Системы цифрового телевидения для тех, кто хочет понять: кодирование, исправляющее ошибки

Константин Гласман

Часть 5. Начало в №№ 6…9/2020

Код Хэмминга: декодирование и реализация кодера и декодера

Если декодер способен сделать правдоподобную оценку шумового блока с использованием синдрома, то он сможет получить и оценку посланного кодового блока:

$$x' = y - e \qquad (32)$$

Результаты вычисления синдромного вектора для кода Хэмминга (кодирование было рассмотрено в № 9/2020, стр. 46), по формуле (31) для нулевого шумового блока и шумовых блоков с одиночными ошибками приведены в таб. 14.

Табл.14. Компоненты синдрома для шумовых блоков с одиночной ошибкой

Шумовой блок e							Синдром s		
e_1	e_2	e_3	e_4	e_5	e_6	e_7	s_1	s_2	s_3
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	1	1	1
0	0	1	0	0	0	0	1	1	0
0	0	0	1	0	0	0	0	1	1
0	0	0	0	1	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0
0	0	0	0	0	0	1	0	0	1

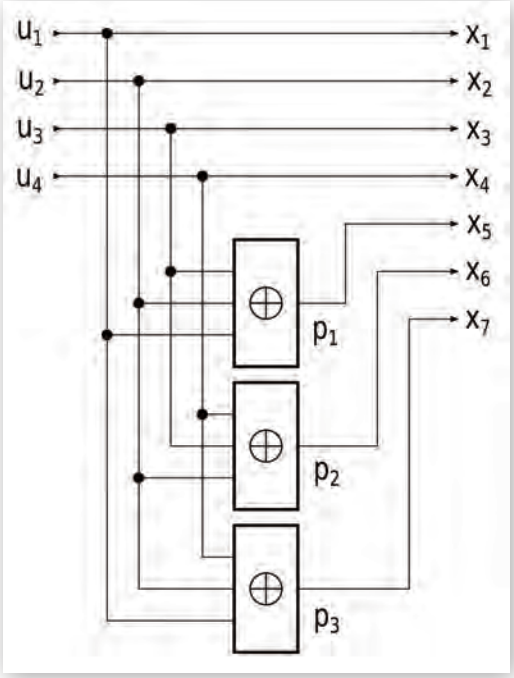

Рис. 7. Схема кодера на основе (7, 4)-кода Хэмминга

Символ в шумовом блоке в некотором разряде, равный 1, означает поражение посланного кодового блока в этом символе. Значение символа в этом разряде кодового слова меняется на противоположное при прохождении по каналу связи. Как следует из табл. 14, при отсутствии ошибок синдром равен нулю (все компоненты синдрома как вектора-строки равны 0). Каждой конфигурации ошибок (то есть каждой позиции ошибки) соответствует определенное значение синдрома. Стоит вспомнить, что минимальное расстояние рассматриваемого кода равно $d^*=3$, поэтому код способен исправлять только одиночные ошибки.

А вот как работает декодер. Пусть, например, на выходе канала связи было принято слово $\boldsymbol{y}=0000011$. Умножение $\boldsymbol{y}$ на $\boldsymbol{H}^T$ и расчет по формуле (29) с использованием данных табл. 13 (№ 9/2020, стр. 48) дает следующие значения компонент синдрома:

$$\begin{aligned} s_1 &= 0*1+0*1+0*1+0*0+0*1+1*0+1*0 = 0 \\ s_2 &= 0*0+0*1+0*1+0*1+0*0+1*1+1*0 = 1 \\ s_3 &= 0*1+0*1+0*0+0*1+0*0+1*0+1*1 = 1 \end{aligned} \qquad (34)$$

Итак, синдром равен $\boldsymbol{s}=011$. Из табл. 14 видно, что синдрому $\boldsymbol{s}=011$ соответствует ошибка $\boldsymbol{e}=0001000$. Вычитание из принятого слова оценки ошибки дает оценку посланного слова $\boldsymbol{x'=y-e}=0000011-0001000=0001011$. Поскольку код систематический, то первые четыре символа указывают на информационное слово $\boldsymbol{u}=0001$. Такое решение соответствует декодированию по методу наибольшего правдоподобия.

