НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Т.А. Гультяева

ОСНОВЫ ТЕОРИИ ИНФОРМАЦИИ И КРИПТОГРАФИИ

Конспект лекций
для студентов III курса ФПМИ по специальностям
“Прикладная математика и информатика” (010500),
“Математическое обеспечение и администрирование информационных систем” (010503),
“Прикладная информатика в менеджменте” (080801)

НОВОСИБИРСК
2010

Рецензенты:

Н.Л. Долозов, канд. техн. наук, доц. кафедры программных систем и баз данных.

В.Е. Хищенко, канд. техн. наук, доц. кафедры защиты информации

Составитель: *Т. А. Гультяева*, ассистент кафедры программных систем и баз данных.

Работа подготовлена на кафедре программных систем и баз данных
для студентов III курса ФПМИ по специальностям
“Прикладная математика и информатика” (010500),
“Математическое обеспечение и администрирование
информационных систем” (010503),
“Прикладная информатика в менеджменте” (080801)

Конспект лекций посвящен основам теории информации и криптографии и охватывает широкий круг вопросов, позволяющих студента получить базовые знания по курсу.

Он также могут быть полезны для инженеров и сотрудников, осваивающих базовые знания основ теории информации и криптографии.

Тема №1
Основные аспекты теории информации

1.1 Введение в теорию информации. Задачи, решаемые в рамках теории информации

Теория информации – наука, появившаяся в 1948 г. в результате публикации работы Клода Шеннона «Математическая теория связи».

Ниже представлена структурная схема типичной системы передачи или хранения информации (рис. 1).

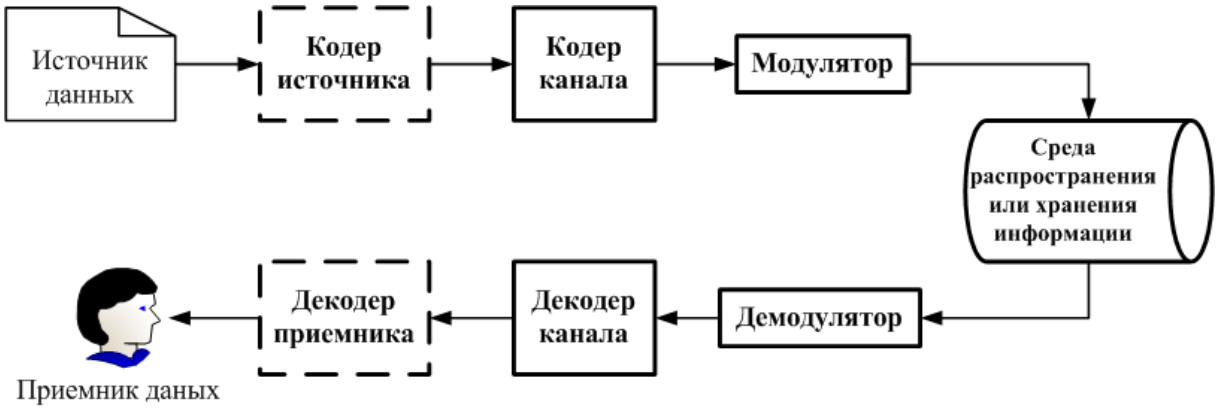

Рис. 1. Структурная схема типичной системы передачи или хранения информации

Источник – любое устройство или объект живой природы, порождающие сообщения, которые должны быть перемещены во времени или пространстве. Например, клавиатура, человек.

Цель кодера источника – представление информации в наиболее компактном виде. Это надо для того, чтобы эффективно использовать ресурсы канала связи или запоминающего устройства.

Кодер канала производит обработку информации с целью защиты сообщений от помех при передаче по каналу связи либо от возможных искажений при их хранении.

Модулятор используется для преобразования сообщений, формируемых кодером канала, в сигналы, согласованные с физической природой канала связи или средой накопления информации.

Остальные блоки выполняют обратные действия и представляют полученную информацию в удобном для использования виде.

Таким образом, в теории информации можно выделить следующие задачи:

1. Кодирование сообщений, переданных дискретным источником (кодирование без потерь, кодирование для канала без шума, сжатие информации).
2. Кодирование информации для передачи по каналу с шумом (защита информации от помех в канале связи).
3. Кодирование с заданным критерием качества. Например, в некоторых системах связи искажение информации допустимо. Здесь речь идет о методах кодирования, обеспечивающих наилучший компромисс между качеством (оцениваемым некоторой мерой искажения) и затратами на передачу информации. Сегодня задачи этого типа особенно актуальны, так как они находят широкое применение для цифровой передачи речи, видео- и аудиоинформации.

4. Кодирование информации для системы со многими пользователями. Например, оптимальное взаимодействие абонентов, использующих какой-либо общий ресурс, например, канал связи (системы мобильной связи, локальные сети ЭВМ).
5. Секретная связь, системы защиты информации от несанкционированного доступа.

1.2 Вероятностно-статистические модели сообщений и их свойства

1.2.1 Источники дискретных сообщений и их вероятностные модели

Пусть рассматривается произвольный источник сообщений. Каждое сообщение – это последовательность символов (например, букв русского алфавита, точек и тире в телеграфии, *0* и *1* в компьютерной логике и т. д.).

Отдельные символы сообщения: ξ – это случайная величина, принимающая всевозможные значения из алфавита сообщений A: $\xi \in A$.

Если мощность A конечна: $A = \{a^{(1)}, a^{(2)}, ..., a^{(N)}\}$, $2 \le N < \infty$, то это источник дискретного сообщения (ИДС). Иначе – источник непрерывного сообщения.

Мы будем использовать модель ИДС как наиболее часто рассматриваемую.

$a^{(1)}, a^{(2)}, \ldots, a^{(N)}$ – это символы алфавита; N – мощность алфавита. Появление символа алфавита в сообщении описывается дискретной вероятностной моделью.

Дискретный ансамбль B – это полная совокупность состояний (символов алфавита) с вероятностью их появления:

$$B = \begin{pmatrix} a^{(1)} & a^{(2)} & ... & a^{(N)} \\ P(a^{(1)}) & P(a^{(2)}) & ... & P(a^{(N)}) \end{pmatrix}, \quad \sum_{i=1}^{N} P(a^{(i)}) = 1, \quad 0 < P(a^{(i)}) < 1,$$

где $P(a) = P\{\xi = a\}$.

Понятно, что данная модель $\langle A, P(a) \rangle$ описывает лишь одиночный случайный символ сообщения.

Сообщение, порождаемое ИДС – это, в общем случае, последовательность $n \ge 1$ случайных символов: $\Xi_n = (\xi_1, \xi_2, ..., \xi_n) \in A^n$. При этом полное вероятностное описание ИДС задается вероятностной моделью случайного временного ряда (случайного процесса) Ξ_n с дискретным временем $t \in \boldsymbol{N}$ и дискретным пространством состояний A, $n = 1,2,...$: $\langle A^n, P_n(a_1, a_2, ..., a_n) \rangle$,

$P_n(a_1, a_2, ..., a_n) = P\{\xi_1 = a_1, \xi_2 = a_2, ..., \xi_n = a_n\}$, $a_1, a_2, ..., a_n \in A$,

где $P_n(a_1, a_2, ..., a_n)$ – n-мерное дискретное распределение вероятностей n-символьного сообщения.

ИДС является стационарным, если случайный процесс Ξ инвариантен относительно сдвига начала отсчета t. Стационарный ИДС является источником без памяти, если порождаемые ИДС случайные символы независимы в совокупности и одинаковы распределены.

1.2.2 Собственная информация

Очевидно, что задача количественного измерения информации возникла при решении конкретных практических задач. Например, можно стремиться уменьшить количество электрических сигналов, необходимых для передачи сообщения по каналам связи. Поэтому разумной мерой ин-

формации, содержащейся в сообщении, является мера, монотонно связанная с затратами на передачу сообщения.

Собственная информация $I(x)$ сообщения x, выбираемого из дискретного ансамбля $B=\{X,P(x)\}$ – это:

$$I(x)=\log_b \frac{1}{P(x)}=-\log_b P(x). \qquad \textbf{(1.1)}$$

Если $b=2$, то количество информации измеряется в битах; $b=e$ – натах; $b=10$ –ди/хартли (Хартли впервые предложил эту формулу в виде $I(x)=-\lg \frac{1}{N}$ в 1928 г.).

Свойства собственной информации:

1. $I(x)\geq 0, x\in X$.
2. Монотонность: если $x_1,x_2\in X$, $P(x_1)\geq P(x_2)$, то $I(x_1)\leq I(x_2)$.
3. Аддитивность: (для независимых сообщений) $x_1,...,x_n : I(x_1,...,x_n)=\sum_{i=1}^{n} I(x_i)$.

Пример

Дан ансамбль: $B=\begin{pmatrix} 'a' & 'b' & 'c' & 'd' \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & \frac{1}{8} \end{pmatrix}$. Кодирование будем производить, используя *0* и *1*.

$I('a')=-\log_2 \frac{1}{2}=1(бит)$

$I('b')=2, \quad I('c')=I('d')=3$.●

Видно, что чем выше вероятность появления символа в сообщение (чем чаще он появляется), тем меньшим числом бит его надо кодировать, чтобы сэкономить на длине закодированного сообщения.

Таким образом, собственная информация характеризует степень неожиданности появления конкретного сообщения.

Пример

Определить собственную информацию, содержащуюся в изображении, при условии, что оно состоит из *500* строк по *500* элементов в каждой строке. Яркость каждого элемента передается *8* уровнями. Яркости различных элементов независимы.

Решение

Случайная величина X – яркость одного элемента изображения. $P(x_i)=\frac{1}{n}=\frac{1}{8}$.

$I(x_i)\overset{(1)}{=}\log_2\left(\frac{1}{8}\right)^{-1}=3$. Изображение содержит $N=25\cdot 10^4$ элементов. Так как яркости элементов независимы, то $I(image)=N\cdot I(x_i)=7.5\cdot 10^5(бит)$.●

Пример

На экране радара 10×10 полос; изображение появляется в виде яркой отметки. Все положения равновероятны. Определить количество собственной информации, содержащейся в сообщениях: (а) объект находится в *46* квадранте, (б) объект находится в *5*-ой горизонтальной строке.

Решение

(а) $I(x_{46}) = -\log_2 \frac{1}{100} \approx 6.644.$

(б) $I(x_5) = -\log_2 \frac{10}{100} \approx 3.322.\bullet$

1.2.3 Взаимная информация

Рассмотрим два ансамбля сообщений: X, Y . $X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ P(x_1) & P(x_2) & \dots & P(x_n) \end{pmatrix}$ – ансамбль сообщений на входе системы, $Y = \begin{pmatrix} y_1 & y_2 & \dots & y_m \\ P(y_1) & P(y_2) & \dots & P(y_m) \end{pmatrix}$ – на выходе. Естественно, что они зависимы.

В результате опыта (приема символа y_k) апостериорная вероятность появления символа x_j изменяется в сравнении с априорной. Тогда количество информации символа сообщения x_j, доставляемое символом y_k, можно определить как логарифм отношения апостериорной вероятности к априорной:

$$I(x_j; y_k) = \log \frac{P(x_j \mid y_k)}{P(x_j)}. \qquad \textbf{(1.2)}$$

Это и есть определение взаимной информации.

Свойства взаимной информации:

1. $-\infty < I(x_j; y_k) < \infty.$

   Если $I(x_j; y_k) < 0$, то считается, что наступление события y_k делает наступление события x_j менее вероятным, чем оно было априори до наблюдения y_k.

2. $I(x_j; y_k) \le I(x_j)$, $I(x_j; y_k) \le I(y_{k,})$ – взаимная информация не превышает свою собственную.

   При данном $P(x_j)$ взаимная информация достигает своего максимума, когда принятый символ y_k однозначно определяет переданный символ x_j.

   При этом $P(x_j \mid y_k) = \begin{cases} 0, j \ne k \\ 1, j = k \end{cases}$ и это максимальное значение равно $I(x_j; y_k) = -\log P(x_j)$, то есть собственной информации, определяемой только априорной вероятностью символа x_j.

3. Симметрия: $I(x_j;y_k)=I(y_k;x_j)$. Информация, содержащаяся в x_j относительно y_k, равна информации, содержащейся в y_k относительно x_j. Поэтому $I(x_j;y_k)$ – это именно *взаимная* информация.
4. Аддитивность: $I(x_j,z_i;y_k,q_l)=I(x_j;y_k)+I(z_i;q_l)$, только при условии, что ансамбли X,Y независимы от Z,Q.

Информация, содержащаяся в реализации y_k принятого сигнала относительно ансамбля передаваемых сообщений X, определяется следующей формулой:

$$I(X;y_k)=M\left(\log\frac{p(X\mid y_k)}{p(X)}\right)=\sum_{j=1}^{n}p(x_j\mid y_k)\log\frac{p(x_j\mid y_k)}{p(x_j)}=$$
$$=\sum_{j=1}^{n}p(x_j\mid y_k)I(x_j;y_k). \qquad \textbf{(1.3)}$$

Наконец, средняя взаимная информация между ансамблем принимаемых сигналов Y и ансамблем передаваемых сообщением X определяется формулой (4):

$$I(X;Y)=M\left(\log\frac{p(X\mid Y)}{p(X)}\right)=\sum_{j=1}^{n}\sum_{k=1}^{m}p(x_j;y_k)\log\frac{p(x_j\mid y_k)}{p(x_j)}=$$
$$=\sum_{k=1}^{m}p(y_k)I(X;y_k), \qquad \textbf{(1.4)}$$

то есть то количество информации, которое содержится в среднем в ансамбле принимаемых символов Y относительно ансамбля передаваемых символов X.

Пример.

По дискретному каналу передаются сообщения x_1 и x_2. Вследствие шумов на выходе канала появляются сигналы y_1,y_2,y_3. Вероятности их совместного появления $P(x_i;y_j)$ заданы в таблице:

	y_1	y_2	y_3
x_1	1/4	1/16	1/8
x_2	1/8	3/16	1/4

Необходимо найти взаимную информацию $I(x_2;y_2)$ и $I(x_1;y_3)$.

Решение

$I(x_1;y_3)=\log\frac{P(x_1\mid y_3)}{P(x_1)}$.

$P(x_1)=\sum_{j=1}^{3}P(x_1;y_j)=\frac{7}{16}$, $P(y_3)=\sum_{i=1}^{2}P(y_3;x_i)=\frac{3}{8}$.

$P(x_1\mid y_3)=\frac{P(x_1;y_3)}{P(y_3)}=\frac{1}{3}$. Значит, $I(x_1;y_3)=\log_2\frac{1/3}{7/16}\approx -0.39(\textit{бит})$.

Аналогично: $I(x_2;y_2)=\log_2\frac{1/3}{1/4}\approx 0.415(\textit{бит})$.●

Пример

Дан ансамбль сообщений:

	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$P(x_i)$	1/2	1/4	1/8	1/32	1/32	1/32	1/32
Код	*001*	*010*	*100*	*011*	*101*	*110*	*111*

Сообщение x_4 поступило в кодер. Вычислить дополнительную информацию об этом сообщении, доставляемую каждым последующим символом на выходе кодера.

Решение

Найдем взаимную информацию, содержащуюся в первом кодовом символе *'1'* относительно сообщения x_4: $I(x_4;'1') = \log\dfrac{P(x_4|'1')}{P(x_4)}$ (см. следующий рисунок).

Кодовое слово на выходе:

1	*?*	*?*

$P(x_4|'1') = \dfrac{P(x_4,'1')}{P('1')} \overset{\text{по ф–ле Байеса}}{=} \dfrac{P(x_4)P('1'|x_4)}{\sum_j P(x_j)P('1'|x_j)}$, где x_4 – это гипотеза о том, что было передано *4*-ое сообщение.

$P('1'|x_j) = \begin{cases} 1, \textit{если} & j = 2,4,5,6 \\ 0, \textit{иначе} & \end{cases}$ – это вероятность появления '1' на первом месте в кодовом слове.

$$P(x_4|'1') = \frac{1/32\cdot 1}{1/8\cdot 1+1/32\cdot 1+1/32\cdot 1+1/32\cdot 1} = \frac{1}{7}.$$

Таким образом, $I(x_4;'1') = \log_2\dfrac{1/7}{1/32} \approx 2.2(\textit{бит})$.

Информация, содержащаяся во втором кодовом символе '0' при условии, что первый кодовый символ равен '1':

$I(x_4;'0'|'1') = \log\dfrac{P(x_4|'1''0')}{P(x_4|'1')} = \log_2\dfrac{1/5}{1/7} \approx 0.48(\textit{бит})$, где

$$P(x_4|'1''0') \overset{\text{по ф–ле Байеса}}{=} \frac{P(x_4)P('1''0'|x_4)}{\sum_j P(x_j)P('1''0'|x_j)} = \frac{1/32\cdot 1}{1/8\cdot 1+1/32\cdot 1} = \frac{1}{5}.$$

Кодовое слово на выходе:

1	*0*	*?*

Информация, содержащаяся в третьем кодовом символе '1' при условии, что первые два кодовых символа равны '10':

$$I(x_4;'1'|'1''0') = \log\frac{P(x_4|'1''0''1')}{P(x_4|'1''0')} = \log_2\frac{1}{1/5} \approx 2.32(\textit{бит}).$$

Кодовое слово на выходе:

1	*0*	*1*

Так как сообщение x_j и кодовые слова однозначно связаны, то

$$I(x_4) = I(x_4;'1') + I(x_4;'0'|'1') + I(x_4;'1'|'1''0') = 5(\textit{бит}).$$

По свойству 2) получаем тот же ответ: $I(x_4) = -\log_2 P(x_4) = \log_2 32 = 5(\textit{бит})$. ●

Заметим, что среднее значение, или математическое ожидание, вычисленные по ансамблю, будут характеристикой информативности всего ансамбля: для этого вводят понятие энтропии.

1.2.4 Энтропия

Пусть ИДС описывается некоторым дискретным ансамблем: $X = \begin{pmatrix} x_1 & x_2 & ... & x_n \\ P(x_1) & P(x_2) & ... & P(x_n) \end{pmatrix}$. Тогда энтропия ИДС (или энтропия случайного символа ξ) вычисляется по формуле (5):

$$H(\xi) = M\big(I(\xi)\big) = M\big(-\log P(\xi)\big) = -\sum_{i=1}^{n} P(x_i)\log P(x_i). \qquad \textbf{(1.5)}$$

Впервые эта мера была предложена Клодом Шенноном в “Математической теории связи”. Вид этой формулы совпадает с полученным ранее результатом Больцмана для энтропии термодинамических систем.

Энтропию можно интерпретировать как количественную меру априорной неосведомленности о том, какое из сообщений будет порождено источником (какой из x_j будет подан на вход). Таким образом, энтропия – это мера неопределенности. Другое название энтропии – среднее количество собственной информации.

Свойства энтропии:

1. $H(\xi) \geq 0$.

   $H(\xi) = 0$, если ξ неслучайно.

2. $H(\xi) \leq \log n$.

   $H(\xi) = \log n$, если $P(x_i) = \frac{1}{n}$. Это возможно, если источник без памяти или все сообщения x_i равновероятны.

