т Ь В О Д
И Н О Ч Е С Т В Е З П Т А Н Е П Р И Р О Д Н О Й О З А Б О Ч Е Н
Н О С Т И Т Ч К А С Н Е Д А В Н И Х П О Р Ш У Т И В О В С Е С Т
Р А Ш И Л С Я З А С Ы П А Т Ь И З З А П О С Т О Я Н Н Ы Х К О Ш
М А Р О В Т Ч К Х О Р О Ш О Е Щ Е З П Т Ч Т О О Н П О Н А Т У Р
Е Ч Е Л О В Е К Л Е Г К О М Ы С Л Е Н Н Ы Й И Н Е С П О С О Б Н
Ы Й З А Ц И К Л И В А Т Ь С Я Н А Ч Е М Л И Б О З П Т И Н А Ч Е
М О Ж Н О И У М О М Т Р О Н У Т Ь С Я'
```

```
>>> key
{'88': 'О', '73': 'В', '16': 'Е', '90': 'З', '40': 'П', '49': 'Т',
 '31': 'С', '78': 'Л', '56': 'И', '70': 'Д', '20': 'У', '74':
 'Я', '94': 'Н', '46': 'Б', '68': 'А', '36': 'Э', '52': 'М',
 '64': 'Ь', '19': 'Р', '00': 'Ы', '28': 'Г', '03': 'Ч', '13':
 'Ш', '66': 'Х', '87': 'К', '83': 'Ж', '82': 'Й', '11': 'Щ',
 '97': 'Ц'}
```

Задача 1.2 Раскрыть шифр вертикальной перестановки:

АЕЧСЕ ЛЫАИЛ ОПЗИЕ СТЫБД ТТДРД ОВИГР ЙВКАЛ МАШ-
ЛУ ПЗЖТЯ РОСЗГ ЕНОПЫ ИОМЕО ОЯТТХ ОДАЛР УИВИО ООН-
НИ ОВЫЫБ ИАОРС ОТГАБ СОЕЧД ВУНЛУ НИМОЕ ШШАВН ЕАВ-
МЙ

Решение.

Длина текста 120 букв. Наиболее целесообразно было бы использо-
вать ключ длины 10 или 12 (близкой к $\sqrt{120}$). Проверим различные
длины ключей на основе известного соотношения гласных к согласным:
44% к 56%.

```
>>> def get_mse(text, n):
...     vn = lambda row: sum([x in list('АЕЁИОУЫЭЮЯ') for x in row])
...     table = np.array(list(text)).reshape((n, len(text) // n)).T
...     ratio = np.array([vn(row) / len(row) for row in table])
```

```

...     return sum((ratio - 0.44) ** 2) / (len(text) // n)
...
>>> mse = [(round(get_mse(text, i), 5), i) for i in [6, 8, 10, 12,
15]]
>>> sorted(mse, key = lambda x: x[0])
[(0.00216, 15), (0.02229, 12), (0.02577, 10), (0.03514, 8),
(0.03966, 6)]

```

Видим, что наименьшая среднеквадратичная ошибка достигается при ключе длины 15.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
А	И	Т	Д	К	П	З	О	Х	В	О	Р	О	У	А
Е	Л	Ы	О	А	З	Г	М	О	И	В	С	Е	Н	В
Ч	О	Б	В	Л	Ж	Е	Е	Д	О	Ы	О	Ч	И	Н
С	П	Д	И	М	Т	Н	О	А	О	Ы	Т	Д	М	Е
Е	З	Т	Г	А	Я	О	О	Л	О	Б	Г	В	О	А
Л	И	Т	Р	Ш	Р	П	Я	Р	Н	И	А	У	Е	В
Ы	Е	Д	Й	Л	О	Ы	Т	У	Н	А	Б	Н	Ш	М
Я	С	Р	В	У	С	И	Т	И	И	О	С	Л	Ш	Й

Обратим внимание на столбцы, в которых есть буква 'Ы' – с ними будет проще всего найти не встречающиеся биграммы. Например, столбец 11 сочетается только с 3 и 5 столбцами. Так как, например, 'ДЫМ' встретится чаще, чем 'МЫД', поставим столбцы в порядке 3 - 11 - 5. Во второй строке получаем триграмму 'ЫВА', после которой может быть 'Н', 'Т', 'Е', 'Ю', 'Л', 'Я'. Отметим кандидатами 1, 2, 13 и 14 столбец. В последней строке получается 'РОУЯ', если выбрать первый столбец – отбраковываем, при 14-м столбце в 5-й строке получится 'ТБАО' – отбраковываем. На третьей строке скорее будет 'БЫЛО', чем 'БЫЛЧ', поэтому остановимся на варианте 3 - 11 - 5 - 2.