Формулу (29) и соотношения (34) для рассматриваемого кода можно переписать в форме, которая явно указывает ненулевые элементы строк проверочной матрицы (табл. 12, № 9/2020, стр. 48) и столбцов транспонированной проверочной матрицы (табл.13) для рассматриваемого (7, 4)-кода Хэмминга:

$$\begin{aligned} s_1 &= y_1 + y_2 + y_3 + y_5 \\ s_2 &= y_2 + y_3 + y_4 + y_6 \\ s_3 &= y_1 + y_2 + y_4 + y_7 \end{aligned} \qquad (35)$$

Такая форма может оказаться более удобной для практической реализации декодера, подобно тому, как соотношения (7) (№ 9/2020, стр. 46) определяют простой способ реализации кодера.

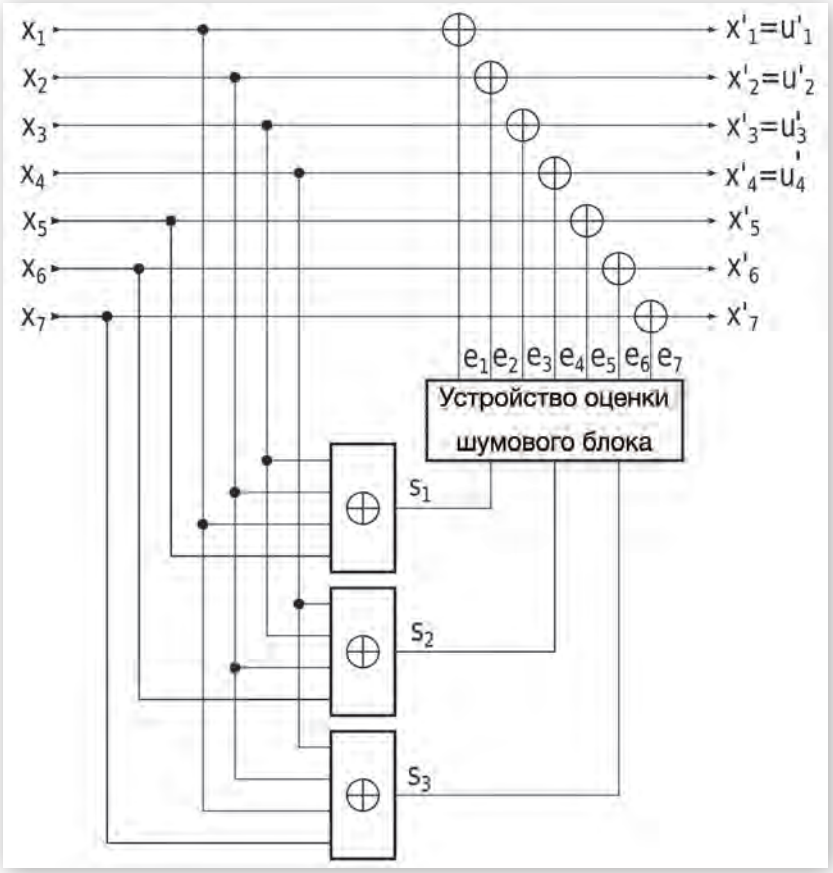

Рис. 8. Схема декодера на основе (7, 4)-кода Хэмминга

Схема кодера, в котором используется (7, 4)-код Хэмминга, показана на рис. 7. Она реализована с применением трех сумматоров по модулю 2, которые формируют проверочные символы в соответствии с соотношениями (7). Схема декодера (7, 4)-кода Хэмминга показана на рис. 8. В ней используются семь сумматоров по модулю 2, которые вычисляют три компоненты синдрома в соответствии с формулами (35). Устройство оценки шумового блока находит элементы блока шумов, которые определяются по методу наибольшего правдоподобия в соответствии с данными табл. 14. На выходе сумматоров по модулю 2 получаются оценки блока кодовых символов ***x***, отправленного в канал связи. При использовании систематического кодирования первые четыре выходных символа $(x'_1, ..., x'_4)$ представляют собой оценки символов отправленного информационного блока ***u***.