3. Аддитивность: в последовательности m независимых символов энтропия равна сумме энтропий, содержащихся в отдельных символах:

   $H[\xi_1,..., \xi_m] = \sum_{i=1}^{m} H(\xi_i)$.

4. Если распределение вероятностей ансамблей одинаково, а ансамбли отличаются только порядком следования элементов, то их энтропии равны.

Пример

Рассмотрим двоичный ансамбль: $X = \begin{pmatrix} '0' & '1' \\ 1-p & p \end{pmatrix}$.

Энтропия: $H(X) = -p\log_2 p - (1-p)\log_2(1-p) \overset{обозначим}{=} h(p)$.

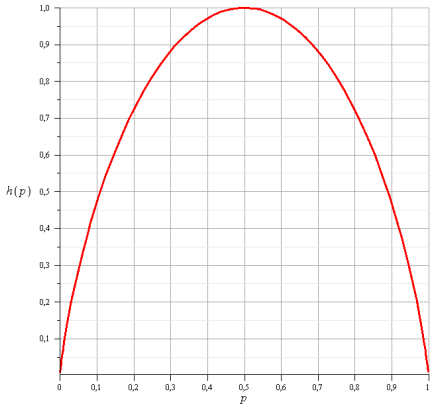

Рис. 2. График функции $h(p)$

Таким образом, *максимальное* среднее количество информации, переносимое двоичным сигналом – это 1 бит.●

Пример

На измерительной станции есть *2* прибора:
1-ый имеет *100* делений, его показания меняются каждые *0.05* секунд.
2-ой имеет *10* делений, его показания меняются каждые *0.01* секунд.
Какова наибольшая средняя информация, поставляемая двумя приборами в *1* секунду?

Решение

Так как все деления на приборах равновероятны, то энтропия *1*-ого прибора: $H_1^{2-ое\ св-во} = \log_2 100$. Аналогично для *2*-ого прибора: $H_2 = \log_2 10$.

Число измерений в секунду *1*-ым прибором: $n_1 = \frac{1}{0.05} = 20$, *2*-ым: $n_2 = \frac{1}{0.01} = 100$. Энтропия двух приборов в *1* секунду (их показания независимы друг от друга): $H^{3-ее\ св-во} = n_1 H_1 + n_2 H_2 \approx 464(бит/сек)$.●

Пример

Производится стрельба по двум мишеням. По *1*-ой сделано *2* выстрела, по *2*-ой – *3*. Вероятность попадания при одном выстреле: *1/2* и *1/3* соответственно. Исход стрельбы (при одном попадании) по какой мишени более не определен?

Решение

Исход стрельбы определяется числом попаданий в мишень, которое подчиняется биномиальному закону распределения: $P(\xi = m) = C_n^m p^m (1-p)^{n-m}$. Здесь m – это число попаданий в мишень, p – вероятность попадания в мишень.
Ряд распределения для числа попаданий в *1*-ую мишень при $n = 2$ (число выстрелов по мишени) и при вероятности попадания в мишень $p = \frac{1}{2}$ приведен далее.

m	0	1	2
$P(\xi = m)$ (вероятность m попаданий)	*1/4*	*1/2*	*1/4*

Аналогично – ряд для *2*-ой мишени:

m	0	1	2	3
$P(\xi = m)$ (вероятность m попаданий)	*8/27*	*4/9*	*2/9*	*1/27*

Мера неопределенности исхода стрельбы – это энтропия числа попаданий по мишени.
Энтропия при стрельбе по *1*-ой мишени:

$$H_1 = -\sum_{i=1}^{3} P(x_i)\log_2 P(x_i) = -\left[2\cdot\frac{1}{4}\log_2\frac{1}{4} + \frac{1}{2}\log_2\frac{1}{2}\right] \approx 1.5(\textit{бит}).$$

Энтропия при стрельбе по *2*-ой мишени: $H_2 = 1.7(\textit{бит})$.
Таким образом, исход стрельбы по *2*-ой мишени обладает большей неопределенностью.●

1.2.5 Условная энтропия

Пусть имеются два статистически независимых конечных ансамбля символов X, Y. Пары символов $x_j y_k$ с вероятностью $P(x_j, y_k)$ – это элементарные символы объединенного ансамбля XY с энтропией, вычисляемой по формуле (6):

$$H(XY) = -\sum_{j=1}^{n}\sum_{k=1}^{m} P(x_j, y_k)\log P(x_j, y_k). \qquad \textbf{(1.6)}$$

Появление символа x_j вызывает появление символа y_k с вероятностью $P(y_k \mid x_j) = \dfrac{P(x_j, y_k)}{P(x_j)}$.

Частная условная энтропия:

$$H(Y \mid x_j) = M\left(-\log P(Y \mid x_j)\right) = -\sum_{k=1}^{m} P(y_k \mid x_j)\log P(y_k \mid x_j). \qquad \textbf{(1.7)}$$

Средняя условная энтропия ансамбля Y при условии, что ансамбль X известен:

$$H(Y \mid X) = \sum_{j=1}^{n} P(x_j)H(Y \mid x_j) = -\sum_{j=1}^{n} P(x_j)\sum_{k=1}^{m} P(y_k \mid x_j)\log P(y_k \mid x_j) = \\ = -\sum_{j=1}^{n}\sum_{k=1}^{m} P(x_j, y_k)\log P(y_k \mid x_j). \qquad \textbf{(1.8)}$$

Свойства условной энтропии:

1. $H(XY) = H(X) + H(Y \mid X) = H(Y) + H(X \mid Y)$, при условии, что X, Y зависимы.
   Кроме того:
   $H(X_1, X_2, ..., X_n) = H(X_1) + H(X_2 \mid X_1) + ... + H(X_n \mid X_1, X_2, ..., X_{n-1})$.
2. $H(XY) = H(X) + H(Y)$, если X, Y независимы.
3. $H(Y \mid X) \le H(Y)$, $H(X \mid Y) \le H(X)$, причем равенство только в случае, если X, Y независимы.
4. $H(X \mid Y) \ge 0$.
5. $H(X \mid YZ) \le H(X \mid Y)$.

   Причем равенство только в случае, если X, Y условно независимы при всех $z \in Z$.

Далее приводятся формулы для *связи* средней взаимной информации и энтропии:

1. $I(X,Y) = H(X) + H(Y) - H(XY) = H(Y) - H(Y \mid X)$.
2. $I(X,Y) = H(Y) - H(Y \mid X) = H(X) - H(X \mid Y)$.

3. $I(X,X)=H(X)$, то есть энтропия – это частный случай функционала средней взаимной информации.
4. $0 \le I(X,Y) \le \min(H(X),H(Y))$.

Причем равенство нулю возможно тогда и только тогда, когда X,Y статистически независимы.

Пример

Вычислить энтропию $H(X)$, условную энтропию $H(X|Y)$ и среднюю взаимную информацию $I(X,Y)$, если дано $P(x_j,y_i)$ двух ансамблей X,Y:

	x_1	x_2	$P(y_i)$
y_1	0.5	0	0.5
y_2	0.25	0.25	0.5
$P(x_j)$	0.75	0.25	

Решение

Найдем $P(x_j|y_i)=\dfrac{P(x_j,y_i)}{P(y_i)}$:

	x_1	x_2
y_1	1	0
y_2	0.5	0.5

Энтропия $H(X|y_1)=0$,

$H(X|y_2)=1$,

$H(X|Y)=0.5\cdot 0+0.5\cdot 1=0.5$,

$H(X)=-0.75\log_2 0.75-0.25\log_2 0.25\approx 0.811.$

Средняя взаимная информация: $I(Y,X)=H(X)-H(X|Y)\approx 0.311.\bullet$

1.2.6 Избыточность

Считают, что имеется избыточность, если количество информации, содержащейся в сигнале (энтропия сигнала), меньше того количества, которое этот сигнал мог бы содержать по своей физической природе.

Пусть сигнал длиной n символов содержит количество информации H; кроме того, наибольшее количество информации, которое может содержаться в данном сигнале с учетом наложенных ограничений (таких, как основание кода, заданная средняя мощность сигнала и т.п.), равно H_{max}. Тогда количественной мерой избыточности является величина:

$$K=1-\frac{H}{H_{\max}}. \qquad \textbf{(1.9)}$$

Причина избыточности – статистические связи между символами сигнала и неэкстремальность распределения вероятностей отдельных символов. Введение избыточности приводит к удлинению сигнала, но повышает его информационную устойчивость при воздействии помех.

Статистические связи между сигналами наблюдаются в случае ИДС с памятью: здесь вероятность выдачи им очередного элемента сообщения x_j зависит от того, какие элементарные сообщения были выданы ранее.

Примером ИДС с памятью является источник связного русского текста, где в качестве элементарных сообщений выступают буквы. Сочетание ‘ар’ будет встречаться чаще, чем ‘аъ’.

Избыточность показывает, какая доля максимально возможной при заданном объеме алфавита неопределенности (энтропии) не используется источником.

Пример

Дан ансамбль: $X = \begin{Bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \\ 0.4 & 0.2 & 0.15 & 0.1 & 0.1 & 0.05 \end{Bmatrix}$.

Символы в последовательности независимы. Найти энтропию источника и вычислить избыточность.

Решение

Энтропия: $H(X) \approx 2.27(бит)$.

Избыточность за счет неоптимальности (не равновероятности) распределения элементарных сообщений в источнике: $K = 1 - \dfrac{2.27}{H_{\max}} \approx 0.13$, где $H_{\max} = \log_2 6$.●

Пример

Алфавит источника состоит из трех букв x_1, x_2, x_3. Определить энтропию на *1* букву текста $X^{(1)}X^{(2)}$ для следующих случаев:

(а) появление букв не равновероятно: $p(x_1) = 0.5, \quad p(x_2) = p(x_3) = 0.25$, а символы в последовательности на выходе источника статистически зависимы. Условные вероятности переходов $P(x_j^{(2)} \mid x_i^{(1)})$ заданы таблицей:

i – индекс предыдущей буквы	j – индекс последующей буквы		
	1	2	3
1	0.4	0.2	0.4
2	0.0	0.6	0.4
3	0.3	0.0	0.7

(б) вероятности букв те же, что и в пункте *(а)*, но символы независимы;

(в) символы в последовательности независимы, а вероятности букв одинаковы.

Вычислить избыточность источника для случаев *(а)* и *(б)*.

Решение

(а) В случае не равновероятных и зависимых сообщений энтропию текста будем считать по свойству 1) условной энтропии, где $H(X^{(1)}) = -\sum_{j=1}^{m} p(x_j) \log p(x_j)$,

а условную энтропию по формуле (8):

$$H(X^{(2)/} \mid X^{(1)}) = -\sum_{i=1}^{3}\sum_{j=1}^{3} p(x_i^{(1)})\, p(x_j^{(2)} \mid x_i^{(1)}) \log_2 p(x_j^{(2)} \mid p(x_i^{(1)}) =$$

$$= -\Big[p(x_1^{(1)})\Big(p(x_1^{(2)} \mid x_1^{(1)}) \log_2 p(x_1^{(2)} \mid x_1^{(1)}) + p(x_2^{(2)} \mid x_1^{(1)}) \log_2 p(x_2^{(2)} \mid x_1^{(1)}) + p(x_3^{(2)} \mid x_1^{(1)}) \log_2 p(x_3^{(2)} \mid x_1^{(1)})\Big) +$$

$$+ p(x_2^{(1)})\Big(p(x_1^{(2)} \mid x_2^{(1)}) \log_2 p(x_1^{(2)} \mid x_2^{(1)}) + p(x_2^{(2)} \mid x_2^{(1)}) \log_2 p(x_2^{(2)} \mid x_2^{(1)}) + p(x_3^{(2)} \mid x_2^{(1)}) \log_2 p(x_3^{(2)} \mid x_2^{(1)})\Big) +$$

$$+ p(x_3^{(1)})\Big(p(x_1^{(2)} \mid x_3^{(1)}) \log_2 p(x_1^{(2)} \mid x_3^{(1)}) + + p(x_2^{(2)} \mid x_3^{(1)}) \log_2 p(x_2^{(2)} \mid x_3^{(1)}) + p(x_3^{(2)} \mid x_3^{(1)}) \log_2 p(x_3^{(2)} \mid x_3^{(1)})\Big)\Big] =$$

$$= -\left[\frac{1}{2}(0.4\log_2 0.4 + 0.2\log_2 0.2 + 0.4\log_2 0.4) + \frac{1}{4}(0 + 0.6\log_2 0.2 + 0.4\log_2 0.4) + \frac{1}{4}(0.3\log_2 0.3 + 0.7\log_2 0.7)\right] =$$

$$= -\left[\frac{1}{2}(2 \cdot 0.52 + 0.46) + \frac{1}{4}(0.52 + 0.44) + \frac{1}{4}(0.52 + 0.36)\right] \approx 1.21 \quad бит$$

Энтропия на один символ:

$$H_1(X^{(1)}X^{(2)}) = \frac{H(X^{(1)}) + H(X^{(2)} \mid X^{(1)})}{2} = \frac{1.5 + 1.21}{2} = 1.36\, бит/симв.$$

(б) При не равновероятных, но независимых сообщениях энтропия по свойству 2) условной энтропии:

$$H(X) = -\sum_{j=1}^{n} p(x_j) \log p(x_j) = -\left[\frac{1}{2}\log_2 \frac{1}{2} + 2\frac{1}{4}\log_2 \frac{1}{4}\right] = 1.5\, бит/симв.$$

Избыточность, обусловленная статистической зависимостью:

$$k_1 = 1 - H_1(X^{(1)}X^{(2)}) / H(X) = 1 - 1.36/1.5 = 1 - 0.9 = 0.1.$$

(в) В случае равновероятных и независимых сообщениях энтропия:

$$H_{\max}(X) = \log_2 m = \log_2 3 \approx 1.585 \quad бит.$$

Избыточность, обусловленная неоптимальностью распределения:

$$k_2 = 1 - H(X) / H_{\max}(X) = 1 - 1.5/1.585 = 1 - 0.95 = 0.05.$$

Полная избыточность (за счет неоптимальности распределения и наличия статистических зависимостей):

$$k = 1 - H_1(X) / H_{\max}(X) = 1 - 1.36/1.585 = 1 - 0.86 = 0.14.\bullet$$

Тема №2
Основные принципы кодирования

2.1 Введение в теорию кодирования

Рисунок 1 из первой темы можно упростить:

Рис. 3. Упрощенная структурная схема типичной системы передачи или хранения информации

Кодирующие устройство, в общем случае, может выполнять следующие операции:

1. Согласование источника с каналом (например, перевод реальных сообщений в электрические сигналы, модуляция непрерывных сигналов, квантование непрерывных сигналов, представление d-ичного дискретного сообщения в m-ичном коде).
2. Экономное представление информации с минимальной избыточностью; или, наоборот, разумное введение избыточности с целью повышения помехоустойчивости сообщения.

Конкретный пример для операции *1*-ого типа:
каждой букве русского алфавита ставится в соответствие символ из 2-ого алфавита: ‘А’→00000, ‘В’→00001,…

Конкретный пример для операции *2*-ого типа:
представление графической информации с использованием блочного сжатия (JPEG).

Наибольший практический интерес представляют операции *2*-ого типа. Далее рассмотрим теоретические аспекты экономного представления информации, то есть сжатие информации (*пункт 2.2*). Здесь выделяют два типа систем:

1. Сжатие без потерь информации (неразрушающие сжатие).
2. Сжатие с потерями информации (разрушающие сжатие).

Далее (*пункт 2.3*) рассмотрим теоретические основы помехоустойчивого кодирования.

2.2 Основы экономного кодирования

2.2.1 Сжатие без потерь информации

В системах неразрушающего сжатия декодер восстанавливает данные источника абсолютно без потерь (рис. 4).

Рис. 4. Схема сжатия без потерь

Здесь X – это последовательность длины n из символов алфавита A.
$B(X)$ – сжатые данные, представленные во второй последовательности длины m; m зависит от n.

Вводится следующий термин: коэффициент сжатия r:

$$r = \frac{размер\quad данных\quad источника}{размер\quad сжатых\quad данных} = \frac{n\log_2|A|}{m}.$$

2.2.2 Сжатие с потерями информации.

Далее приведена система разрушающего сжатия (рис.5):

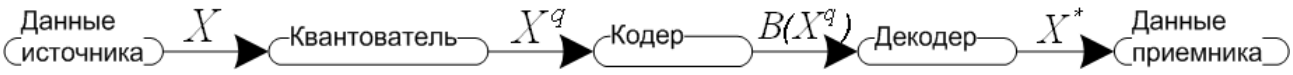

Рис. 5. Схема сжатия с потерями

Квантователь понижает размер алфавита (например, округление данных). Между X^q и $B(X^q)$ есть однозначное соответствие, но система в целом остается разрушающей, так как двум различным последовательностям X_1 и X_2 может соответствовать один и тот же X^*.

Вводят такое понятие, как искажение: $D = \frac{1}{n}\sum_i (x_i - x_i^*)^2$. Искажение является мерой среднеквадратичного различия между сообщениями X и X^*.

В лекционном курсе будут рассматриваться только кодеры, основанные на системе сжатия без потерь информации.

2.2.3 Кодеры, основанные на системе сжатия без потерь информации

В данном случае кодирующие устройство должно удовлетворять следующим условиям:

1. Обеспечивать безошибочную передачу информации, то есть взаимно однозначное соответствие между X и Y.
2. Обеспечивать кодирование наиболее экономным образом (с минимальной избыточностью).

Для выполнения *1*-ого требования:

а. Необходимо, чтобы различным буквам алфавита соответствовали различные кодовые слова.

б. Необходимо, чтобы была предусмотрена возможность разделения кодовых слов при их последовательной передаче. Для обеспечения этой возможности:

- используют специальные разделяющие символы;
- применяют кодовые слова одинаковой длины (равномерное кодирование);
- кодовая таблица составляется таким образом, чтобы никакое кодовое слово не являлось началом другого кодового слова.

Для выполнения *2*-ого требования необходимо добиваться при кодировании минимальной средней длины кодового слова: $\bar{l} = \sum_{k=1}^{s} l_k P(x_k)$, где l_k – это длина k-ого слова x_k с учетом разделительной буквы (если она используется).

Теорема 2.1 (Теорема Шеннона о кодировании в дискретных каналах без шума)

При кодировании множества сигналов с энтропией $H(X)$ при условии отсутствия шумов средняя длина кодового слова $\bar{l}$ не может быть меньше, чем $H(X)/\log_2 m$, где m – размер алфавита кодера, 2 – основание алфавита кода.

Если вероятности сигналов не являются целочисленными отрицательными степенями m (то есть все вероятности сообщений $x \in X$ имеют вид: $P(x) = 2^{-l(x)}$, где $l(x)$ – целое положительное число), достижение указанной нижней границы невозможно, но при кодировании достаточно длинными блоками к ней можно сколь угодно приближаться.