1	6	3	11	5	2	7	8	9	10	4	12	13	14	15
А	П	Т	О	К	И	З	О	Х	В	Д	Р	О	У	А
Е	З	Ы	В	А	Л	Г	М	О	И	О	С	Е	Н	В
Ч	Ж	Б	Ы	Л	О	Е	Е	Д	О	В	О	Ч	И	Н
С	Т	Д	Ы	М	П	Н	О	А	О	И	Т	Д	М	Е
Е	Я	Т	Б	А	З	О	О	Л	О	Г	Г	В	О	А
Л	Р	Т	И	Ш	И	П	Я	Р	Н	Р	А	У	Е	В
Ы	О	Д	А	Л	Е	Ы	Т	У	Н	Й	Б	Н	Ш	М
Я	С	Р	О	У	С	И	Т	И	И	В	С	Л	Ш	Й

В первой строке видно слово 'ВОЗДУХ', 10 - (8, 13) - 7 - 4 - 14 - 9. На

третьей строке оказывается 'ОЕЕ', если выбрать 8-й столбец, и 'ОЧЕ', если выбрать 13-й. Установим столбцы по второму варианту.

1	6	3	11	5	2	12	8	15	10	13	7	4	14	9
А	П	Т	О	К	И	Р	О	А	В	О	З	Д	У	Х
Е	З	Ы	В	А	Л	С	М	В	И	Е	Г	О	Н	О
Ч	Ж	Б	Ы	Л	О	О	Е	Н	О	Ч	Е	В	И	Д
С	Т	Д	Ы	М	П	Т	О	Е	О	Д	Н	И	М	А
Е	Я	Т	Б	А	З	Г	О	А	О	В	О	Г	О	Л
Л	Р	Т	И	Ш	И	А	Я	В	Н	У	П	Р	Е	Р
Ы	О	Д	А	Л	Е	Б	Т	М	Н	Н	Ы	Й	Ш	У
Я	С	Р	О	У	С	С	Т	Й	И	Л	И	В	Ш	И

Видно, что эти два блока можно объединить. Кроме того, можно заметить слова 'ПОТОКИ' и 'ОЧЕВИДНО': 9 - 15 - 12, 6 - 8 - 3. Остаётся последний столбец, для которого становится ясно, что он должен находиться в конце таблицы.

Окончательный ответ:

П	О	Т	О	К	И	В	О	З	Д	У	Х	А	Р	А
З	М	Ы	В	А	Л	И	Е	Г	О	Н	О	В	С	Е
Ж	Е	Б	Ы	Л	О	О	Ч	Е	В	И	Д	Н	О	Ч
Т	О	Д	Ы	М	П	О	Д	Н	И	М	А	Е	Т	С
Я	О	Т	Б	А	З	О	В	О	Г	О	Л	А	Г	Е
Р	Я	Т	И	Ш	И	Н	У	П	Р	Е	Р	В	А	Л
О	Т	Д	А	Л	Е	Н	Н	Ы	Й	Ш	У	М	Б	Ы
С	Т	Р	О	У	С	И	Л	И	В	Ш	И	Й	С	Я

Подставим a_{n+r} из первого уравнения вместо a_{n-r} во второе уравнение и a_{n+3} вместо a_{n-3} в третье.

$$\begin{cases} a_n = a_{n+r} + a_{n+3}, & (1) \\ a_n = a_{n-r} + a_{n-r+3}, & (2) \\ a_n = a_{n-3} + a_{n+r-3}, & (3) \\ a_n = a_{n-2r} + a_{n-2r+3} + a_{n-r+3}, & (1+2) \\ a_n = a_{n-6} + a_{n+r-6} + a_{n+r-3}. & (1+3) \end{cases}$$

Теперь подставим второе уравнение в пятое.