С использованием данных таб. 14, которая фактически является компактной формой таблицы декодирования, можно рассчитать вероятность правильного декодирования для двоичного симметричного канала (рис. 2, № 6/2020, стр. 6). В табл. 14 описана одна безошибочная ситуация (первая строка) и семь случаев правильного декодирования при одной ошибке в канале связи. Следуя методике, приведенной в [16], и повторяя рассуждения, сделанные при выводе формулы (5), получаем вероятность правильного декодирования P_{cor} и вероятность ошибки P_{err} при декодировании по методу наибольшего правдоподобия для (7, 4)-кода Хэмминга в двоичном симметричном канале без памяти:

$$P_{cor} = (1-\varepsilon)^7 + 7(1-\varepsilon)^6\varepsilon,$$
$$P_{err} = 1 - (1-\varepsilon)^7 - 7(1-\varepsilon)^6\varepsilon \quad (36)$$

Если принять вероятность ошибки при передаче одного символа в канале равной ε=0,01, то вероятность ошибки при использовании (7, 4)-кода Хэмминга окажется равной P_{err}=0,002.

О стратегии проектирования хороших кодов

Выдающийся ученый Клод Шеннон в работе «Математическая теория связи» [6] доказал, что с помощью канального кодирования можно обеспечить передачу данных со сколь угодно малой вероятностью ошибки, если скорость передачи данных не превышает пропускной способности канала связи. Однако Шеннон не указал, как находить алгоритм такого кодирования. История разработки практических систем кодирования, исправляющего ошибки, убедительно показала, что неотъемлемый атрибут хороших кодов – это большая длина кодового слова. Например, Роберт Галлагер, изобретатель кодов с малой плотностью проверок на четность, доказал, исследуя блоковые коды [16], что вероятность P_{err} ошибки декодирования стремится к нулю с ростом длины блока *n* при любой скорости передачи данных R_c, меньшей пропускной способности канала С:

$$P_{err} \le e^{-nE_r(R_c)},$$

где $E_r(R_c)$ – показатель экспоненты случайного кодирования, зависящий от свойств канала связи; $E_r(R_c) > 0$, при всех $R_c < C$.

Из приведенного соотношения следует, что для достижения скорости передачи данных, приближающейся к пропускной способности канала, надо увеличивать длину кодового блока *n*.

Если сообщение имеет большую длину и состоит из большого числа символов, то лучше использовать один кодовый блок большой длины, чем много коротких блоков [10]. Это связано с тем, что статистическая структура шумов и помех, вызывающих ошибки при передаче символов сообщений в канале связи, сложна. Наряду с ошибками, имеющими случайную природу и происходящими время от времени с некоторой вероятностью, существуют так называемые пакетные ошибки, представляющие собой серии ошибок, следующих друг за другом. Длина таких пакетов ошибок носит случайный характер, поэтому периодически могут происходить серии ошибок большой длины.

Сравним эффективность кодов с разной длиной кодового слова в отношении исправления пакетных ошибок. Пусть есть код, имеющий длину кодового слова, например, 2000 битов (***n=2000***). Пусть каждое слово используется для кодирования 1000 информационных символов (***k=1000***). Код способен исправлять, например, 100 ошибок в каждом кодовом слове (***t=100***). Для передачи 10000 информационных символов надо использовать 10 блоков кода с общей длиной 20000 кодовых символов. Код способен исправить до 1000 ошибок при передаче 10000 информационных символов, но только в том случае, если ошибки распределены равномерно по десяти кодовым словам. Длина исправляемой пакетной ошибки ограничена 100 битами.

Пусть создан второй код, все параметры которого увеличены в 10 раз (*n=20000, k-10000, t-1000*). 10000 информационных символов передаются с помощью одного кодового слова из 20000 битов. Декодер второго кода способен исправить 1000 ошибок при любом их распределении. Например, может быть исправлена пакетная ошибка длиной в 1000 символов, что невозможно сделать с использованием первого кода. Коды с очень длинными кодовыми словами существенно эффективнее в отношении исправления ошибок со сложной статистической структурой.

Может показаться, что достаточно задать требования к коду, например, мощность, длину кодового слова, число исправляемых ошибок, а потом найти хороший код, перебирая множество всех кодов с использованием машинного поиска. Реально ли это? Каждое кодовое слово имеет длину *n* символов. Всего должно быть найдено $M=2^k$ кодовых слов. Полное описание кода требует сформировать *nM* двоичных символов. Всего можно насчитать 2^{nM} вариантов выбора символов этой последовательности. Следовательно, существует $2^{n\cdot 2^k}$ вариантов кодирования информационных слов длиной *k* с помощью кодовых слов с длиной *n*, то есть $2^{n\cdot 2^k}$ различных кодов.