Существует несколько способов, позволяющих получать коды с малой избыточностью; причем все они обладают следующими свойствами:

1. Более вероятным буквам источника ставятся в соответствие более короткие кодовые слова.
2. Никакое кодовое слово не является началом другого более длинного кодового слова.
3. Все буквы алфавита, используемые для передачи информации приблизительно равновероятны.
4. Символы в последовательности на выходе кодера практически независимы.

2.2.4 Основные методы побуквенного кодирования

Простейшими кодами, с помощью которых можно выполнять сжатие информации, являются коды без памяти. Наилучшим из кодов, обладающих свойствами, перечисленными выше, является код Хаффмана.

Код Хаффмана – это двоичный префиксный код без памяти. Далее рассмотрим некоторые теоретические основы этого кода.

Префиксным множеством двоичных последовательностей S называется конечное множество двоичных последовательностей таких, что ни одна последовательность в этом множестве не является префиксом, или началом, другой последовательности в S.

Например, множество двоичных слов $S_1 = \{00,01,100,110,1010,1011\}$ является префиксным множеством, а $S_2 = \{00,001,1110\}$ – не является.

Каждый префиксный код является дешифрируемым (обратное неверно).

Кодирование кодом без памяти осуществляется следующим образом:

1. Необходимо составить полный список символов $a_1, a_2, ..., a_k$ алфавита A входного сигнала; на j-ом месте стоит символ с вероятностью $P(a_j)$, то есть на *1*-ом месте стоит наиболее часто встречающийся символ.
2. Для любого a_j назначить кодовое слово w_j из префиксного множества двоичной последовательности S.
3. На выходе кодера объединить в одну последовательность все полученные двоичные слова.

Если $S = \{w_1, w_2, ..., w_k\}$ – префиксное множество, то можно определить некоторый вектор $V(S) = (l_1, l_2, ..., l_k)$, состоящий из чисел, являющихся длинами соответствующих префиксных последовательностей (l_i является длиной w_i). Вектор $V(S)$ состоит из неуменьшающихся положительных целых чисел и называется вектор Крафта. Для него выполняется неравенство Крафта:

$$\sum_{i=1}^{k} 2^{-l_i} \le 1 \qquad \textbf{(2.1)}$$

Длины двоичных последовательностей в префиксном множестве удовлетворяют данному соотношению. Если в (1) строгое равенство, то код называется оптимальным.

Пример.
$S_1 = \{0,10,11\}$, $V(S) = (1,2,2)$.
$S_2 = \{0,10,110,111\}$, $V(S) = (1,2,3,3)$.●

Средняя длина двоичной последовательности на выходе кодера – это $\bar{l}(Y)$. Лучший код без памяти – это код, для которого средняя длина $\bar{l}(Y)$ минимальна. Для того, чтобы это выполнялось, необходимо найти соответствующий вектор Крафта $V(S) = (l_1, l_2, ..., l_k)$. Простым перебором такую задачу трудно осуществить при больших k.

2.2.4.1 Код Хаффмана

Алгоритм Хаффмана дает эффективный способ поиска оптимального вектора Крафта. Код, получаемый с использованием этого оптимального вектора, называется кодом Хаффмана[1].

Далее приведен алгоритм построения кодового дерева Хаффмана:

Инициализация. Список подлежащих обработки узлов состоит из $M_0 = M$ узлов, где M – мощность алфавита входных сообщений.

Каждому из узлов приписывается вероятность появления буквы.

Цикл. Пока $M_0 > 1$ в списке необработанных узлов находим два узла с наименьшими вероятностями. Они исключаются из списка, и вместо них вводится новый узел, которому приписывается суммарная вероятность двух исключенных узлов. Новый узел связывается ребрами с исключенными узлами. $M_0 \leftarrow M_0 - 1$.

Теорема 2.2
Код Хаффмана является оптимальным.

Пример

Дан ансамбль: $X = \begin{pmatrix} a & b & c & d & e & f \\ 0.35 & 0.2 & 0.15 & 0.1 & 0.1 & 0.1 \end{pmatrix}$.

Энтропия ансамбля: $H(X) = -\sum_{i=1}^{6} P(x_i) \log_2 P(x_i) \approx 2.4$.

На рис. 6 приведено кодовое дерево. Движение при кодировании и декодировании идет справа налево.

[1]Данный способ кодирования был придуман в 1952 г. студентом Д. Хаффманом в процессе выполнения домашнего задания.

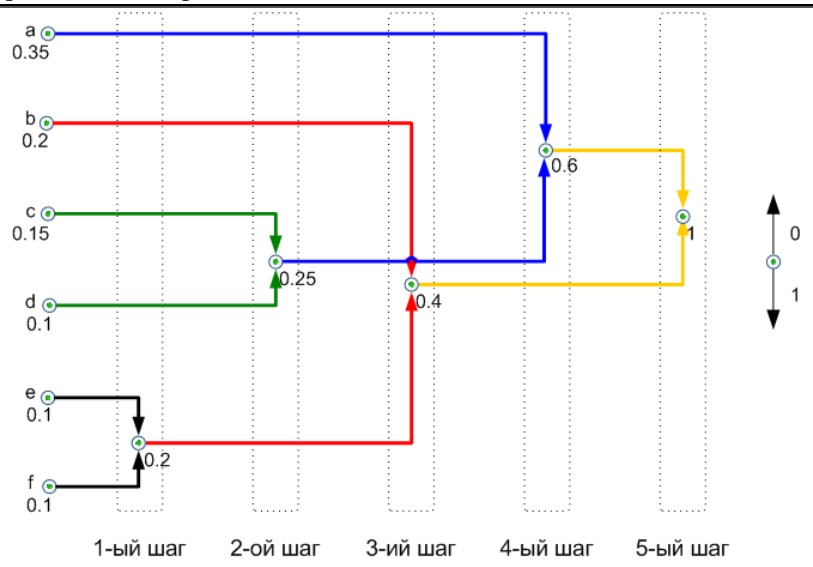

Рис. 6. Кодовое дерево Хаффмана

Таким образом:

буква	код
a	00
b	10
c	010
d	011
e	110
f	111

Найдены и кодовые слова, и вектор Крафта: $V(S)=(2,2,3,3,3,3)$. Проверим неравенство (2.1): $2\cdot 2^{-2}+4\cdot 2^{-3}=1$, значит найденный код оптимальный. Длина двоичной последовательности: $\bar{l}(Y)=2.45$. Эта длина не меньше энтропии.

Найдем избыточность данного неравномерного кода: $r=\bar{l}-H(X)=0.05$. Это показывает, что при данном кодировании тратится на *0.05* бит больше, чем могло бы быть затрачено в принципе при теоретически лучшем принципе кодирования. ●

Для декодирования выходной последовательности используется тоже двоичное кодовое дерево; если в потоке встречается 0-ой бит, то берется верхняя ветвь, иначе – нижняя. Так продолжается до тех пор, пока не получен лист дерева, соответствующий исходному символу последовательности.

Средняя длина кодовых слов для данного алгоритма: $\bar{l}(Y)\le H(X)+1$, что и было видно выше в примере.

2.2.4.2 Код Шеннона

Далее приведен алгоритм кодирования по Шеннону:

Инициализации. Все буквы из алфавита записываются по убыванию вероятностей встречаемости в сообщениях. Каждой букве ставится в соответствие кумулятивная вероятность по правилу: $q_1 = 0, q_2 = P_1, \ldots, q_M = \sum_{i=1}^{M-1} p_i$, где M – мощность алфавита.

Цикл. Кодовым словом для j-ого сообщения является двоичная последовательность, представляющая собой первые $l_j = \left\lceil -\log_2 P(x_j) \right\rceil$ разрядов[2] после запятой в двоичной записи числа q_j[3].

Так же, как и для предыдущего кода, средняя длина кодового слова: $\bar{l}(Y) \leq H(X) + 1$.

Пример

Рассмотрим тот же пример, что и выше.

Построим код Шеннона:

Буква x_i	$P(x_i)$	q_i	l_i	$\lceil q_i \rceil_2$	Код
a	0.35	0.0	$\lceil 1.5 \rceil = 2$	0.00…	00
b	0.2	0.35	$\lceil 2.3 \rceil = 3$	0. 0101100110011 00…	010
c	0.15	0.55	$\lceil 2.7 \rceil = 3$	0. 1000110011001 10…	100
d	0.1	0.7	$\lceil 3.3 \rceil = 4$	0. 1011001100110 01…	1011
e	0.1	0.8	$\lceil 3.3 \rceil = 4$	0. 1100110011001 10…	1100
f	0.1	0.9	$\lceil 3.3 \rceil = 4$	0. 1110011001100 11…	1110

Декодирование аналогично декодированию кода, составленного по методу Хаффмана: строится дерево и проход осуществляется по направлению от корня к листьям.

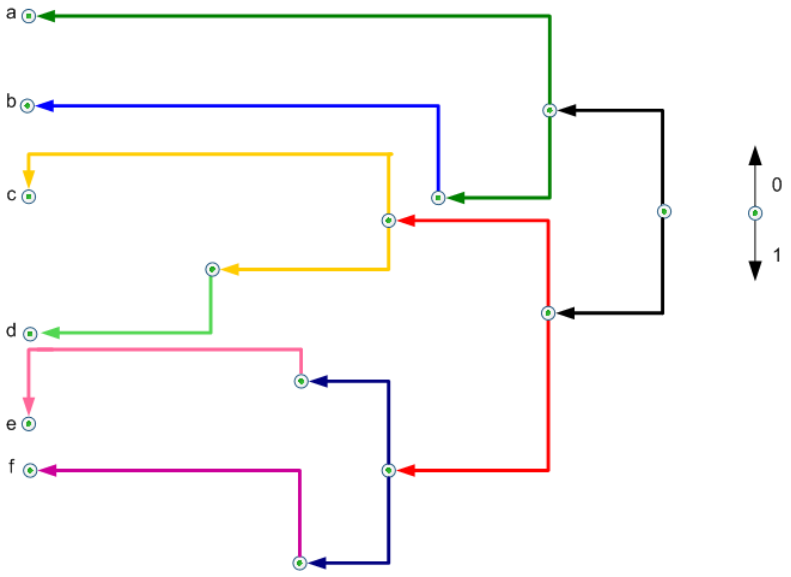

Рис. 7. Кодовое дерево Шеннона

[2] Наименьшее целое, большее или равное числу в скобках.

[3] Программный код, переводящий число $0 < x < 1$ из десятичной системы исчисления в двоичную, написанный в MathCad 11.0, приведен в приложении к методическим указаниям по выполнению лабораторных работ по данному курсу.

Средняя длина кодового слова $\bar{l}(Y) = 2.95$, и избыточность данного неравномерного кода: $r = \bar{l} - H(X) = 0.55$. Кроме того, неравенство Крафта имеет вид строгого неравенства. Это означает, что в целом данный код хуже кодирования Хаффмана. Однако построить кодирование Шеннона в вычислительном смысле легче.●

2.2.4.3 Код Шеннона-Фано

Ниже приведен алгоритм кодирования по методу Шеннона-Фано:

Инициализации. Все буквы алфавита записываются в порядке убывания вероятностей.

Цикл. Всю совокупность букв делят на две примерно равные по сумме вероятностей группы. Одной из них присваивают двоичный ноль, другой – двоичную единицу. Далее каждую группу вновь делят на две и повторяют цикл до тех пор, пока в группе не останется одна буква.

Пример

Рассмотрим тот же пример, что и выше.

Построим код Шеннона-Фано:

Буква x_i	$P(x_i)$	1-ый шаг	2-ой шаг	3-ий шаг		Код
a	0.35	1	1	stop		11
b	0.2		0	stop		10
c	0.15	0	1	1	stop	011
d	0.1			0	stop	010
e	0.1		0	1	stop	001
f	0.1			0	stop	000

Рис. 8. Код Шеннона-Фано

Заметим, что полученный код аналогичен коду Хаффмана для этого примера. В общем случае это не так.

Декодирование полученного кода аналогично декодированию, приведенному в предыдущих пунктах. ●

При построении кода Шеннона и Шеннона-Фано требуется упорядочить сообщения по убыванию вероятностей. Это, пожалуй, является наиболее трудоемкой частью алгоритмов. Следующий код не обладает этим недостатком.

2.2.4.4 Код Гильбера-Мура

Далее приведен алгоритм кодирования по методу Гильбера-Мура.

Инициализация. Сопоставим каждой букве кумулятивную вероятность $q_m = \sum_{i=1}^{m-1} p_i$ и вычислим $\sigma_m = q_m + \frac{p_m}{2}$.

Цикл. Кодовым словом являются первые $l_m = \left\lceil -\log_2 \frac{p_m}{2} \right\rceil$ разрядов после запятой в двоичной записи числа σ_m.

Средняя длина кодового слова: $\bar{l}(Y) \le H(X) + 2$.

Пример

Рассмотрим тот же пример. Только для наглядности изменим последовательность в ансамбле. Построим код Гильбера-Мура:

Буква x_i	$P(x_i)$	q_i	σ_i	l_i	$[\sigma_i]_2$	Код
e	0.1	0.0	0.05	$\lceil 4.3 \rceil = 5$	0.0000110011001100…	00001
c	0.15	0.1	0.175	$\lceil 3.7 \rceil = 4$	0.0010110011001100…	0010
b	0.2	0.25	0.35	$\lceil 3.3 \rceil = 4$	0.0101100110011001…	0101
d	0.1	0.45	0.5	$\lceil 4.3 \rceil = 5$	0.1000000000000000…	10000
a	0.35	0.55	0.725	$\lceil 2.5 \rceil = 3$	0.1011100110011001…	101
f	0.1	0.9	0.95	$\lceil 4.3 \rceil = 5$	0.1111001100110011…	11110

Средняя длина кодового слова $\bar{l}(Y) = 3.95$, и избыточность данного неравномерного кода: $r = \bar{l} - H(X) = 1.55$. Кроме того, неравенство Крафта имеет вид строгого неравенства. Это означает, что в целом данный код хуже всех рассмотренных для данного примера. ●

Все рассмотренные методы побуквенного кодирования обладают следующими недостатками:

1. Необходимо иметь таблицу вероятностей символов входного сообщения. В случае, когда вероятности неизвестны (что чаще всего бывает), методы работают неэффективно.
2. Минимальная длина кодового слова не может быть меньше одного бита, тогда как энтропия сообщения может быть *0.1(бит/буква)* или *0.01(бит/буква).* В этом случае кодирование становится неэффективным.
3. Средняя длина кодового слова будет равна энтропии тогда и только тогда, когда все вероятности сообщений $x \in X$ имеют вид: $P(x) = 2^{-l(x)}$, где $l(x)$ – целое положительное число.

Решение этих проблем:

1. Использовать адаптивные алгоритмы: дерево кода постоянно обновляется с прочтением очередного символа сообщения.
2. Использовать блоки символов, но тогда усложняется процесс кодирования/декодирования.

2.3 Помехоустойчивое кодирование

Помехоустойчивые коды – это коды, позволяющие обнаружить и (или) исправить ошибки в кодовых словах, которые возникают при передачи по каналам связи. Эти коды строятся таким образом, что для передачи сообщения используется лишь часть кодовых слов, которые отличаются

друг от друга более, чем в один символ. Эти кодовые слова называются разрешенными, все остальные не используются и называются запрещенными.

Теорема Шеннона (о кодировании в дискретных каналах с шумом).

Пусть пропускная способность канала $C = H_{\max}(A)$, скорость создания информации $V = H_{ист}(X)F_{ист}$, где $H_{ист}(X)$ – энтропия источника, $F_{ист}$ – частота следования символов источника.

1. Прямая теорема.

Если скорость создания информации V источником на входе зашумленного канала без памяти с пропускной способностью C такова, что $V < C$, то существует такой код, при котором вероятность ошибки на приемном конце сколь угодно мала, а скорость передачи информации сколь угодно близка к скорости её создания.

2. Обратная теорема.

Если $V > C$, то никакой код не может сделать вероятность ошибок на приемном конце сколь угодно малой и достигнуть надежности, меньшей, чем $V - C$, а скорость передачи информации сколь угодно близкой к скорости её создания. Основной способ повышения помехоустойчивости системы передачи информации – разумное введение избыточности в передаваемый сигнал A.

Помехозащитные коды делятся на *две группы*:

1. Код с исправлением ошибок имеет цель восстановить с вероятностью близкой к единице посланное сообщение.
2. Код с обнаружением ошибок имеет цель выявить с вероятностью близкой к единицы наличие ошибок.

Введем основные параметры помехоустойчивых кодов:

- Расстояние Хэмминга (кодовое расстояние между двумя кодовыми словами) d – число позиций, в которых два кодовых двоичных слова отличаются друг от друга.
- Кодовое расстояние кода d_0 – наименьшее расстояние Хэмминга между различными словами кода.

Считают, что в канале произошла q-кратная ошибка, если кодовое слово на выходе канала отличается от кодового слова на входе ровно в q символах (то есть $d = q$).

Основные зависимости:

- $d_0 \geq q_{обнар} + 1$ – код позволяет обнаружить любую ошибку кратности $q_{обнар}$, если его кодовое расстояние удовлетворяет этому условию.
- $d_0 \geq 2q_{исправ} + 1$ – код позволяет исправить любую ошибку кратности $q_{исправ}$, если его кодовое расстояние удовлетворяет этому условию.
- Граница Хэмминга: $r = n - k \geq \log_2 \sum_{i=0}^{q_{ист}} C_n^i$, где n – длина кодового слова, k – количество информационных разрядов, r – количество проверочных разрядов. Это выражение является нижней границей в том смысле, что оно устанавливает то минимальное соотношение корректирующих и информационных разрядов, ниже которого код не может сохранять заданные корректирующие способности.
- Граница Плоткина: $d_0 \leq n\frac{2^{k-1}}{2^k - 1}$.

Границы Хэмминга и Плоткина являются нижними границами для кодового расстояния при заданных n, k, задающими минимальную избыточность, при которой существует

помехоустойчивый код, имеющий минимальное кодовое расстояние и гарантированно исправляющий $q_{испр ав}$-кратные ошибки.

- Граница Варшамова-Гильберта: $2^{n-k} = 2^r > \sum_{i=0}^{d_0-2} C_{n-1}^i$ является нижней границей для числа проверочных разрядов r в случае кодов большой разрядности, необходимого для обеспечения заданного кодового расстояния d_0.

Среди всех корректирующих кодов наибольшее применение нашли систематические коды (далее будем рассматривать именно такие коды). Систематическим кодом (n,k) называется двоичный корректирующий код, удовлетворяющий следующим требованиям:

1. Все кодовые слова содержат n символов, из которых k информационных и r проверочных. Проверочные и информационные занимают определенные позиции в кодовом слове.
2. Сумма по модулю два ($\oplus$) любых кодовых слов снова дает кодовое слово, принадлежащее данному систематическому коду.

2.3.1 Коды с обнаружением ошибок

Рассмотрим простейший метод, принадлежащий к этой группе кодов и основанный на проверке на четность.

Схема кодирования: $x_{k+1} = \oplus\sum_{i=1}^{k} x_i$, то есть контрольный бит с номером $k+1$ дополняет остальные по четности ($x_{k+1} = 0$, если $\oplus\sum_{i=1}^{k} x_i$ четна; иначе – *1*).