$$\begin{cases} a_n = a_{n+r} + a_{n+3}, & (1) \\ a_n = a_{n-r} + a_{n-r+3}, & (2) \\ a_n = a_{n-3} + a_{n+r-3}, & (3) \\ a_n = a_{n-2r} + a_{n-2r+3} + a_{n-r+3}, & (1+2) \\ a_n = a_{n-6} + a_{n+r-6} + a_{n+r-3}. & (1+3) \\ a_n = a_{n-r-6} + a_{n-r-3} + a_{n+r-6} + a_{n+r-3}. & (1+2+3) \end{cases}$$

Таким образом, мы получили систему из $m = 6$ уравнений. Теперь подставим r , вместо членов последовательности a подставим члены последовательности z , опустим индексы n и получим систему линейных форм:

$$\begin{cases} z + z_5 + z_3 = L_1, \\ z + z_{-5} + z_{-2} = L_2, \\ z + z_{-3} + z_2 = L_3, \\ z + z_{-10} + z_{-7} + z_{-2} = L_4, \\ z + z_{-6} + z_{-1} + z_2 = L_5, \\ z + z_{-11} + z_{-8} + z_{-1} + z_2 = L_6. \end{cases}$$

Каждый z_i представляет собой $a_i \oplus \gamma_i$, где γ_i — это н.о.р.с.в. с $P(\gamma = 0) = P(f = 0) = \frac{3}{4}$. Пусть b_{ij} — это слагаемые правой стороны уравнений системы с a_i , а y_{ij} — слагаемые левой стороны уравнений системы с z_i , не содержащие z . Тогда уравнения первой системы принимают вид $a + \sum_{j=0}^t b_{ij} = 0$, а второй — $z + \sum_{j=0}^t y_{ij} = L_i$. Заметим, что в таком случае $P(z_i = a_i) = P(y_{ij} = b_{ij}) = \frac{3}{4} = p$.

Пусть вероятность $s = s(t, p) = P(y_i = b_i)$ не зависит от i . По формуле полной вероятности получим рекуррентное соотношение:

$$\begin{cases} s(t, p) = p \cdot s(t-1, p) + (1-p)(1-s(t-1, p)), \\ s(1, p) = p. \end{cases}$$

Поскольку $t = 2$, то $s = s(2, \frac{3}{4}) = \frac{3}{4} \cdot \frac{3}{4} + (1 - \frac{3}{4})(1 - \frac{3}{4}) = \frac{5}{8}$. Определим апостериорную вероятность того, что $z = a$ при условии события B_k : k из m линейных форм L_i равны нулю.

$$P(z = a | B_k) = \frac{\binom{m}{k} p s^k (1-s)^{m-k}}{\binom{m}{k} p s^k (1-s)^{m-k} + \binom{m}{k} (1-p) s^{m-k} (1-s)^k} = p^*$$

Найдём матожидания этой величины в двух разных случаях: $z = a$ и $z \neq a$:

$$\begin{aligned} E_0(p^*) &= E(p^* | z = a) = \\ &= \sum_{k=0}^m \binom{m}{k} \frac{p s^k (1-s)^{m-k}}{p s^k (1-s)^{m-k} + (1-p) s^{m-k} (1-s)^k} s^k (1-s)^{m-k} = \\ &= \sum_{k=0}^6 \binom{6}{k} \frac{\frac{3}{4} \cdot (\frac{5}{8})^k (\frac{3}{8})^{6-k}}{\frac{3}{4} \cdot (\frac{5}{8})^k (\frac{3}{8})^{6-k} + \frac{1}{4} \cdot (\frac{5}{8})^{6-k} (\frac{3}{8})^k} \left(\frac{5}{8}\right)^k \left(\frac{3}{8}\right)^{6-k} \approx 0.81 \end{aligned}$$

$$\begin{aligned} E_1(p^*) &= E(p^* | z \neq a) = \\ &= \sum_{k=0}^m \binom{m}{k} \frac{p s^k (1-s)^{m-k}}{p s^k (1-s)^{m-k} + (1-p) s^{m-k} (1-s)^k} s^{m-k} (1-s)^k = \\ &= \sum_{k=0}^6 \binom{6}{k} \frac{\frac{3}{4} \cdot (\frac{5}{8})^k (\frac{3}{8})^{6-k}}{\frac{3}{4} \cdot (\frac{5}{8})^k (\frac{3}{8})^{6-k} + \frac{1}{4} \cdot (\frac{5}{8})^{6-k} (\frac{3}{8})^k} \left(\frac{5}{8}\right)^{6-k} \left(\frac{3}{8}\right)^k \approx 0.56 \end{aligned}$$

Составим таблицу в соответствии с последней системой.