Используя данные приведенного выше примера, находим число различных кодов: $2^{20000\cdot 2^{10000}}$. Записанное число столь велико, что говорить о поиске хороших кодов путем перебора вариантов практически невозможно. Но еще более сложным является решение проблемы создания практических алгоритмов кодирования и декодирования. Код с очень большой длиной теоретически можно описать таблицами кодирования и декодирования, но построить и использовать такие таблицы в кодерах и декодерах практически невозможно.

Должна быть развита некоторая теория кодирования, позволяющая создавать хорошие коды и находить перспективные и практичные алгоритмы кодирования и декодирования. Такие коды должны обладать особой математической структурой. Наилучшие на сегодняшний день результаты достигнуты в сфере кодов, алгебраических по своей структуре, и в сфере кодов, выбираемых случайным образом. Их математическая структура используется для того, чтобы добиться практической реализуемости кодирования и декодирования при большой длине слова.

Циклические коды

Многочлены и поля Галуа

Поля Галуа на основе кольца целых чисел

Идеи теории кодирования, используемые в широко применяемых циклических кодах, основаны на арифметических системах конечных полей, или полей Галуа. Нужно напомнить, что поле – это множество элементов, замкнутое по двум операциям, называемым сложением и умножением. В ч.3 (№ 8/2020, стр. 52) было введено поле из двух элементов. Для понимания циклических кодов требуется использование полей Галуа с большим числом элементов. Число элементов в поле принято называть порядком поля. Надо иметь в виду, что порядок поля не может быть произвольным, потому что поля Галуа существуют не для любого числа элементов.

Один из способов построения поля Галуа основывается на кольце целых чисел. В кольце целых чисел не всегда возможно деление, но зато всегда возможно деление с остатком. Если остаток от деления числа *a* на *q* равен *s*, то это записывается

как $s=R_q[a]$. Если число q простое, то есть оно делится только само на себя и на единицу, то арифметику поля $GF(q)$ можно ввести как сложение и умножение по модулю числа q. Именно так были составлены правила арифметики в поле из двух элементов **GF(2)** в табл. 5 и 6 (№ 8/2020, стр. 52).

Поля Галуа на основе кольца многочленов

Другой способ построения полей Галуа основан на кольцах многочленов. Выражение вида $f(z) = f_n z^n + f_{n-1} z^{n-1} + \ldots + f_0$ называется многочленом над полем **GF(q)** степени n, если коэффициенты f_n, f_{n-1}, …, f_0 являются элементами поля **GF(q)** и первый коэффициент f_n не равен нулю. Символ z в многочлене нельзя интерпретировать как переменную или неизвестный элемент поля, он является неопределенным символом. В большинстве случаев интерес представляет не $f(z)$ как функция, а последовательность элементов f_n, f_{n-1}, …, f_0. Два многочлена называются равными, если они соответствуют одной и той же последовательности коэффициентов. Многочлен называется нормированным или приведенным, если его старший коэффициент равен единице. Многочлен, который можно представить в виде произведения многочленов низших степеней с коэффициентами из поля $GF(q)$, называется приводимым, в противном случае — неприводимым.

Легко определяется сложение, вычитание и умножение многочленов по обычным правилам сложения и умножения многочленов, как это делалось в школе, однако действия с коэффициентами многочленов над полем **GF(q)** надо выполнять в соответствии с арифметикой поля **GF(q)**. Но таким простым способом не определяется операция, обратная умножению, то есть деление многочленов. Множество многочленов вместе с операциями сложения, вычитания и умножения образует кольцо многочленов над полем **GF(q)**.

По аналогии с арифметическими действиями по модулю простого числа можно определить деление многочлена $f(z)$ на многочлен $p(z)$ как остаток от деления $f(z)$ на $p(z)$: $R_{p(z)}[f(z)]$. Этот остаток называется вычетом многочлена $f(z)$ по модулю многочлена $p(z)$. Можно также образовать кольцо многочленов по модулю нормированного многочлена $p(z)$. Это будет множество всех многочленов над полем $GF(q)$, степень которых меньше степени многочлена $p(z)$, с операциями сложения, вычитания и умножения по модулю многочлена $p(z)$. Наконец, можно ввести эквивалент простого числа. В кольце многочленов это будет простой многочлен – неприводимый многочлен со старшим коэффициентом, равным единице.