Схема декодирования: $z = \oplus\sum_{i=1}^{k+1} y_i$, если z – четно, то есть равно нулю, то четность соблюдена и передача произошла без ошибок. Иначе – где-то произошла ошибка.

Для данного кода $d_0 = 2$, значит, кратность гарантированно распознанных ошибок $q_{обнар} = 1$. Однако достоинством этого кода является то, что он распознает все сочетания ошибок нечетной кратности (то есть не только $q_{обнар} = 1$, но и *3,5,7*, и т. д.).

Пример

Надо послать сообщение $x = 001001111$(0), контрольный бит $x_{11} = 1$. Посылаем сообщение: $x = 00100111101$. В результате вредительства сообщение исказилось и было принято: $y = 00100001001$. $z = 1$, то есть данному каналу связи больше доверять нельзя.●

2.3.2 Коды с исправлением ошибок

2.3.2.1 Линейные блоковые коды

Линейным блоковым (*n, k*) кодом называется множество N последовательностей длины n над полем Галуа $GF(q)$, называемых кодовыми словами. Это поле характеризуется тем, что сумма двух кодовых слов является кодовым словом и произведение любого кодового слова на элемент поля – также кодовое слово.

Если $q = 2$, то линейный код называется групповым, так как кодовые слова образуют математическую структуру, называемую группой. При формировании данного кода линейной операцией является сумма по модулю два ($\oplus$).

Далее рассмотрим один из способов задания линейных блоковых кодов – матричный, который основан на проверке ортогональности кодовых векторов проверочной матрице.

- Порождающая матрица: $G_{(n,k)} = \left| I_k, R_{k,r} \right|$, где I_k – единичная матрица, $R_{k,r}$ – матрица, составленная из проверочных символов.
- Проверочная матрица: $H_{(n,k)} = \left| R_{k,r}{}^T, I_r \right|$.
-

Основные свойства линейных кодов:

1. Если v_i – это кодовое слово, то: $v_i H_{(n,k)}{}^T = \begin{bmatrix} 0 & 0 & \dots & 0 \end{bmatrix}$.
2. Произведение некоторого кодового слова v_i' с ошибкой на $H_{(n,k)}{}^T$ даст синдром S_i: $v_i' H_{(n,k)}{}^T = S_i$.
3. $G_{(n,k)} H_{(n,k)}{}^T = 0$.
4. Кодовое расстояние d_0 (n,k)-кода равно минимальному числу линейно зависимых столбцов $H_{(n,k)}$.
5. Произведение информационного слова на $G_{(n,k)}$ даст кодовое слово.
6. Два кода называются эквивалентными, если их порождающие матрицы $G_{(n,k)}$ отличаются только перестановкой столбцов и элементарными операциями над строками.
7. Кодовое расстояние любого линейного (n,k)-кода удовлетворяет неравенству $d_0 \le n - k + 1$ (граница Сингтона). Если выполняется строгое равенство, то данный код называется кодом с максимальным расстоянием.
8. Граница Плоткина для минимального количества контрольных разрядов: $r \ge 2d_0 + 2 - \log_2 d_0$ при $n \ge 2d_0 - 1$.

Пример

Дана следующая порождающая матрица: $G_{(5,3)} = \left[\begin{array}{ccc|cc} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{array}\right]$.

Тогда кодирование задается отображениями: 000→00000, 001→00110, 010→01011, 011→01101, 100→10001, 101→10111, 110→11010, 111→11100.

Например:

$$\begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \cdot G_{(5,3)} = \left[0 \quad 1 \quad 1 \quad \underbrace{0 \cdot 0 \oplus 1 \cdot 1 \oplus 1 \cdot 1}_{0} \quad \underbrace{0 \cdot 1 \oplus 1 \cdot 1 \oplus 1 \cdot 0}_{1} \right] = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Проверочная матрица: $H_{(5,3)} = \left[\begin{array}{ccc|cc} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}\right]$.

Проверим свойство *1*: $\begin{bmatrix} 0 & 1 & 1 & 0 & 1 \end{bmatrix} \cdot H_{(5,3)}{}^T = \begin{bmatrix} 0 & 0 \end{bmatrix}$.

Проверим свойство *4*: линейно зависимые столбцы в $H_{(5,3)}$ – это *1*-ый и *5*-ый (например). Значит, $d_0 = 2$. Посмотрим на (*): видим, что минимальное кодовое расстояние кода действительно равно *2*.

Свойство *7* (граница Сингтона): $2 \le 5-3+1$.●

2.3.2.2 Коды Хэмминга

Код Хэмминга представляет собой один из важнейших классов линейных кодов, нашедших широкое применение на практике и имеющих простой и удобный для технической реализации алгоритм обнаружения и исправления одиночной ошибки. Кодом Хэмминга называется (n,k)-код с проверочной матрицей $H_{(n,k)}$, у которой $r=n-k$ строк и n столбцов, причем столбцы являются различными ненулевыми последовательностями.

Код Хэмминга, обеспечивающий исправление всех одиночных ошибок, должен иметь минимальное кодовое расстояние $d_{\min}=3$. При передаче кода может быть искажён любой символ (n комбинаций). Однако может быть и такой случай, когда ни один из символов не искажён. Тогда с помощью r контрольных символов должно быть создано такое количество комбинаций, чтобы различить $n+1$ вариант. Поэтому $2^r = n+1$. (Сравнить с границей Хэмминга).

Например, $(7,4)$-код Хэмминга : $H_{(7,4)} = \left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$.

Если столбцы проверочной матрицы представляют упорядоченную запись десятичных чисел, то есть *1,2,3,…* в двоичном формате, то вычисленный синдром $S_i(1,0) = S_{r-1}...S_1S_0 = v_i(1,0)H_{(n,k)}{}^T$ однозначно указывает на номер позиции искаженного символа.

Пример

Для $(7,4)$-код Хэмминга проверочная матрица в упорядоченном виде имеет вид: $H_{(7,4)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$. Пусть переданное слово $v = 1001100$, а принятое $v' = 1101100$.

Проверим, действительно ли есть ошибка (по свойству *1*):

$v'H_{(7,4)}{}^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$. Да, ошибка есть. Значит, вычисляем синдром.

Синдром, соответствующий принятому слову, будет равен: $S = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} = 2^1 = 2$
нулевой ↑ *разряд*.

Значит, во второй позиции v' произошла ошибка. Инвертируем второй символ в v'.

Порождающую матрицу $G_{(n,k)} = \left|I_k, R_{k,r}\right|$ легко получить, приведя матрицу $H_{(n,k)}$ к виду $H_{(n,k)} = \left|R_{k,r}{}^T, I_r\right|$ элементарными преобразованиями над строками:

$$H_{(7,4)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{matrix} \oplus 2-ая & строка \\ \oplus 3-ая & строка \rightarrow \end{matrix}$$

$$\rightarrow \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{matrix} & \\ \oplus 1-ая & строка \rightarrow \\ \oplus 1-ая & строка \end{matrix}$$

$$\rightarrow \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Теперь мы знаем $R_{k,r}{}^T$, а значит и $G_{(n,k)} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]$. ●

2.3.2.3 Циклические коды

В отличие от кода Хемминга в циклическом коде используют математические операции не над векторами, а над многочленами. (Напомним, что произвольному вектору $[x_1, x_2, ..., x_n]$ всегда можно поставить в соответствие многочлен вида $f(a) = x_1 a^{n-1} \oplus x_2 a^{n-2} \oplus \ldots \oplus x_n a^0$, где a – основание).

Циклический код – это линейный блоковый код (n,k), который характеризуется:

- свойством цикличности, то есть сдвиг влево на один шаг любого разрешенного кодового слова дает также разрешенное кодовое слово, принадлежащее этому же коду;
- множество кодовых слов представляет совокупность многочленов степени $n-1$ и менее, делящихся на некоторый многочлен $g(x)$ степени r и являющихся сомножителями двучлена $x^n + 1$. При этом $g(x)$ – это порождающий многочлен. Результатом деления $x^n + 1$ на $g(x)$ является проверочный многочлен $h(x)$.

Идея циклического кода состоит в том, что для информационных разрядов $x_1, x_2, ..., x_k$ избыточные разряды $x_{k+1}, ..., x_n$ подбираются таким образом, чтобы получившемуся вектору отвечал многочлен $f(a)$, нацело делящийся на проверочный полином $g(a)$. Если в результате передачи полученный $f^*(a)$ также делится на $g(a)$ без остатка, то считается, что передача была верной. Если же остаток $z(a)$ ненулевой, то это говорит об ошибке (в частном случае по виду остатка $z(a)$ ошибку можно локализовать и исправить).

Циклический код может быть задан порождающей и проверочной матрицами. Для их построения достаточно знать порождающий $g(x)$ и проверочный $h(x)$ многочлены. Для систематического циклического кода порождающая матрица строится, исходя из выражения: $G_{(n,k)} = \left|I_k, R_{k,r}\right|$, где строки матрицы $R_{k,r}$ определяются из выражения: $r_i(x) = R_{g(x)}\left[x^{n-i}\right]$.

Пример

Для $(7,4)$-кода на основе порождающего многочлена $g(x)=x^3+x+1$ строится следующая последовательность:

$r_1(x)=R_{g(x)}\left[x^6\right]=x^2+1$, так как:

$$
\begin{array}{l}
x^6 \quad \left|\begin{array}{l}\underline{x^3+x+1}\\ x^3+x+1\end{array}\right.\\
\underline{\oplus\, x^6+x^4+x^3}\\
x^4+x^2+x\\
\underline{\oplus\, x^4+x^2+x}\\
x^3+x^2+x\\
\underline{\oplus\, x^3+x+1}\\
x^2+1
\end{array}
$$

Аналогичным образом определяется $r_2(x)=R_{g(x)}\left[x^5\right]=x^2+x+1$, $r_3(x)=R_{g(x)}\left[x^4\right]=x^2+x$, $r_4(x)=R_{g(x)}\left[x^3\right]=x+1$.

Тогда $G_{(7,4)}=\left[\begin{array}{cccc|ccc}1&0&0&0&1&0&1\\0&1&0&0&1&1&1\\0&0&1&0&1&1&0\\0&0&0&1&0&1&1\end{array}\right]$, $H_{(7,4)}=\left[\begin{array}{cccc|ccc}1&1&1&0&1&0&0\\0&1&1&1&0&1&0\\1&1&0&1&0&0&1\end{array}\right]$.

Пусть передано кодовое слово 1110100 (1110→1110100), а принято – 1100100. Найдем синдром:

$$
\begin{array}{l}
x^6+x^5+x^2 \quad \left|\begin{array}{l}\underline{x^3+x+1}\\ x^3+x^2+x\end{array}\right.\\
\underline{\oplus\, x^6+x^4+x^3}\\
x^5+x^4+x^3+x^2\\
\underline{\oplus\, x^5+x^3+x^2}\\
x^4\\
\underline{\oplus\, x^4+x^2+x}\\
x^2+x
\end{array}
$$

Как же в этом случае исправить ошибку? Для этого необходимо знать соответствие вектора синдрома вектору e, описывающему ошибку. Соответствие вектора синдрома и вектора ошибки может быть вычислено следующим образом: $Y=X+e \Rightarrow S=Y\cdot H^T=(X+e)\cdot H^T=e\cdot H^T$. Это соответствие однозначно. Построим таблицу синдромов. По таблице получается, что $e=0010000$. Исправляем: $1100100 \oplus 0010000 = 1110010$. ●

Ошибка e	Синдром S
0000001	$1=1/g(x)$
0000010	$x=x/g(x)$
0000100	$x^2=x^2/g(x)$
0001000	$x+1=x^3/g(x)$
0010000	$x^2+x=x^4/g(x)$
0100000	$x^2+x+1=x^5/g(x)$
1000000	$x^2+1=x^6/g(x)$

Одна из основных задач, стоящих перед разработчиками устройств защиты от ошибок при передаче дискретных сообщений по каналам связи, является выбор порождающего многочлена $g(x)$ для построения циклического кода, обеспечивающего требуемое минимальное кодовое расстояние для гарантийного обнаружения и исправления q-кратных ошибок. Существуют специальные таблицы по выбору $g(x)$ в зависимости от предъявляемых требований к корректирующим возможностям кода.

Тема №3
Основы криптографии

3.1 Терминология и основные понятия криптологии

Проблемой защиты информации путем ее преобразования занимается криптология. Она делится на два раздела: криптография и криптоанализ.

3.1.1 Основные аспекты криптографии

Криптография – (до 70-ых годов) область научной и практической деятельности, связанной с разработкой, применением и анализом шифросистем; (в настоящее время) область науки, техники, практической деятельности, связанной с разработкой, применением и анализом криптографических систем защиты информации.

Основные функции криптографических систем:

- обеспечение конфиденциальности
- обеспечение аутентичности

различных аспектов информационного взаимодействия.

Источником угроз при решении криптографических задач считаются *преднамеренные* действия противника или *недобросовестного* участника информационного взаимодействия, а не случайное изменение информации вследствие помех, отказов и др.

Конфиденциальность – это защищенность информации от ознакомления с ее содержанием со стороны лиц, не имеющих права доступа к ней.

Аутентификация – это установление (то есть проверка и подтверждение) подлинности различных аспектов информационного взаимодействия: сеанса связи, сторон (идентификация), содержания (имитозащита), источника (установление авторства) передаваемых сообщений, времени взаимодействия и прочее. Особенно остро проблема аутентификации стоит в случае недоверия друг другу сторон, когда источник угрозы может быть не только третья сторона (противник), но и сторона, с которой осуществляется информационное взаимодействие.

Сфера интересов криптоанализа – наука о методах анализа криптографических отображений информации с целью раскрытия информации.

Следующий рисунок иллюстрирует определение криптографии и показывает основные составляющие части.

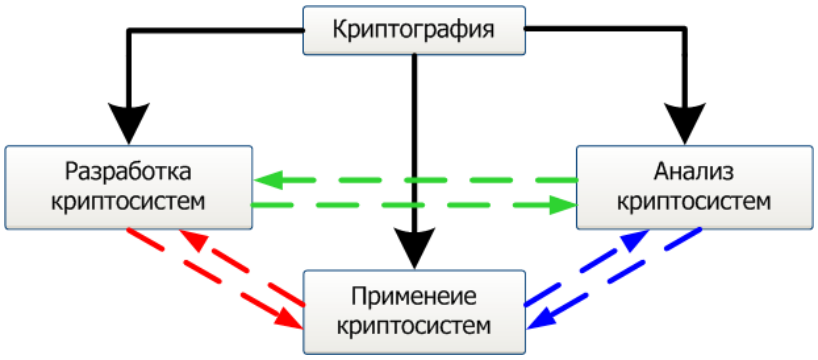

Рис. 9. Схема составляющих криптографии и их взаимодействие

Криптосистема – система обеспечения безопасности защищенной сети, использующая криптографические средства. В качестве подсистем она может включать в себя системы шифрова-

ния, идентификации, имитозащиты, цифровой подписи и др., а также ключевую систему, обеспечивающую работу остальных систем.

В основе выбора и построения криптосистемы лежит условие обеспечения криптографической стойкости. В зависимости от ключевой системы различают симметричные и асимметричные системы.

Криптографические средства – это:

- методы и средства обеспечения безопасности информации, использующие криптографические преобразования информации (то есть преобразование информации с использованием одного из криптографических алгоритмов, определяемой целевым назначением криптографической системы)
- (в более узком смысле) средства в виде документов, механических, электромеханических и электронных технических устройств или программ, предназначенных для выполнения функций криптографических систем.

Предположим, что отправитель хочет послать письмо получателю. Более того, этот отправитель желает, что при перехвате его сообщение никто не сможет прочесть. Само сообщение называется открытым текстом, изменение вида сообщения – шифрование, зашифрованное сообщение – шифротекст. Процесс преобразования шифротекста в открытый текст – дешифровка.

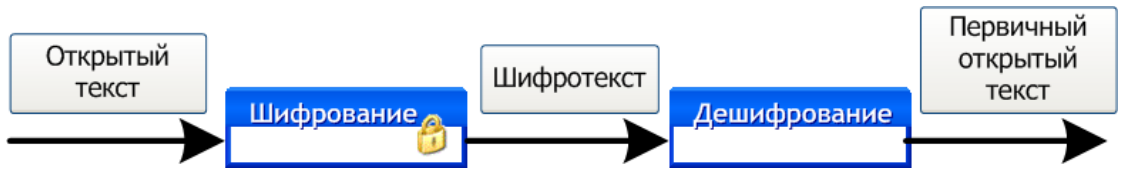

Рис. 10. Процесс шифрования/дешифрования

Обозначим:

- открытый текст M *(message).* Это может быть поток битов, текстовый файл и проч.
- Шифротекст C *(ciphertext).* $size(C) \geq size(M)$.
- Функция шифрования: $E(M) = C$.
- Функция дешифрования: $D(C) = M$.

Естественно, должно выполняться: $D(E(M)) = M$.

Криптографический алгоритм (шифр) представляет собой математические функции, используемые для шифрования/дешифрования.

Если безопасность алгоритма основана на сохранении самого алгоритма в тайне, то это ограниченный алгоритм. Такой алгоритм представляет только историческую ценность и абсолютно не соответствует современным стандартам. Большая и изменяемая группа пользователей не может использовать такие алгоритмы, так как при уходе одного из участников надо будет изменять алгоритм шифрования.

Современная криптография решает эту проблему с помощью ключа k. Такой ключ может иметь любое значение, выбранное из большого множества K. Множество возможных ключей – пространство ключей.

Таким образом, криптосистема представляет собой алгоритмы и всевозможные открытые тексты, шифротексты и ключи.

Если при шифровании и дешифровании используется один и тот же ключ, то это симметричное шифрование. Естественно, этот ключ должен храниться в секрете, поэтому второе название таких систем – системы с секретным ключом.

Для несимметричного шифрования (или алгоритмы с открытым ключом) существует два различных ключа: k_1 –открытый ключ, доступный всем и k_2 –закрытый, хранящийся только у отправителя сообщения; $E_{k_1}(M) = C$ и $D_{k_2}(C) = M$. Вычисление ключ k_2 дешифрования по k_1 связано с высокой сложностью решения.

Безопасности алгоритмов таких типов полностью основаны на ключах, а это значит, что данные алгоритмы шифрования могут быть опубликованы и проанализированы.

Преимущества симметричных алгоритмов заключаются в высокой скорости работы, меньших размерах используемых ключей и в более основательных гарантиях обеспечения криптостойкости.

Несимметричные криптосистемы используют более удобные протоколы.

Установленная последовательность действий, выполняемых для решения какой-либо криптографической задачи, называется криптографическим протоколом. В отличие от алгоритма, протокол реализуется в результате взаимодействия нескольких сторон протокола.

Одной из важнейших криптографических задач является задача распределения ключей. На рис. 11 представлено разделение данных протоколов по видам.