N	z	z_5	z_3	z_{-5}	z_{-2}	z_{-3}	z_2	z_{-10}	z_{-7}	z_{-6}	z_{-1}	z_{-11}	z_{-8}	L_1	L_2	L_3	L_4	L_5	L_6
1	0	1	1	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	1
2	1	1	0	1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	0
3	0	0	1	0	0	0	0	1	0	1	1	1	0	1	0	0	1	0	0
4	1	1	1	0	1	0	1	1	1	0	0	1	0	1	0	0	0	0	1
5	0	1	0	0	0	1	1	0	0	0	1	1	1	1	0	0	0	0	0
6	1	1	1	0	1	0	0	0	0	0	0	0	0	1	0	1	0	1	1
7	1	1	1	1	0	1	1	1	0	0	1	0	0	1	0	1	0	1	1
8	0	0	1	0	1	0	1	0	0	1	1	1	0	1	1	1	1	1	1
9	1	0	1	1	1	1	1	0	1	0	0	0	0	0	1	1	1	0	0
10	1	1	0	0	0	1	1	0	0	1	1	0	1	0	1	1	1	0	0
11	1	1	0	1	1	0	0	0	1	0	1	0	0	0	1	1	1	0	0
12	1	0	1	1	1	1	0	1	0	1	1	0	1	0	1	0	1	1	1
13	0	0	1	0	1	1	1	0	1	1	1	1	0	1	1	0	0	1	1
14	0	0	0	1	1	1	1	1	1	0	0	0	1	0	0	0	1	1	0
15	1	1	0	1	0	1	0	0	0	1	0	1	1	0	0	0	1	0	1
16	1	0	0	1	0	0	0	1	1	1	1	0	0	1	0	1	1	1	0
17	0	1	1	1	1	0	0	1	1	1	1	1	1	0	0	0	1	0	1
18	0	1	0	0	1	1	1	0	1	1	0	1	1	1	1	0	0	0	1
19	0	1	1	0	0	1	0	1	1	0	0	0	1	0	0	1	0	0	1
20	1	1	1	1	0	0	1	1	0	0	0	1	1	1	0	0	0	0	0
21	0	0	1	1	0	0	1	1	0	1	1	1	0	1	1	1	1	1	1
22	1	0	1	0	1	0	1	1	1	1	0	1	0	0	0	0	0	1	1
23	1	1	0	0	0	1	1	0	1	0	1	1	1	0	1	1	0	1	1
24	1	0	0	0	1	0	0	0	0	0	1	0	1	1	0	1	0	0	1
25	1	0	1	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	0
26	0	0	0	0	1	1	1	1	0	1	1	1	0	0	1	0	0	1	1
27	0	0	0	1	1	1	0	0	1	0	0	1	0	0	0	1	0	0	1
28	1	1	0	1	0	1	0	0	0	1	0	0	1	0	0	0	1	0	0
29	0	0	0	1	0	0	0	0	1	1	1	0	0	0	1	0	1	0	1
30	0	1	1	1	1	0	0	1	1	1	0	0	1	0	0	0	1	1	1
31	0	0	0	0	0	1	1	0	1	1	0	1	1	0	0	0	1	0	1

Выберем r строк, в которых L_i принимают наибольшее количество нулей. Это строки 2, 1, 5, 20, 28. Выразим a с соответствующими номерами через начальное заполнение регистров.

$$\begin{cases} a_2 = x_2 + x_5 = 1, \\ a_1 = x_1 + x_4 = 0, \\ a_5 = x_1 + x_3 + x_4 + x_5 = 0, \\ a_{20} = x_3 + x_4 + x_5 = 1, \\ a_{28} = x_2 = 1. \end{cases}$$

Решив систему уравнений, получим $\vec{x} = (1, 1, 0, 1, 0)$. Выполним проверку:

```
>>> def check_solution(x):
...     true_z = [int(c) for c in '0101011011110011000101111001000']
...     z = []
...     for i in range(len(true_z)):
...         gamma = x[3] * x[4]
...         a = (x[0] + x[3]) % 2
...         z += [(gamma + a) % 2]
...         x = x[1:] + [a]
...     return z == true_z
...
>>> x = [1, 1, 0, 1, 0]
>>> check_solution(x)
False
```

К сожалению, не повезло. Значит, надо выбрать другие строки. Попробуем взять 2, 1, 5, 3, 28. Тогда четвёртое уравнение в системе изменится на $a_3 = x_1 + x_3 + x_4 = 0$. Новая система будет иметь два решения: $\vec{x} = (0, 1, 0, 0, 0)$ и $\vec{x} = (1, 1, 0, 1, 0)$. Второе из них уже было проверено выше, проверим первое:

```
>>> x = [0, 1, 0, 0, 0]
>>> check_solution(x)
True
```

Победа.

Ответ: $m = 6$, $E_0(p^*) \approx 0.81$, $E_1(p^*) \approx 0.56$, $\vec{x} = (0, 1, 0, 0, 0)$.