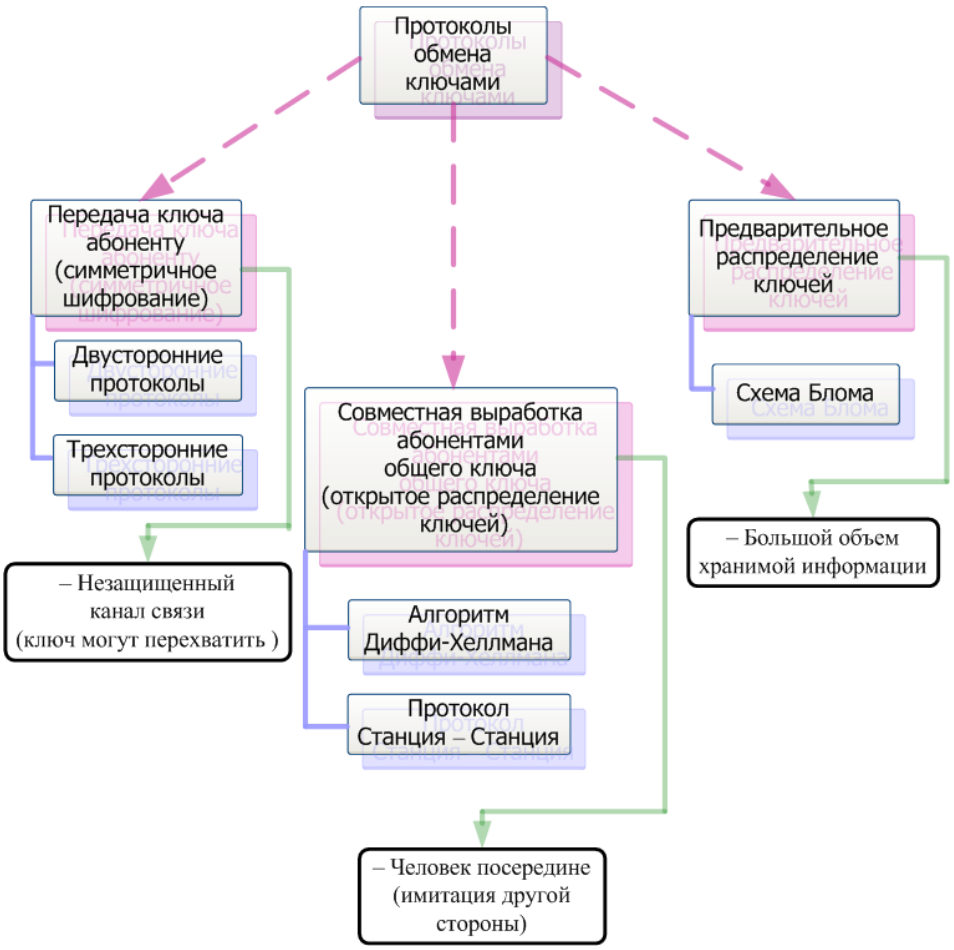

Рис. 11. Виды протоколов обмена ключами

3.1.2 Основные аспекты криптоанализа

Попытка криптоанализа называется вскрытием. *Основное предположение криптоанализа*, впервые сформулированное в XIX веке Д. А. Керкхофсом, состоит в том, что безопасность полностью определяется ключом. Он предполагал, что у криптоаналитика есть полное описание алгоритма шифрования и его реализация.

Существуют несколько *типов криптоаналитического вскрытия*:

1. Вскрытие только с использование шифротекста: у криптоаналитика есть несколько зашифрованных одним и тем же алгоритмом сообщений.
Дано: $C_1 = E_k(M_1),...,C_i = E_k(M_i)$.
Найти: либо $M_1 \div M_i, k$; либо алгоритм, как получить M_{i+1} из $C_{i+1} = E_k(M_{i+1})$.
2. Вскрытие с использованием открытого текста.
Дано: $M_1, C_1 = E_k(M_1),...,M_i, C_i = E_k(M_i)$.
Найти: либо k либо алгоритм, как получить M_{i+1} из $C_{i+1} = E_k(M_{i+1})$.
3. Вскрытие с использованием выбранного открытого текста.
Дано: $M_1, C_1 = E_k(M_1),...,M_i, C_i = E_k(M_i)$, при этом $M_1 \div M_i$ можно выбирать любыми.
Найти: либо k либо алгоритм, как получить M_{i+1} из $C_{i+1} = E_k(M_{i+1})$.
4. Адаптивное вскрытие с использованием открытого текста (в отличие от *2*-ого пункта, где выбирается один большой блок открытого текста, криптоаналитик в данной ситуации может выбрать меньший блок, затем второй, используя результаты шифрования первого блока и т. д.).
5. Вскрытие с использование выбранного шифротекста (обычно применяется к алгоритмам с открытым ключом).
Дано: $C_1, M_1 = D_k(C_1),...,C_i, M_i = D_k(C_i)$, при этом $C_1 \div C_i$ можно выбирать любыми.
Найти: k
6. Вскрытие с использование выбранного ключа (у криптоаналитика имеется некоторая информация о связи между различными ключами).
7. Бандитский криптоанализ (угрозы, шантаж, и т. д. Это наилучший путь взлома криптосистемы).

Различные алгоритмы предоставляют различную степень безопасности в зависимости от того, насколько трудно взломать алгоритм. Ларс Кнудсен предложил следующие деление на категории:

1. Полное вскрытие: находится k такой, что $D_k(C) = M$.
2. Глобальная дедукция: получается такой альтернативный алгоритм, что он эквивалентен $D_k(C)$ без знания k.
3. Локальная дедукция: получается M для перехваченного C.
4. Информационная дедукция: получается некоторая информация о k или M (несколько бит ключа, сведения о форме открытого текста и т. д.).

Алгоритм шифрования безусловно безопасный, если независимо от объема имеющихся у криптоаналитика шифротекстов, информации для получения открытого текста недостаточно.

3.2 Шеноновские модели криптографии

На рис. 12 представлена общая схема симметричной криптосистемы.

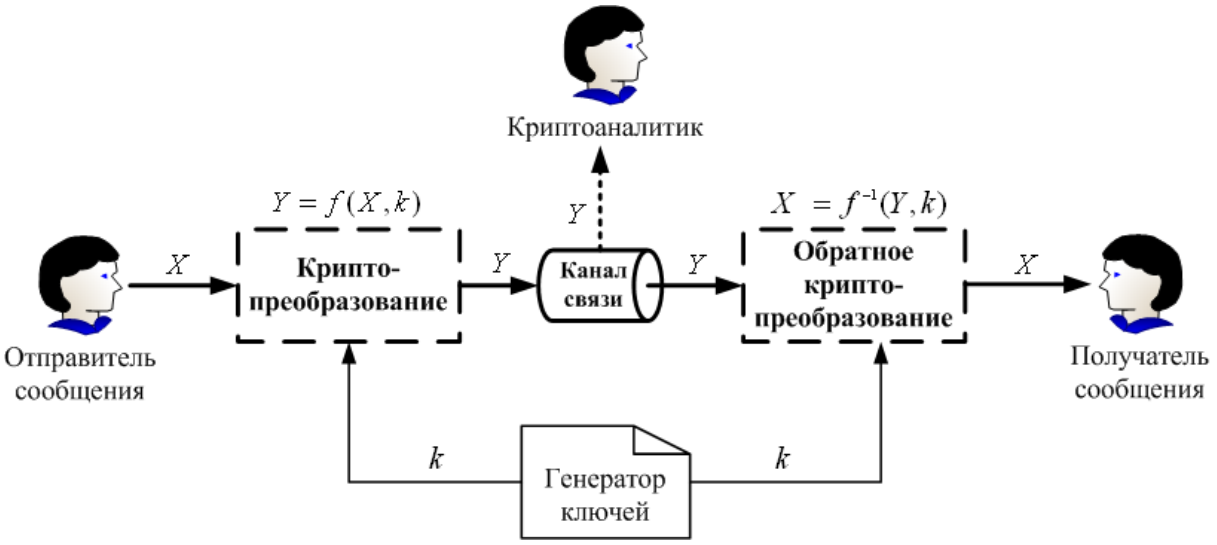

Рис. 12. Схема симметричной криптосистемы

Далее рассмотри *математические модели элементарных криптосистем*:

1. Подстановка: имеется некоторое открытое сообщение $[x_1,...,x_n]$ в алфавите A; система отображений $\Phi = \{\varphi_1,...,\varphi_n\}$ символов алфавита A в алфавит шифрованного текста B. Тогда зашифрованный текст: $Y = [\varphi_1(x_1),...,\varphi_n(x_n)]$.

Если φ_i – биективное отображение, то система Φ является подстановочным шифром замены, иначе – многозначным шифром замены.

2. Перестановка: имеется некоторое открытое сообщение $[x_1,...,x_n]$; перестановка $\Psi = \{\psi_1,...,\psi_n\}$. Тогда зашифрованный текст: $Y = [x_{\psi_1},...,x_{\psi_n}]$.

Недостатки этой модели:

- часто встречаемые символы в X и Y совпадают;
- ограниченность размера ключа, что приводит к его многократному использованию;

Для усиления криптостойкости данного криптопреобразования рекомендуется использовать композицию нескольких перестановок с различной длиной блоков.

3. Шифр Виженера и его модификация: имеется некоторое открытое сообщение $[x_1,...,x_n]$; шифрованное сообщение получается путем преобразования: $y_t = (x_t + k^\tau) \bmod |X|$, где $t = (i-1)\cdot T + \tau$, $i = 1,2...$; $\tau = \overline{1,T}$, где T – длина блока. Ключ k представляет собой фиксированный набор символов алфавита открытого сообщения.

Обратное преобразование: $x_t = (y_t - k^\tau + |X|) \bmod |X|$.

Частные случай: шифр Цезаря, Бофора.

4. Криптопреобразование Вернама (поточный шифр): это частный случай шифра Виженера, когда $T = n$. Ключ k в данном случае называется бегущей строкой, одноразовым блокнотом или гаммой.

5.

6. Биграммная подстановка: эта подстановка использует тот же принцип, что и простая подстановка, но вместо подстановки один символ – один символ, делают одновременную подстановку m символов – m символов.

3.3 Теоретико-информационные оценки стойкости симметричных криптосистем

В этом пункте мы будем исследовать общие вопросы устойчивости симметричных криптосистем к криптоанализу, используя теоретико-информационный подход Шеннона.

Свойство устойчивости криптосистемы к криптоанализу принято называть криптостойкостью.

Исходное сообщение X К. Шеннон предполагал случайным n - вектором с дискретным распределением вероятностей: $P\{X = X^{(i)}\} = q_i, i = \overline{1,\nu^n}, \sum_i q_i = 1$, где $X^{(i)} \in A_\nu^n$ – i-ый возможный вариант исходного n-символьного сообщения из алфавита A_ν, ν – основание алфавита исходного сообщения. Ключ $k = (k_1,..., k_m)$ также генерируется случайным вектором, не зависящим от X, с дискретным распределением вероятностей: $P\{k = k^{(j)}\} = p_j, j = \overline{1,\mu^m}, \sum_j p_j = 1$, где $k^{(j)} \in A_\mu^m$ – j-ый возможный вариант ключевой m-символьной последовательности в алфавите A_μ, μ – основание алфавита зашифрованного сообщения; p_j – априорная вероятность ключа $k^{(j)}$.

Симметричная криптосистема называется **совершенно криптостойкой**, если апостериорное распределение вероятностей исходного случайного сообщения X при регистрации случайного шифротекста $Y = f(X;k)$ совпадает с априорным распределением вероятностей: $P\{X = X^{(i)} \mid Y = Y^{(l)}\} = P\{X = X^{(i)}\} = q_i, \quad i,l = \overline{1,\nu^n}$.

Смысл этого условия в том, что хотя криптоаналитик и имеет шифротекст, он не добавляет ему информации о переданном сообщении.

Теорема 3.1

Необходимое и достаточное условие совершенной криптостойкости состоит в том, что условное распределение вероятностей шифротекста $Y \in A_\nu^n$ при фиксированном сообщении $X \in A_\nu^n$ не зависит от X:

$$P\{Y = Y^{(l)} \mid X = X^{(i)}\} = P\{Y = Y^{(l)}\}, \quad i,l = \overline{1,\nu^n}.$$

Следствие

Если выполняется условие совершенной криптостойкости, то количество информации по Шеннону, содержащейся в шифротексте Y об исходном сообщении X, равно нулю: $I\{X,Y\} = I\{Y,X\} = 0$.

Теорема 3.2 (Пессимистическое утверждение Шеннона)

Необходимым условием выполнения свойства совершенной криптостойкости является справедливость следующих неравенств энтропии:

$$\begin{aligned} H(k) &\geq H(X), \\ H(k) &\geq H(Y). \end{aligned} \qquad \textbf{(3.1)}$$

Поскольку распределение $\{q_i\}$ исходного текста X может быть произвольным, $H_{\max}(X) = \log \nu^n$, то неравенство (1) примет вид: $H(k) \geq \log \nu^n$. Для его выполнения в случае

$\nu = \mu$ требуется, чтобы длина ключа m была не меньше длины шифруемого текста: $m \geq n$. Для практики наиболее интересен случай самых коротких ключей: $m = n$.

Теорема 3.3

Если $\nu = \mu$, $m = n$ и $\forall i,l \in \{1,2,...,\nu^n\}$ уравнение $f(X^{(i)};k^{(j)}) = Y^{(l)}$ имеет единственное решение $j = j_{il}$, то необходимым и достаточным условием совершенной криптостойкости является равновероятность используемых ключей: $p_j = \frac{1}{\nu^n}$, $j = \overline{1,\nu^n}$.

Следствие

Криптопреобразование Вернама при условии равновероятности ключей обладает свойством совершенной криптостойкости.

Данное следствие объясняет почему:

- для передачи и защиты наиболее важной информации широко используются поточные криптосистемы, базирующиеся на криптопреобразовании Вернама;
- к качеству генерации ключевой последовательности $k = (k_1,..., k_n)$ (гаммы) предъявляются столь высокие требования.

Шифр называется **идеально стойким**, если невозможно определить однозначно открытый текст при известном шифрованном тексте сколь угодно большой длины (совершенно стойкий шифр является идеально стойким).

Если ключ является фрагментом случайной двоичной последовательности с равномерным законом распределения, причем его длина равна длине исходного сообщения и используется этот ключ только один раз, после чего уничтожается, то такой шифр является **абсолютно стойким**, даже если криптоаналитик располагает неограниченным ресурсом времени и неограниченным набором вычислительных ресурсов. Таким образом, необходимым и достаточным условием абсолютной стойкости шифра является:

- полная случайность ключа;
- равенство длин ключа и открытого текста;
- однократное использование ключа.

К. Шеннон теоретически показал существование криптосистемы, обладающей совершенной безопасностью (в которых шифр не дает никакой информации об открытом сообщении, кроме, возможно, его длины). Это возможно только если число возможных ключей также велико, как число возможных сообщений (то есть ключ должен быть не короче открытого сообщения и не должен использоваться дважды).

Далее перечислим некоторые *показатели криптостойкости*:

1. Количество всевозможных ключей;
2. Среднее время, необходимое для криптоанализа;
3. Количество и качество (достоверность) всех перехваченных криптограмм;
4. Шифросистема имеет высокую криптографическую стойкость, если она выражается через длину ключа экспоненциально;
5. Количество материала, который необходимо проанализировать для вскрытия шифра (например, полный перебор ключей);
6. Стоимость дешифрирования системы (разработка нового вычислительного комплекса).

Для современных криптографических систем защиты информации сформулированы следующие общепринятые *требования*:

- зашифрованное сообщение должно поддаваться чтению только при наличии ключа;

- число операций, необходимых для определения использованного ключа шифрования по фрагменту шифрованного сообщения и соответствующего ему открытого текста, должно быть не меньше общего числа возможных ключей;
- число операций, необходимых для расшифровывания информации путем перебора всевозможных ключей должно иметь строгую нижнюю оценку и выходить за пределы возможностей современных компьютеров (с учетом возможности использования сетевых вычислений);
- знание алгоритма шифрования не должно влиять на надежность защиты;
- незначительное изменение ключа должно приводить к существенному изменению вида зашифрованного сообщения даже при использовании одного и того же ключа;
- структурные элементы алгоритма шифрования должны быть неизменными;
- дополнительные биты, вводимые в сообщение в процессе шифрования, должен быть полностью и надежно скрыты в шифрованном тексте;
- длина шифрованного текста должна быть равна длине исходного текста;
- не должно быть простых и легко устанавливаемых зависимостью между ключами, последовательно используемыми в процессе шифрования;
- любой ключ из множества возможных должен обеспечивать надежную защиту информации;
- алгоритм должен допускать как программную, так и аппаратную реализацию, при этом изменение длины ключа не должно вести к качественному ухудшению алгоритма шифрования.

Тема №4
Математические основы криптологии

4.1. Псевдослучайные последовательности

Случайные числа и их генераторы являются неотъемлемыми элементами современных криптосистем. Ниже приводятся конкретные примеры использования случайных чисел в криптологии:

1. Сеансовые и другие ключи для симметричных криптосистем (DES, ГОСТ 28147-89, Blowfish).
2. Стартовые значения для программ генерации ряда математических величин в асимметричных криптосистемах (например, «больших простых чисел» в криптосистемах RSA, ElGamal).
3. Случайные слова, комбинируемые с парольными словами для нарушения «атаки угадывания» пароля криптоаналитиком.
4. Вектор инициализации для блочных криптосистем, работающих в режиме обратной связи.
5. Случайные значения параметров для многих систем электронной цифровой подписи (например, DSA или белорусского стандарта СТБ 1176.2—99).
6. Случайные выборы в протоколах аутентификации (например, в протоколе Цербер (Kerberos)).
7. Случайные параметры протоколов для обеспечения уникальности различных реализаций одного и того же протокола (например, в протоколах SET и SSL).

Отметим, что для некоторых из этих криптографических применений необходимы огромные массивы случайных чисел, которые по своему назначению требуют конфиденциального использования. Проблема генерации случайной последовательности с произвольным законом распределения вероятностей сводится к проблеме генерации равномерно распределенной случайной последовательности.

4.1.1. Равномерно распределенная случайная последовательность

Равномерно распределенная случайная последовательность (РРСП) (или «чисто случайной» последовательность) – это случайная последовательность $x_1, x_2, ..., x_t, x_{t+1}...$ со значениями в дискретном множестве $A = \{0, ..., N-1\}$, определенная на вероятностном пространстве (Ω, F, P) и удовлетворяющая двум свойствам:

1. $\forall n \in N$ и произвольных значений индексов $1 \le t_1 < ... < t_n$ случайные величины $x_{t_1}, ..., x_{t_n}$ независимы в совокупности.
2. $\forall t \in N$ случайная величина x_t, имеет дискретное равномерное распределение вероятностей на A: $P\{x_t = i\} = \frac{1}{N}, i \in A$.

Из этих базовых свойств вытекают следующие дополнительные свойства, используемые при генерации случайных чисел:

3. Если $\{x_t\}$ – РРСП, то $\forall n \in N$ и любой фиксированной последовательности индексов $1 \le t_1 < ... < t_n$ n-мерное дискретное распределение вероятностей вектора (слова) $(x_{t_1}, ..., x_{t_n})$ является равномерным: $P\{x_{t_1} = i_1, ..., x_{t_n} = i_n\} = \frac{1}{N^n}; i_1, ..., i_n \in A$.

4. Воспроизводимость при прореживании: для любой фиксированной последовательности моментов времени $1 \le t_1 < ... < t_n$ при «прореживании» РРСП $\{x_t\}$ возникает подпоследовательность $y_1 = x_{t_1}, ..., y_n = x_{t_n}, ...$, которая также является РРСП.

5. Воспроизводимость при суммировании: если $\{x_t\}$ – РРСП, $\{\xi_t\} \in A$ – произвольная неслучайная либо случайная последовательность, не зависящая от $\{x_t\}$, то случайная последовательность $y_t = (x_t + \xi_t) \bmod N$ также является РРСП.

6. Если $\{x_t\}$ – РРСП, то $\forall n \in N$ количество информации по Шеннону, содержащейся в отрезке последовательности $X_n = (x_1, ..., x_n) \in A^n$, о будущем элементе x_{n+1} равно нулю: $I(x_{n+1}, X_n) = 0$. Поэтому для любого алгоритма прогнозирования $\tilde{x}_{n+1} = f(X_n)$ вероятность ошибки прогнозирования не может быть меньше, чем для «угадывания по жребию»: $P\{\tilde{x}_{n+1} \ne x_{n+1}\} \ge \frac{1}{N}$.

Определим понятие генератора случайной последовательности: генератор РРСП – это устройство, позволяющее по запросу получить реализацию равномерно распределенной случайной последовательности $x_1, ..., x_n \in A$ длиной $n \in A$; элементы $x_1, ..., x_n$ этой реализации принято называть случайными числами.

Существует три типа генераторов РРСП:

- табличный;
- физический;
- программный.

Мы будем рассматривать программные генераторы. Программный генератор РРСП – программа имитации на компьютере реализации РРСП.

Имитируемая последовательность $\{x_t\}$ называется псевдослучайной, так как она вычисляется по известному детерминированному соотношению, и в то же время ее статистические свойства близки (по определенным критериям) к свойствам РРСП.

4.1.2. Алгоритмы генерации псевдослучайных последовательностей

На рис. 13 приведена классификация алгоритмов генерации псевдослучайных последовательностей.

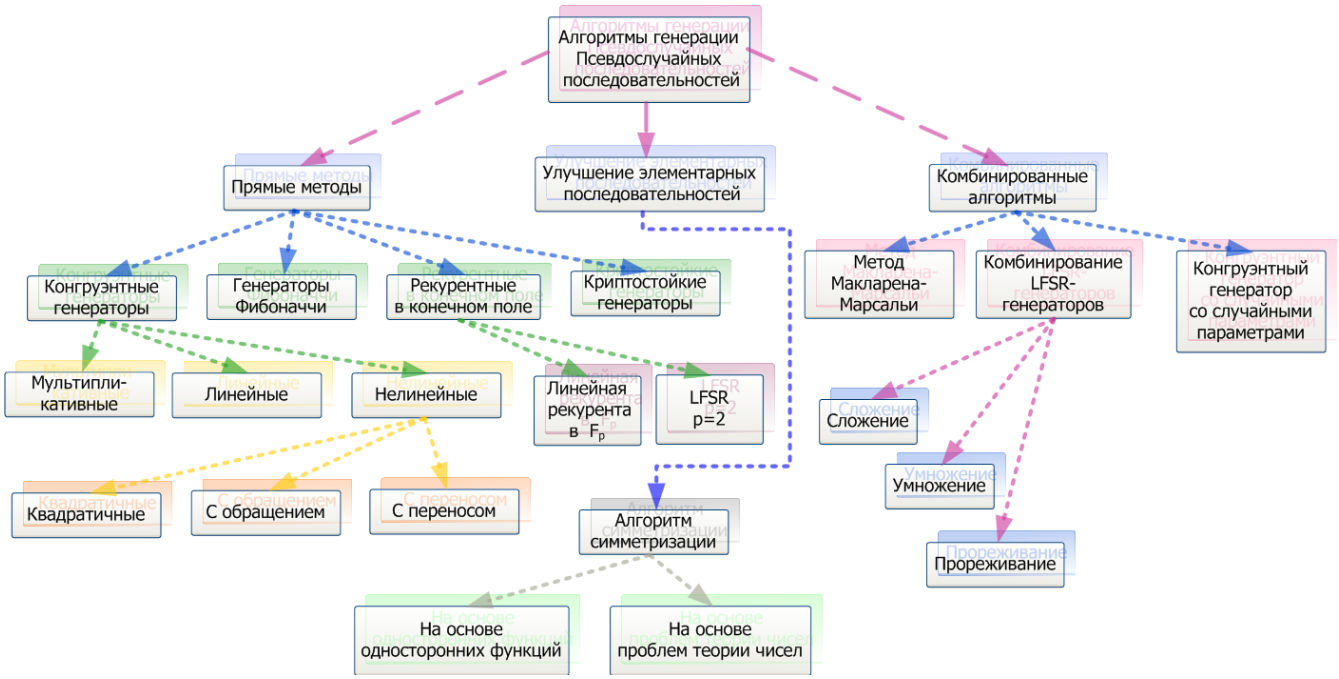

Рис. 13. Классификация алгоритмов генерации псевдослучайных последовательностей.

Выделяются три основных подхода к построению алгоритмов генерации:

1. Прямые методы построения элементарных рекуррентных последовательностей.
2. Методы «улучшения элементарных последовательностей», заключающиеся в специальных функциональных преобразованиях этих последовательностей для уменьшения отклонения их статистических свойств от свойств РРСП.
3. Комбинирование алгоритмов генерации, построенных с помощью первого или второго подхода.

Различные варианты генераторов рассматриваются в лабораторной работе номер 5.

4.2. Теория чисел

4.2.1. Простые числа

Натуральное число $p > 1$ называют простым, если оно не имеет других натуральных делителей, кроме 1 и p.

Простым числом будет наименьший, отличный от 1 делитель целого $b > 1$.

Теорема 4.1 (Теорема Евклида)
Существует бесконечно много простых чисел.

Свойства простого числа p:

1. Если $НОД(p,a) \neq 1$, то p делит нацело a;
   Если $НОД(p,a) = 1$, то p и a взаимно простые числа.
   Критерий взаимной простоты чисел: два целых числа a, b будут взаимно простые тогда и только тогда, когда $\exists u, v$ (целые) такие, что: $a \cdot u + b \cdot v = 1$ (например, $a = 17, b = 14$: $\exists u = 5, v = -6$ $17 \cdot 5 - 14 \cdot 6 = 1$, действительно: $НОД(17,14) = 1$).
2. Если p делит нацело $a \cdot b$, то p делит нацело a или p делит нацело b.

3. Всякое целое число $a>1$ разложимо в произведение простых множителей. Это разложение единственно с точностью до порядка следования множителей: $a=\prod_{p_i-простое\ \ число} p_i^{\alpha_i}$.

Криптография, особенно криптография с открытыми ключами, часто использует простые числа (длиной *512* бит и более; существует приблизительно 10^{151} простых чисел длиной $[1;512]$ бит).

Если обозначить $\pi(x)$ – число простых чисел, не превосходящих положительное x, то $\lim_{x\to\infty}\frac{\pi(x)\ln(x)}{x}\to 1$ (теорема о простых числах).

4.2.2. Тестирование чисел на простоту и построение больших простых чисел

Зачастую криптостойкость алгоритмов шифрования с открытыми ключом существенно зависит от того, насколько велики простые числа.

Общая схема для большинства методов получения простых чисел состоит из следующих двух шагов:

1. Выбор большого нечетного числа.
2. Тестирование этого числа на простоту.

Если оно оказалось простым, то алгоритм заканчивает работу. Иначе перейти на шаг 1.

Существует два подхода к реализации второго шага:

- Вероятностный тест (проверка простоты): с вероятностью *1* можно определить – является ли число составным; и только с вероятностью близкой к *1* можно определить случай простого числа.

  Общая схема вероятностного подхода состоит в проведении некоторого эвристического теста над числом – кандидатом. С каждой новой итерацией теста увеличивается вероятность того, что число простое.
- Детерминированный тест (построение больших простых чисел): результатом данного теста является доказуемое высказывание о том, что тестируемое число простое (составное).

Различные варианты тестов на простоту рассматриваются в лабораторной работе номер 6.

4.2.3. Теория сравнения

4.2.3.1. Арифметика вычетов

Будем рассматривать целые числа в связи с остатками от деления на натуральное число m, называемое модулем.

$$a\equiv b(\mathrm{mod}\,m), \qquad \textbf{(4.1)}$$

где $a=b+km$ для некоторого целого k.

Иногда b называют вычетом a по модулю m, a – конгруэнтным b по модулю m.

Формулу (4.1) также можно интерпретировать следующим образом: если два числа имеют одинаковые остатки от деления на m, то они сравнимы по модулю m (например, $8\equiv 5(mod\ 3)$).

В отличие от (4.1) операция $c \bmod t$ означает остаток от деления $\frac{c}{t} \in [0, m-1]$ (например, $5 \bmod 3 = 2$). Эта операция называется приведением по модулю.

Далее приведены *свойства функции сравнения*:

1. Сравнения по одинаковому модулю можно почленно складывать: $a_1 \equiv b_1 (\bmod m), a_2 \equiv b_2 (\bmod m) \Rightarrow a_1 + a_2 \equiv (b_1 + b_2) (\bmod m)$.
2. Слагаемое, стоящее в какой-либо части сравнения, можно переносить в другую часть, изменив его знак на обратный: $a + b \equiv c (\bmod m) \Rightarrow a \equiv (c - b) (\bmod m)$.
3. К любой части сравнения можно прибавить любое число, кратное модулю: $a \equiv b (\bmod m) \Rightarrow a + mk \equiv b (\bmod m)$.
4. Сравнения по одинаковому модулю можно почленно перемножать: $a_1 \equiv b_1 (\bmod m), a_2 \equiv b_2 (\bmod m) \Rightarrow a_1 a_2 \equiv (b_1 b_2) (\bmod m)$.
5. Обе части сравнения можно возвести в одну и ту же степень.
6. Как следствие из вышеперечисленных свойств, получаем: $a_0 \equiv b_0 (\bmod m), a_1 \equiv b_1 (\bmod m), \ldots, a_n \equiv b_n (\bmod m), \quad x \equiv y (\bmod m) \Rightarrow$ $a_0 x^n + a_1 x^{n-1} + \ldots + a_n = (b_0 y^n + b_1 y^{n-1} + \ldots + b_n) (\bmod m)$
7. Обе части сравнения можно разделить на их общий делитель, взаимно простой с модулем.
8. Обе части сравнения и его модуль можно умножить на одно и то же целое число или разделить на их общий делитель.
9. Если сравнение $a \equiv b$ имеет место по нескольким разным модулям, то оно имеет место и по модулю, равному наименьшему общему кратному этих модулей.
10. Если сравнение имеет место по модулю m, то оно имеет место и по модулю d, равному любому делителю числа m.
11. Если одна часть сравнения и модуль делятся на некоторое число, то и другая часть сравнения должна делиться на то же число.
12. К обеим частям сравнения можно прибавить одно и тоже число; обе части сравнения можно умножить на одно и тоже число, взаимно простое с с модулем.

Свойство операции приведения по модулю: $(a \lozenge b) \bmod m = [(a \bmod m) \lozenge (b \bmod m)] \bmod m$, где $\lozenge = \{+, -, *\}$.

Пример

а) Проверим *7*-ое свойство функции сравнения: $24 \equiv 6 (mod\ 9)$, общие делители *24* и *6 – 2,3*. Взаимно простое с *9* – это *2*: $НОД(2,9) = 1$. Тогда делим на *2* обе части сравнения: $12 \equiv 3 (mod\ 9)$, иначе было бы неверно: $8 \overline{\equiv} 2 (\bmod 9)$.

b) Проверим свойство операции приведения: $(24-5) \bmod 9 = \{19 \bmod 9 = 1\} = ((24 \bmod 9) - (5 \bmod 9)) \bmod 9 = (6-5) \bmod 9$ $= 1 \bmod 9 = 1 \bullet$

Пример

Найдем $a^8 \bmod n$ при помощи свойства операции приведения по модулю: $a^8 \bmod n = (a^4 \bmod n \cdot a^4 \bmod n) \bmod n =$ $= [(a^2 \bmod n \cdot a^2 \bmod n) \bmod n \cdot (a^2 \bmod n \cdot a^2 \bmod n) \bmod n] \bmod n =$ $= [(a^2 \bmod n)^2 \bmod n]^2 \bmod n$

Такой прием называется цепочкой разложения, в основе которой лежит двоичное представление числа[4]. ●

Числа, сравнимые по модулю m, образуют класс вычетов по модулю m. Все числа из одного и того же класса имеют один и тот же остаток r от деления на m. Любое число a из класса вычетов называется вычетом по модулю m.

Всего имеется m классов вычетов по модулю m (если классы обозначать как $\bar{a}$): $\bar{0}, \bar{1}, \ldots, \overline{m-1}$.

Свойства классов вычетов:

1. $a \equiv b (\mathrm{mod}\, m) \Leftrightarrow \bar{a} = \bar{b}$.
2. $a \not\equiv b (\mathrm{mod}\, m) \Leftrightarrow \bar{a} \cap \bar{b} = \varnothing$.
3. Любые m чисел попарно не сравнимые по модулю m образуют полную систему вычетов.
4. Если $НОД(a,m) = 1$ и x «пробегает» полную систему вычетов по модулю m, то $ax + b$, где b – любое число, также пробегает полную систему вычетов по модулю m.

Полная система вычетов – это совокупность вычетов, взятых по одному из каждого класса эквивалентности.

Приведенная система вычетов по модулю m – это совокупность всех вычетов из полной системы, взаимно простых с модулем m.

Пример

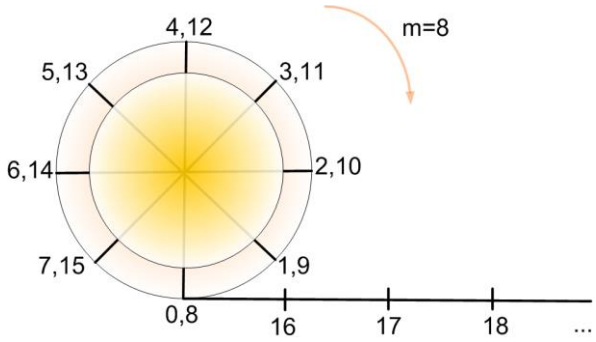

Рис. 14. Иллюстрация к образованию классов вычетов по модулю *8*

На рис. 14 изображен процесс «наматывания» цепочки целых чисел на колечко с $m = 8$ делениями, при этом на одно деление автоматически попадают сравнимые между собой числа.

Существуют *8* классов вычетов по модулю $m = 8$. В каждом классе находятся числа, дающие при делении на m одинаковые остатки:

Члены класса	0,8,16, …	1,9,17, …	2,10,15, …	3,11,19, …	4,12,20, …	5,13,21, …	6,14,22, …	7,15,23, …
Название класса	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$	$\bar{6}$	$\bar{7}$

Например, одна из полных систем вычетов по модулю *8*: *1200, 33, 42, 3, 12, 629, 22, 807.*
Приведенная система вычетов: *0, 1, 3, 5, 7.* ●

[4] Данная функция приведена в приложении к лабораторным работам на языке С.

4.2.3.2. Функция Эйлера

Количество классов вычетов в приведенной системе вычетов минус один обозначают как $\varphi(m)$ и называют функцией Эйлера (или представляет собой количество чисел ряда $1,2,...,m-1$ взаимно простых с m). Например, $\varphi(2)=1$, $\varphi(3)=2$, $\varphi(4)=2$, $\varphi(5)=4$, $\varphi(6)=2$, $\varphi(7)=6$, $\varphi(8)=4$. Кроме того, $\varphi(1)=1$.

Свойства функции Эйлера:

1. Если p – простое, то $\varphi(p)=p-1$.
2. Если p – простое, то $\varphi(p^k)=p^k-p^{k-1}$.
3. Мультипликативность функции Эйлера: если $НОД(a,b)=1 \Rightarrow \varphi(ab)=\varphi(a)\cdot\varphi(b)$.

Следствие из этого свойства: $\varphi(a)=a\prod_{t\ делит\ a}\left(1-\frac{1}{t}\right)$.

Пример

Дано $a=15$. Найдем простые все числа $t < a$, на которые делится a без остатка: *3, 5*. Тогда $\varphi(15)=15\cdot\left[\left(1-\frac{1}{3}\right)\cdot\left(1-\frac{1}{5}\right)\right]=8$. Посчитаем для проверки количество классов в приведенной системе вычетов: *1, 2, 4, 7, 8, 11, 13, 14*. Оно равно *8*. ●

Теорема 4.2 (Теорема Эйлера)

Пусть $m > 1$, $НОД(a, m) = 1$, $\varphi(m)$ – функция Эйлера. Тогда: $a^{\varphi(m)} \equiv 1(\mathrm{mod}\, m)$.

Например, $m=6, a=11$, $\varphi(6)=2 \Rightarrow 11^2=121 \equiv 1(\mathrm{mod}\, 6)$.

Теорема 4.3 (Малая теорема Ферма)

Если $НОД(a, p) = 1$, p – простое, то $a^{p-1} \equiv 1(\mathrm{mod}\, p)$

Следствие из теоремы Ферма: $a^p \equiv a(\mathrm{mod}\, p)$, где *не обязательно* $НОД(a,p)=1$, но p – простое.

Пример

Девятая степень однозначного числа оканчивается цифрой 7. Найти это число.

Решение

$a^9 \equiv 7(\mathrm{mod}10)$. Понятно, что $НОД(7,10)=1$, $НОД(a,10)=1$, $\varphi(10)=4$.

По теореме Эйлера: $a^4 \equiv 1(mod\ 10)$, возведем в квадрат обе части сравнения: $a^8 \equiv 1(\mathrm{mod}10)$. Почленно поделим $a^9 \equiv 7(\mathrm{mod}10)$ на $a^8 \equiv 1(\mathrm{mod}10)$: $a \equiv 7(\mathrm{mod}10) \Rightarrow a=7$. ●

Пример

Доказать, что $1^{18}+2^{18}+...+6^{18} \equiv 6(\mathrm{mod}7)$.

Решение

$НОД(1,7)=1, НОД(2,7)=1,..., НОД(6,7)=1$. $p = 7$ – простое. По теореме Ферма: $a^{7-1} \equiv 1(mod\ 7)$, где $a=1,2,...,6$. Получаем систем:

$1^6 \equiv 1(\mathrm{mod}\,7)$

$2^6 \equiv 1(\mathrm{mod}\,7)$

...

$6^6 \equiv 1(\mathrm{mod}\,7)$

Возводим каждое сравнение почленно в третью степень и складываем: $1^{18} + 2^{18} + ... + 6^{18} \equiv 6(\mathrm{mod}\,7)$.●

Пример

Найти остаток от деления 7^{402} на 101.

Решение

$p = 101$ – простое, $НОД(7,101) = 1 \Rightarrow$ по теореме Ферма $7^{100} \equiv 1(\mathrm{mod}101)$. Возведем данное сравнение в 4-ую степень и умножим на $7^2 \equiv 49(\mathrm{mod}101)$: $7^{402} \equiv 49(\mathrm{mod}101) \Rightarrow$ остаток равен 49.●

Пример

Найти две последние цифры числа 243^{402}.

Решение

Две последние цифры числа – это остаток от деления на *100*.

$243(\mathrm{mod}100) = (200+43)(\mathrm{mod}100) = 43(\mathrm{mod}100)$, значит:

$243^{402}(\mathrm{mod}100) = 43^{402}(\mathrm{mod}100)$. Так как $НОД(43,100) = 1$, по теореме Эйлера $43^{\varphi(100)} \equiv 1(\mathrm{mod}100)$. $\varphi(100) = \varphi(2^2 \cdot 5^2) = \varphi(2^2) \cdot \varphi(5^2)$, *2* и *5* простые числа, значит: $\varphi(100) = (2^2 - 2^1) \cdot (5^2 - 5^1) = 40$. Тогда: $43^{40} \equiv 1(\mathrm{mod}100)$; возведем в *10* степень и умножим на $43^2 \equiv 49(\mathrm{mod}100)$: $43^{402} \equiv 49(\mathrm{mod}100)$.

Значит, последние две цифры *4* и *9*. ●

Пример

Показать, что $73^{12} - 1$ делится на *105*.

Решение

$105 = 3 \cdot 5 \cdot 7$, $НОД(73,3) = НОД(73,5) = НОД(73,7) = 1$. По теореме Ферма:

$73^2 \equiv 1(\mathrm{mod}3) \overset{10св-во}{=} 1(\mathrm{mod}105)$

$73^4 \equiv 1(\mathrm{mod}5) \overset{10св-во}{=} 1(\mathrm{mod}105)$.

$73^6 \equiv 1(\mathrm{mod}7) \overset{10св-во}{=} 1(\mathrm{mod}105)$

Умножим эти сравнения и вычтем $1 \equiv 1(\mathrm{mod}105)$: $73^{12} - 1 \equiv 1 - 1(\mathrm{mod}105) = 0(\mathrm{mod}105)$.●

Пример

Найти вычет числа $2^{5432675}$ по *mod* 13.

Решение

Разделим степень на $p - 1 = 13 - 1$ с остатком $k = q(p-1) + r = 452722 \cdot 12 + 11$; $a^k(\mathrm{mod}\,p) = a^{q(p-1)+r}(\mathrm{mod}\,p) = (a^{p-1})^q a^r(\mathrm{mod}\,p) = (a^{p-1})^q(\mathrm{mod}\,p) \cdot a^r(\mathrm{mod}\,p)$. По теореме Ферма (так как p – простое): $a^{p-1} \equiv 1(\mathrm{mod}\,p)$.

Тогда $a^k(\mathrm{mod}\,p) = a^{q(p-1)+r}(\mathrm{mod}\,p) = \left(a^{p-1}\right)^q a^r(\mathrm{mod}\,p)$. Значит, $2^{5432675}(\mathrm{mod}13) \equiv 2^{11}(\mathrm{mod}13) = 7(\mathrm{mod}13)$.●

4.2.3.3. Сравнение первой степени

Рассмотрим сравнение:

$$ax \equiv b(\mathrm{mod}\,m). \qquad \textbf{(4.2)}$$

Под решением любого сравнения понимается класс вычетов по модулю m, один элемент которого (а значит и все) удовлетворяют данному сравнению. Решить сравнение – значит найти все x, которые удовлетворяют данному сравнению, при этом весь класс вычетов по модулю m считается за одно решение.

Пример

Сравнение 5-ой степени: $x^5 + x + 1 \equiv 0(\mathrm{mod}7)$.

Класс вычетов для $m = 7 : \overline{0,6}$. Данному сравнению удовлетворяют: $x_1 = 2$, $x_2 = 4$, так как $35 \equiv 0(\mathrm{mod}7)$ и $1029 \equiv 0(\mathrm{mod}7)$. Таким образом, у данного сравнения есть два решения: $35 \equiv 0(\mathrm{mod}7)$ и $x_2 \equiv 4(mod\ 7)$.●

Естественно, метод перебора затруднителен.

В случае сравнения первой степени если $НОД(\mathbf{a,m}) = 1$ решение означает отыскание целых u, v таких, что: $au + mv = 1$, то есть $au \equiv 1(mod\ m)$. u – это обратный к a по модулю m элемент.

Умножим (2) на u: $aux \overset{12св-во}{\equiv} bu(\mathrm{mod}\,m)$. С учетом $au \equiv 1(\mathrm{mod}\,m)$ получаем:

$$x \equiv bu(\mathrm{mod}\,m) \qquad \textbf{(4.3)}$$

Значит, сравнение первой степени имеет одно решение по модулю m.

Если же $НОД(\mathbf{a,m}) = d$, то для решения сравнения (2) необходимо, чтобы d делил b без остатка. При этом сравнение будет иметь d решений: $x_1, x_1 + m, ..., x_1 + (d-1)m$.

Пример

Решить сравнение $111x = 75(mod\ 321)$.

Решение

$НОД(a, m) = НОД(111,321) = 3$. Поэтому по свойствам 11,7,8 функции сравнение: $37x = 25(mod\ 107)$. Далее для решения сравнения можно использовать два подхода:

1. Алгоритм Евклида (применяется при небольших m для нахождения u: $au \equiv 1(mod\ m)$).
2. Способ Эйлера. ●

4.2.3.3.1. Решение сравнения первой степени с использованием алгоритма Евклида

Рассмотрим алгоритм Евклида для нахождения наибольшего общего делителя $НОД(a, b)$[5].

Выполним следующие деление с остатком:

$a = bq_1 + r_1, 0 \le r_1 < b$

$b = r_1 q_2 + r_2, 0 \le r_2 < r_1$

$r_1 = r_2 q_3 + r_3, 0 \le r_3 < r_2$

....

$r_{n-2} = r_{n-1} q_n + r_n, 0 \le r_n < r_{n-1}$

$n_{n-1} = r_n q_{n+1}.$

Тогда $НОД(a, b) = НОД(b, r_1) = НОД(r_1, r_2) = ... = r_n$. То есть $НОД(a, b)$ равен последнему отличному от нуля остатку в алгоритме Евклида.

Пример

Найдем $НОД(1234,54)$.

Решение

$1234 = 54 \cdot 22 + 46$

$54 = 46 \cdot 1 + 8$

$46 = 8 \cdot 5 + 6$

$8 = 6 \cdot 1 + 2$

$6 = 2 \cdot 3 + 0$

Значит, $НОД(1234,54) = 2$.

Если мы хотим найти линейное разложение НОД'а: $r_n = au + bv$, то надо остатки r_i выражать друг через друга:

$r_n = 2 = 8 - 6 = 8 - (46 - 8 \cdot 5) = (54 - 46) - (46 - (54 - 46) \cdot 5) =$

$= (54 - [1234 - 54 \cdot 22]) - ([1234 - 54 \cdot 22] - (54 - [1234 - 54 \cdot 22]) \cdot 5) =$

$= -7 \cdot 1234 + 160 \cdot 54 \bullet$

Пример

Решим сравнение $111x \equiv 75 (\mathrm{mod} 322)$.

Решение

$НОД(111,322) = 1$.

Найдем линейное разложение НОД'а: $au + mv = 1$:

$322 = 111 \cdot 2 + 100$

$111 = 100 \cdot 1 + 11$

$100 = 11 \cdot 9 + 1$

$11 = 1 \cdot 11$

[5] Данный алгоритм на языке С приведен в приложении к лабораторным работам

$r_n = 1 = 100 - 11\cdot 9 = 100 - (111 - 100)\cdot 9 =$
$= \left[\overline{322} - \underline{111}\cdot 2\right] - \left(\underline{111} - \left[\overline{322} - \underline{111}\cdot 2\right]\right)\cdot 9 =$
$= 10\cdot 322 - 29\cdot 111$
Значит, $v = 10, u = -29$. По формуле (4.3): $x \equiv 75\cdot(-29)(\mathrm{mod}322) = -2175(\mathrm{mod}322) = 79(\mathrm{mod}322)$.
Проверим, действительно ли так: $111\cdot 79 \mathrm{mod} 322 = 75$. ●

Пример
Решим сравнение $111x \equiv 75(\mathrm{mod}321)$.

Решение
НОД$(111,321) = 3$. Получаем: $37x = 25(\mathrm{mod}107)$. Найдем линейное разложение НОД'а: $au + mv = 1$:
$107 = 37\cdot 2 + 33$
$37 = 33\cdot 1 + 4$
$33 = 4\cdot 8 + 1$
$4 = 4\cdot 1$
$r_n = 1 = 33 - 4\cdot 8 = 33 - [37 - 33]\cdot 8 = \left(\overline{107} - \underline{37}\cdot 2\right) - \left[\underline{37} - \left(\overline{107} - \underline{37}\cdot 2\right)\right]\cdot 8 =$
$= 9\cdot 107 - 26\cdot 37$
Значит, $u = -26, v = 9$.
Получается, что $x_1 \equiv -26\cdot 25(\mathrm{mod}107) = 99(\mathrm{mod}107)$.
Исходное решение: $x_1 \equiv 99(\mathrm{mod}321), x_2 \equiv (99+107)(\mathrm{mod}321) = 206(\mathrm{mod}321), x_3 \equiv (99+2\cdot 107)(\mathrm{mod}321) = 313(\mathrm{mod}321)$ ●

4.2.3.3.2. Решение сравнения первой степени с использованием расширенного алгоритма Евклида

Рассмотрим расширенный алгоритм Евклида[6].

Как мы убедились в обычном алгоритме Евклида чтобы найти u, v надо выражать через друг друга остатки r_i, что не удобно.

Данный алгоритм позволяет найти u, v в $d = au + mv$, *НОД*(a, m) = d.

Необходимо работать с системой равенств:

$$\begin{cases} y_1 a + y_2 m = u_3 \\ z_1 a + z_2 m = v_3 \\ t_1 a + t_2 m = t_3 \end{cases}$$

$$(y_1, y_2, y_3) = (1,0,m)$$

На первом шаге полагаем: $(z_1, z_2, z_3) = (0,1,a)$

$$(t_1, t_2, t_3) = (0,0,0)$$

На очередном шаге проверяем: если $z_3 = 0$, то искомые значения d, u, v равны y_3, y_2, y_1. Иначе производим деление с остатком: $y_3 = qz_3 + r$,

[6] Данный алгоритм на языке С приведен в приложении к лабораторным работам

$(t_1, t_2, t_3) = (v_1, v_2, v_3)$
$(v_1, v_2, v_3) = (u_1 - qv_1, u_2 - qv_2, u_3 - qv_3)$.
$(u_1, u_2, u_3) = (t_1, t_2, t_3)$

Пример

Решим сравнение расширенным алгоритмом Евклида $111x \equiv 75 (\mathrm{mod} 322)$.

Решение

$y = (1 \quad 0 \quad 322)$
$z = (0 \quad 1 \quad 111)$

Шаг 1.

Так как $z_3 \neq 0$, то производим деление: $322 = 2 \cdot 111 + 100$. Значит, $q = 2$. Далее вычисляем:

$t = (0 \quad 1 \quad 111)$
$z = (1 - 2 \cdot 0 \quad 0 - 2 \cdot 1 \quad 322 - 2 \cdot 111) = (1 \quad -2 \quad 100)$
$y = (0 \quad 1 \quad 111)$

Шаг 2.

Так как $z_3 \neq 0$, то производим деление: $111 = 1 \cdot 100 + 11$. Значит, $q = 1$. Далее вычисляем:

$t = (1 \quad -2 \quad 100)$
$z = (0 - 1 \cdot 1 \quad 1 - 1 \cdot (-2) \quad 111 - 1 \cdot 100) = (-1 \quad 3 \quad 11)$
$y = (1 \quad -2 \quad 100)$

Шаг 3.

Так как $z_3 \neq 0$, то производим деление: $100 = 9 \cdot 11 + 1$. Значит, $q = 9$. Далее вычисляем:

$t = (-1 \quad 3 \quad 11)$
$z = (1 - 9 \cdot (-1) \quad -2 - 9 \cdot 3 \quad 100 - 9 \cdot 11) = (10 \quad -29 \quad 1)$
$y = (-1 \quad 3 \quad 11)$

Шаг 4.

Так как $z_3 \neq 0$, то производим деление: $11 = 11 \cdot 1 + 0$. Значит, $q = 11$. Далее вычисляем:

$t = (10 \quad -29 \quad 1)$
$z = (-11 - 1 \cdot 10 \quad 3 - 11 \cdot (-29) \quad 11 - 11 \cdot 1) = (-111 \quad 320 \quad 0)$
$y = (10 \quad -29 \quad 1)$

Шаг 5.

Так как $z_3 = 0$, то найдено разложение: $НОД(a, m) = d = y_3 = 1$, $u = y_2 = -29$, $v = y_1 = 10$.
$d = au + mv$: $1 = 111 \cdot (-29) + 322 \cdot 10$. Верно. По формуле (4.3) находим решение исходного сравнения:
$x \equiv 75 \cdot (-29) (\mathrm{mod} 322) = -2175 (\mathrm{mod} 322) = 79 (\mathrm{mod} 322)$.●

4.2.3.3.3. Решение сравнения способ Эйлера

Теорема 4.4

Пусть $m > 1$, $НОД(a, m) = 1$. Тогда сравнение (4.2) имеет решение (4.3), где $u = a^{\varphi(m)-1}$.

Пример

Решить сравнение $7x \equiv 3 (\mathrm{mod} 10)$.

Решение

$НОД(7,10) = 1$, $\varphi(10) = 4$.

Значит: $x = 3 \cdot 7^3 (\mathrm{mod} 10) = 1029 (\mathrm{mod} 10) = 9 (\mathrm{mod} 10)$. ●

Отрицательным моментом этого метода является то, что необходимо возводить a в степень.

4.2.3.4. Первообразные корни

Число a, взаимно простое с модулем m ($НОД(a,m) = 1$), принадлежит показателю n, если n такое наименьшее натуральное число, что выполняется сравнение: $a^n \equiv 1 (mod\ m)$.

Свойства данного числа a:

1. Числа $a^0, a^1, ..., a^{n-1}$ попарно не сравнимы по модулю n.
2. $a^t \equiv a^{t'} (\mathrm{mod} m) \Leftrightarrow t \equiv t' (\mathrm{mod} n)$.
3. n делит $\varphi(m)$ без остатка.

Число, принадлежащее показателю $\varphi(m)$, называется первообразным корнем.

4. По любому простому модулю p существует первообразный корень.
5. Пусть $c = \varphi(m)$, $q_1, q_2, ..., q_k$ – различные простые делители числа c. Число a, *взаимно простое с модулем* m, будет первообразным корнем тогда и только тогда, когда не выполнено ни одно из следующих сравнений: $a^{c/q_1} \equiv 1 (\mathrm{mod} m)$, $a^{c/q_2} \equiv 1 (\mathrm{mod} m), \ldots, a^{c/q_k} \equiv 1 (\mathrm{mod} m)$.

Теорема 4.5

Если p – нечетное простое число, то по модулям вида p^k и $2 \cdot p^k$ ($k \geq 1$) существуют первообразные корни.

Пример

Найти первообразные корни для числа $m = 7$.

Решение

Функция Эйлера от простого числа: $c = \varphi(7) = 7 - 1 = 6$. Простые делители c: *2* и *3*. Значит, первообразный корень a не должен удовлетворять сравнениям: $a^3 \equiv 1 (\mathrm{mod} 7)$ и $a^2 \equiv 1 (\mathrm{mod} 7)$. Числа, взаимно простые с m: $\overline{1, m-1}$.

Проверим их:

$2^3 \pmod 7 = 1$ *не подходит, т.к.* $1 \equiv 1 \pmod 7$

$3^3 \pmod 7 = 6$

$3^2 \pmod 7 = 2$ *подходит*

$4^3 \pmod 7 = 1$ *не подходит, т.к.* $1 \equiv 1 \pmod 7$

$5^3 \pmod 7 = 6$

$5^2 \pmod 7 = 4$ *подходит*

Значит, первообразными корнями по модулю $m = 7$ являются *3* и *5*.

Проверим выполнение сравнения: $a^{\varphi(m)} \equiv 1 \pmod m$. $3^6 \equiv 1 \pmod 7$, $5^6 \equiv 1 \pmod 7$ – верно. ●

4.2.3.5. Дискретные логарифмы в конечном поле

Ранее мы рассматривали задачу: по заданным n, a, p найти $b = a^n \bmod p$.

$$a^n \equiv b \pmod p \qquad \textbf{(4.4)}$$

Обратная задача: для сравнения (4.4) по b, a, p найти его дискретный логарифм (или индекс) n. Наиболее быстрый алгоритм решения данной задачи имеет сложность, близкую к факторизации n методом решета числового поля.

Безопасность многих алгоритмов с открытыми ключами основана на сложности вычисления дискретного логарифма.

С помощью перебора можно решить (4.4) за $O(p)$ арифметических операций.

Решение $\log_a b$ уравнения (4) можно находить по формуле $\log_a b \equiv \sum_{j=1}^{p-1} \left(1 - a^j\right)^{-1} b^j \pmod{(p-1)}$. Однако сложность вычисления по этой формуле хуже, чем для перебора.

Следующий алгоритм решения (4) имеет сложность $O\left(p^{1/2} \log p\right)$ арифметических операций.

Алгоритм согласования.

1. Присвоить $H = \left[p^{1/2}\right] + 1$.
2. Найти $c \equiv a^H \pmod p$.
3. Составить таблицу значений $c^u \pmod p, 1 \le u \le H$, и упорядочить ее.
4. Составить таблицу значений $b \cdot a^v \pmod p, 0 \le v \le H$, и упорядочить ее.
5. Найти совпавшие элементы из первой и второй таблиц. Для них $c^u \equiv b \cdot a^v \pmod p$ откуда $a^{Hu-v} \equiv b \pmod p$.
6. Выдать $x \equiv Hu - v \pmod{(p-1)}$.

4.2.4. Примеры систем шифрования, основанные на проблемах теории чисел

4.2.4.1. Система шифрования RSA

Данная криптосистема является шифрованием с открытым ключом. Алгоритм RSA[7] опирается на тот факт, что найти большие простые числа вычислительно легко, но разложить большое число на произведение двух таких чисел практически невозможно.

Эта система построена на основе степенной функции: $f(x) = x^e \bmod n$.

Блок x открытого текста, представленный как целое число из сегмента $[0, n-1]$, преобразуется в блок шифротекста из того же сегмента с помощью вычисления: $y = x^e \bmod n$. Пара (e,n) называется открытым ключом (используется при шифровании). Число $n = p \cdot q$, где p, q – различные случайные простые числа (они генерируются только одной из сторон-участниц).

При расшифровывании используется функция $\tilde{f}(x) = x^d \bmod n$: $x = y^d \bmod n$. Пара (d,n) называется закрытым ключом (используется при дешифровании).

В основе согласованности этих преобразований лежит теорема Эйлера.

(пусть $n > 1$, $НОД(z,n) = 1$; тогда: $z^{\varphi(n)} \equiv 1 (\bmod n)$).

Для того чтобы воспользоваться этой теоремой, в RSA используют числа e, d такие, что: $ed \equiv 1 (\bmod \varphi(n))$, $\varphi(n) = (p-1)(q-1)$. Если e известно, то d – это решение сравнения первой степени; или, проверив $НОД(e, \varphi(n)) = 1$, можно по расширенному алгоритму Евклида найти d, t: $ed + t\varphi(n) = 1$.

Проверим, действительно ли формула дешифрования работает:

$$x = y^d \bmod n = \left(x^e \bmod n\right)^d \bmod n = \left[\left(x^e \bmod n\right) \cdot \ldots \cdot \left(x^e \bmod n\right)\right] \bmod n =$$

$$= x^{ed} \bmod n = x^{1-t\varphi(n)} \bmod n = x \cdot x^{-t\varphi(n)} \bmod n =$$

$$= \left[x \bmod n \cdot \left(x^{\varphi(n)} \bmod n\right)^{-t}\right] \bmod n \overset{T.Эйлера}{=} \left[x \bmod n \cdot 1^{-t}\right] \bmod n = x$$

Криптостойкость системы шифрования RSA основана на том, что, зная открытый ключ (e,n), злоумышленник может найти d:

a) зная функции Эйлера $\varphi(n)$. Данная задача – есть задача разложения (факторизации) n на простые множители, так как найти функцию Эйлера по определению вычислительно долго. Используя самый эффективный алгоритм факторизации (решето числового поля), число n, имеющее длину в *50* десятичных разрядов, можно разложит приблизительно за 10^{10} операций (суперкомпьютеры), а *800* десятичных разрядов – приблизительно за 10^{51} операций (не раскладывается в настоящее время).

b) Либо найти $\sqrt[e]{y}$. (это задача дискретного логарифма – см. пункт *4.2.3.5*).

Пример

Для простоты вычислений будем использовать маленькие числа.

Для перевода сообщений в числа будем использовать таблицу ASCII кодов минус *64*.

Пусть абонент A сгенерировал два простые числа: $p = 3, q = 11 \Rightarrow n = 33, \varphi(n) = 20$. Находим e: $НОД(e,n) = 1$, $1 < e < \varphi(n)$, например $e = 7$.

Пара $(7,33)$ публикуется как открытая.

Находим d: $7d \equiv 1 (\bmod \varphi(n)) \Rightarrow d = 3$.

Пара $(3,33)$ держится абонентом A в тайне.

[7] RSA был предложен в 1977г. Роном Ривестом, Ади Шамиром, Леном Адлеманом.

Абонент B желает послать секретное сообщение: «CAB». Тогда $A \to 1, B \to 2, C \to 3$.

Шифруем: $C_1 = 3^7 \bmod 33 = 9$, $C_2 = 1^7 \ mod \ 33 = 1$, $C_3 = 2^7 \bmod 33 = 29$.

Попробуем дешифровать: $C_1' = 9^7 \bmod 33 = 15$ (не получается!).

Абоненту A посылают: «IA]».

Абонент A переводи сообщение в числовое представление и дешифрует:
$M_1 = 9^3 \bmod 33 = 3$, $M_2 = 1^3 \bmod 33 = 1$, $M_3 = 29^3 \bmod 33 = 2$. Значит, сообщение: «CAB».●

4.2.4.2. Система шифрования Диффи-Хеллмана

Данный алгоритм[8] также является алгоритмом секретного обмена между абонентами ключом. Он основан на трудности вычисления дискретного логарифма.

Рассмотрим простейшую схему создания общего ключа двумя абонентами.

Абоненты A, B выбирают большое простое число n и g – первообразный корень по модулю n. Получается пара (n, g), которая объявляется открытой.

Абонент A выбирает случайное большое секретное число $x < n$ и посылает абоненту B: $X = g^x \ mod \ n$.

Абонент B выбирает случайное большое секретное число $y < n$ и посылает абоненту A: $Y = g^y \ mod \ n$.

Абонент A вычисляет ключ: $k_A = Y^x \bmod n$.

Абонент B вычисляет ключ: $k_B = X^y \bmod n$.

Проверим, одинаковый ли у абонентов ключ:

$$k_A = Y^x \bmod n = \left(g^y \bmod n\right)^x \bmod n = g^{xy} \bmod n = \left(g^x \bmod n\right)^y \bmod n =$$

$$= X^y \bmod n = k_B$$

Далее полученным ключом можно работать в симметричной криптосистемой.

Для того чтобы сократить размер возможных окончательных значений ключа, важно чтобы функция модульного возведения в степень была настолько почти взаимнооднозначной, насколько это возможно. В том случае, когда n является простым числом, всегда существует такое g, что $g^x \bmod n$ принимает каждое из значений в промежутке $\overline{0, n-1}$, когда x пробегает значения из того же интервала. Поэтому g выбирают как первообразный корень по модулю n.

Пример

$n = 7, \varphi(n) = 6 = 2 \cdot 3$.

В примере пункта *4.2.3.4* мы выяснили, что первообразные корни по модулю 6 : *3* и *5*. Возьмем, например $g = 3$. $(7,3)$ – открыта.

Абонент A сгенерировал случайное число $x = 2 < n$, и вычислив $X = 3^2 \bmod 7 = 2$, послал абоненту B.

Абонент B сгенерировал случайное число $y = 5 < n$, и вычислив $Y = 3^5 \bmod 7 = 5$, послал абоненту A.

Абонент A принял $Y = 5$ и вычислил ключ: $k_A = 5^2 \bmod 7 = 4$.

Абонент B принял $X = 2$ и вычислил ключ: $k_A = 2^2 \bmod 7 = 4$.●

[8] Предложен в 1976г. У. Диффи и М.Э.Хеллманом.

4.2.5. Разложение на множители (факторизация)

Разложение на множители числа (например, $252601 = 41 \cdot 61 \cdot 101$) является одной из древнейших проблем теории чисел. Приступая к факторизации числа, следует сначала убедиться, что оно не простое. Это можно сделать с помощью одного из описанных выше тестов простоты. Их трудоемкость, как правило, значительно меньше, чем у алгоритмов факторизации.

Самые лучшие алгоритмы на сегодняшний день:

1. Решето общего числового поля (самый быстрый из известных алгоритмов для чисел размером 110 и более десятичных разрядов).
2. Квадратичное решето для чисел размером (для чисел менее 110 десятичных разрядов; в 1994г. за 8 месяцев на 1600 процессорах было разложено 129 разрядное число).
3. Метод эллиптической кривой.
4. Монте-Карло Поларда.
5. Метод непрерывных дробей (по времени не подходит для реализации).
6. Проверка делением (самый старый алгоритм разложения; состоит в проверке на деление на каждое простое число, меньше или равное квадратному корню из раскладываемого числа).

Почти все эти алгоритмы субэкспоненциальной сложности. Алгоритмов факторизации, имеющих оценки сложности лучше указанной, в настоящее время не существует.

В лабораторной работе №7 мы рассмотрим более простые методы, имеющие экспоненциальную сложность.

4.2.6. Вычисление в поле Галуа

Напомним некоторые определения из алгебраических основ:

Поле F_p – это множество p элементов, на котором определены операции сложения и умножения, обладающие свойствами коммуникативности, ассоциативности и дистрибутивности, при этом относительно этих двух операций существуют нейтральные элементы и $\forall a$ существует обратный элемент относительно операции сложения, $\forall b \neq 0$ существует обратный элемент относительно операции умножения.

Пусть $f(\lambda)$ – неприводимый многочлен над полем F, для него существует конечное расширение поля F, содержащее все корни многочлена $f(\lambda)$ – поле разложения.

Группа – это непустое множество G с алгебраической операцией * на нем, для которой выполняются аксиомы:

1. Операция * ассоциативна $\forall a,b,c \in G: a*(b*c)=(a*b)*c$.
2. В G $\exists e$ единичный элемент такой, что $\forall a \in G: a*e=e*a=a$.
3. $\forall a \in G\ \exists a^{-1} \in G: a * a^{-1} = a^{-1} * a = e$.

Если дополнительно группа удовлетворяет аксиоме:

4. $\forall a,b \in G: a * b = b * a$.

То это абелева (коммуникативная) группа.

Кольцом называется множество R с двумя бинарными операциями $(+,\cdot)$ такими, что:

1. R – абелева группа относительно операции $+$.
2. Операция умножения ассоциативна: $\forall a, b, c \in R: (ab)c = a(bc)$;
3. Выполняется закон дистрибутивности $\forall a,b,c \in R: a(b+c) = cb + ac$.

Кольцо классов вычетов Z_p называется полем Галуа порядка p и обозначается $GF(p)$ (где p – простое). В поле Галуа определены операции $+,-,*,/$.

Теорема 1.

Поле Галуа $GF\left(p^n\right)$ есть поле разложения всякого неприводимого многочлена $f(\lambda)$ степени n над полем $F_p = GF\left(p\right)$.

Арифметика поля Галуа широко используется в криптографии. Данное поле содержит только числа конечного размера, при делении отсутствуют ошибки округления. Многие криптосистемы основаны на $GF(p)$, где p – большое простое число.

Криптографы также используют арифметику по модулю неприводимых многочленов степени n, коэффициентами которых являются целые числа по модулю q, где q – простое число. Эти поля называются $GF(q^n)$.

Пример (байтовые операции)

Рассмотрим поле Галуа $GF(2^8)$. Оно может интерпретироваться как работа с битами одного байта, который будет рассматриваться как элемент этого конечного поля с бинарными операциями: $\oplus, \cdot$.

Сложение: $\oplus$ – это поразрядное суммирование по модулю 2.

Умножение – это умножение полиномов, соответствующих байтам, но по модулю неприводимого двоичного полинома $m(x) = x^8 + x^4 + x^3 + x + 1$ $(11B_{16})$.

Элемент этого поля может быть представлен в полиномиальном виде:

$b(x) = b_7 x^7 + ... + b_1 x + b_0$, где b_i – i-ый бит байта b.

Например: $b = 87_{10} = 1010111_2$, тогда $b(x) = x^6 + x^4 + x^2 + x + 1$.

Сложение: $\left(87 + 131\right)_{10} = \left(x^6 + x^4 + x^2 + x + 1\right) \oplus \left(x^7 + x + 1\right) = x^7 + x^6 + x^4 + x^2$.

Умножение: умножение на x эквивалентно побитовому сдвигу влево на один бит. Операция умножения $b(x) \cdot x$ обозначается $xtime(b)$. Умножение на x^i эквивалентно побитовому сдвигу влево на i бит и равносильна k-кратной композиции $xtime(b)$.

$\left(87 \cdot 16\right)_{10} = \left(x^6 + x^4 + x^2 + x + 1\right) \cdot \left(x^4 + x + 1\right)$.

Умножаем $\left(x^6 + x^4 + x^2 + x + 1\right)$ на $1 = x^0$, то есть сдвигаем на 0 бит, получаем: $\left(x^6 + x^4 + x^2 + x + 1\right)$.

Умножаем $\left(x^6 + x^4 + x^2 + x + 1\right)$ на $x = x^1$, то есть сдвигаем на 1 бит, получаем: $\left(x^7 + x^5 + x^3 + x^2 + x\right)$.

Чтобы умножить $\left(x^6 + x^4 + x^2 + x + 1\right)$ на x^4 надо произвести умножение на $x^0 \div x^4$, и результат сложить. Поэтому:

Умножаем $\left(x^7 + x^5 + x^3 + x^2 + x\right)$ на x: $\left(x^8 + x^6 + x^4 + x^3 + x^2\right)$, приводим по $m(x) = x^8 + x^4 + x^3 + x + 1$ и получаем: $\left(x^6 + x^2 + x + 1\right)$.

Умножаем $\left(x^6 + x^2 + x + 1\right)$ на x: $\left(x^7 + x^3 + x^2 + x\right)$.

Умножаем $\left(x^7 + x^3 + x^2 + x\right)$ на x: $\left(x^8 + x^4 + x^3 + x^2\right)$, приводим по $m(x) = x^8 + x^4 + x^3 + x + 1$ и получаем: $\left(x^2 + x + 1\right)$.

В итоге получаем

$\left(87 \cdot 16\right)_{10} = \left(x^6 + x^4 + x^2 + x + 1\right) \cdot \left(x^4 + x + 1\right) =$

$:= \left(x^2 + x + 1\right) \oplus \left(x^7 + x^5 + x^3 + x^2 + x\right) \oplus \left(x^6 + x^4 + x^2 + x + 1\right) =$

$= x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x \bullet$

Пример (операции с 4-байтовыми векторами)

Любой многочлен принадлежащий $GF(2^8)$ степени не больше 3 может быть представлен 4-х байтовым вектором.

Сложение 4-х байтовых векторов производится $\oplus$ побитово.

Умножение: с последующим приведением по модулю полинома $M(x) = x^4 + 1$.

Пусть дано два многочлена: $a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$, $b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$. Их произведение равно $c(x) = c_6x^6 + ... + c_1x + c_0$, где:

$c_0 = a_0 * b_0$

$c_1 = a_1 * b_0 \oplus a_0 * b_1$

$c_2 = a_2 * b_0 \oplus a_1 * b_1 \oplus a_0 * b_2$

$c_3 = a_3 * b_0 \oplus a_2 * b_1 \oplus a_1 * b_2 \oplus a_0 * b_3$

$c_4 = a_3 * b_1 \oplus a_2 * b_2 \oplus a_1 * b_3$

$c_5 = a_3 * b_2 \oplus a_2 * b_3$

$c_0 = a_3 * b_3$

Далее делим $c(x)$ на $M(x)$ и окончательно получаем, что если $d(x) = a(x) \cdot b(x)$, то:

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \cdot \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}.$$

Примерами алгоритмов шифрования, основанных на вычисление в поле Галуа является RIJNDAEL (AES), A5/1.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Шеннон К.* Теория связи в секретных системах. / В кн. Лекции по теории информации и кибернетике.–М.: ИЛ, 1963.
2. *Харин Ю.С., Берник В.И., Матвеев Г.В., Агиевич* С.В. Математические и компьютерные основы криптологии: Учеб. пособие. / Ю.С. Харин, В.И. Берник, Г.В. Матвеев, С.В. Агиевич. – Минск: ООО Новое знание, 2003.– 382 с.
3. *Шнайер Б.*, Прикладная криптография. Протоколы, алгоритмы и исходные тексты на языке С, 2-ое издание. / Брюс Шнайер. Пер. с англ. – М.: "Триумф", 2002. – 816 с.
4. *Фомичев В. М.* Дискретная математика и криптология. / В. М. Фомичев.– М.: Диалог-МИФИ, 2003.– 400 с.
5. *Черемушкин А.В.* Лекции по арифметическим алгоритмам в криптографии./ А.В. Черемушкин.– М.: МЦНМО, 2002. – 104 с.
6. *Терехов А.А.* Криптографическая защита информации: Конспект лекций./ А.А.Терехов. – СПб: Кафедра системного программирования СПбГУ, 1999.– 52 с.
7. Введение в криптографию. /Под общей ред. В.В. Ященко. – СПб: Питер, 2000.–288 с.
8. *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии: Монография. / О. Н. Василенко. – М.: Московский центр непрерывного математического образования, 2003.–328 с.
9. *Потапов В.Н.* Теория информации. Кодирование дискретных вероятностных источников: Учеб. пособие. / В.Н. Потапов.– Новосибирск: НГУ, 1999. – 71с.
10. *Шульгин В.И.* Основы теории передачи информации. Ч. 1. Экономное кодирование): Учеб. пособие. /В.И. Шульгин. – Харьков: Нац. аэрокосм. ун-т “ХАИ”, 2003. – 102 с.
11. *А.А.Ожиганов, М.В.Тарасюк* Передача данных по дискретным каналам (учебно-методическое пособие по дисциплине "Теория информации"), Санкт-Петербург, 1999
12. *Богуш Р.П.* Элементы теории информации: Учеб.-метод. комплекс для студ. спец. 1-40 01 01 «Программное обеспечение информационных технологий». / Сост. и общ. ред. Р.П. Богуша. – Новополоцк: УО «ПГУ», 2006. – 160 с.
13. *Ломакин Д. В.* Прикладная теория информации: Конспект лекций. / Д. В. Ломакин. – Н. Новгород, 1998 – 62 с.
14. *Лидовский В.В.* Теория информации: Учеб. пособие. / В.В. Лидовский. – М.: Компания Спутник+, 2004. – 111 с.
15. *Погорелов Б. А., Черемушкин А. В., Чечета С. И.* Об определении основных криптографических понятий. / Б. А. Погорелов, А.В. Черемушкин, С. И. Чечета // Доклад на конференции "Математика и безопасность информационных технологий", МаБИТ-03, МГУ, 23-24 октября 2003 г.
16. *Цой Е. Б., Самочернов И. В.* Моделирование и управление в экономике. Ч. 1: Курс лекций. / Е. Б. Цой, И. В. Самочернов. – Новосиб. гос. техн. ун-т. – Новосибирск: Изд-во НГТУ, 2003. – 104 с.
17. *Сизый С. В.* Лекции по теории чисел: Учеб. пособие для математических специальностей. / С. В.Сизый. – Екатеринбург: Уральский государственный университет им. А. М. Горького, 1999.
18. *Коутинхо С.* Введение в теорию чисел. Алгоритм RSA. / С. Коутинхо. Пер. с англ. – М.: Постмаркет, 2001. – 323 с.
19. *Асосков А.В., Иванов М.А., Мирский А.А.* Поточные шифры. / А. В. Асосков, М. А. Иванов, А. А. Мирский и др. Поточные шифры. – М.: КУДИЦ-ОБРАЗ, 2003. – 334 с.
20. *Самсонов Б. Б., Плохов Е. М., Филоненков А. И., Кречет Т. В.* Теория информации и кодирования. / Б. Б. Самсонов, Е. М. Плохов, А. И. Филоненков, Т. В. Кречет. – Ростов-на-Дону: Феникс, 2002. – 288 с.
21. *Мао В.*Современная криптография. Теория и практика. – Пер. с англ. –М.:Издательский дом ”Вильямс”, 2005. – 768с.

22. *Столлингс В.* Криптография и защита сетей. Принципы и практика – Изд. 2-е: Пер. с англ. – М.:Издательский дом ”Вильямс”, 2001. – 672с
23. *Гультяева Т.*А. Основы теории информации и криптографии . Ч. 1 : методические указания к выполнению лабораторных работ для 3 курса ФПМИ по специальностям "Математическое обеспечение и администрирование информационных систем" (010503), "Прикладная информатика в менеджменте" (080801) / Новосиб. гос. техн. ун-т – Изд-во НГТУ , 2006 – 27 с.
24. *Гультяева Т.А*. Основы теории информации и криптографии . Ч. 2 : методические указания к выполнению лабораторных работ для 3 курса ФПМИ по специальностям "Математическое обеспечение и администрирование информационных систем" (010503), "Прикладная информатика в менеджменте" (080801) / Новосиб. гос. техн. ун-т – Изд-во НГТУ , 2009 – 44 с.

ОГЛАВЛЕНИЕ