В теории полей Галуа доказывается, что кольцо многочленов по модулю нормированного многочлена $p(z)$ является полем тогда и только тогда, когда $p(z)$ – простой многочлен. Это положение открывает возможность расширения поля Галуа, то есть построения поля Галуа $GF(q^n)$, содержащего q^n элементов, если над полем Галуа $GF(q)$, в котором находится q элементов, найден простой многочлен степени n.

В качестве примера построим поле $GF(4)$, расширяя поле $GF(2)$ с использованием простого многочлена $p(z) = z^2 + z + 1$. Всего есть 4 многочлена, степень которых меньше степени $p(z)$, то есть не превышает единицы: $0, 1, z, z+1$. Эти многочлены являются элементами поля $GF(4)=\{0, 1, z, z+1\}$. Найдем таблицу сложения элементов поля: $0+0=0$, $0+1=1$, $1+1=0$, $z+0=z$, $z+1=z+1$, $z+z=0$, $z+1+0=z+1$, $z+1+1=z$, $z+1+z=1$, $z+1+z+1=0$. Так же можно найти часть таблицы умножения: $0*0=0$, $0*1=0$, $0*z=0$, $0*(z+1)=0$, $1*z=z$, $1*(z+1)=z+1$. Степень всех многочленов, полученных в результате проведенных операций, меньше степени p(z), поэтому нахождение остатка от деления на $p(z)$ фактически не требовалось. Но, например, умножение $z*z$ требует вычисления по полным правилам вычислений по модулю $p(z)$. $z*z=z^2=R_{p(z)}[z^2]=z+1$, $z*(z+1)=R_{p(z)}[z^2+z]=1$, $(z+1)*(z+1)=R_{p(z)}[z^2+z]=z$. Результаты выполненных вычислений сведены в табл. 15 и 16.

После построения арифметических таблиц поля $GF(4)$ можно добавить новые обозначения элементов поля (табл. 17). Двоичные обозначения представляют собой наборы коэффициентов многочленов первой степени, которые были введены как элементы кольца многочленов по модулю нормированного многочлена $p(z)$. Целочисленные обозначения можно ввести как десятичные эквиваленты двоичных наборов.

Из теории полей Галуа известно, что множество ненулевых элементов поля $GF(q)$ образует замкнутую группу по умножению, или мультипликативную группу. Число элементов в этой группе равно $(q-1)$. Замкнутость означает, что произведение элементов всегда является элементом группы. Можно взять некоторый элемент этой группы (например, элемент β) и умножать его на самого себя. Степени числа β будут также принадлежать группе в силу ее замкнутости. Получится ряд чисел: β, ββ = β^2, $\beta\beta^2 = \beta^3$, … и т.д. Поскольку группа ненулевых элементов поля имеет конечное число элементов, то в последовательности результатов умножения обязательно появится повторение. Число n различных элементов, которые можно получить, возводя β в степень, называется порядком элемента β. Очевидно, что $\beta^n=1$, так как именно следующее умножение на β приведет к повторению: $\beta\beta^n = \beta$.

Среди элементов поля обязательно найдется хотя бы один примитивный элемент. Примитивным элементом поля α называется такой элемент, когда все элементы, кроме нуля, могут быть представлены в виде степени элемента α. Это значит, что порядок примитивного элемента равен $(q-1)$. Примитивные элементы удобны тем, что можно легко построить таблицу умножения в поле, если найден примитивный элемент.

При построении расширения поля в виде множества многочленов удобно выбирать такой простой многочлен, чтобы многочлену z соответствовал примитивный элемент поля α. Специальные простые многочлены, которые позволяют добиться этого, называются примитивными. Простой многочлен $p(z) = z^2 + z + 1$ является примитивным, поэтому примитивный элемент α поля **GF(4)** в табл. 17 соответствует элементу z. Каждому элементу поля, кроме нулевого, соответствует некоторая степень примитивного элемента. Эти степени показаны как степенные обозначения элементов поля **GF(4)** (табл. 17).

Рассмотренный способ расширения поля Галуа как кольца многочленов по модулю простого многочлена достаточен для получения всех конечных полей. Подводя итог, следует отметить несколько важных свойств конечных полей:

- число элементов любого поля Галуа равно целой степени простого числа;
- для любого простого числа p и целого положительного числа m наименьшим подполем поля **GF(p^m)** является поле **GF(p)**. Число p называют характеристикой поля **GF(p^m);**
- в поле Галуа с характеристикой $p=2$ для каждого элемента β выполняется равенство: $-\beta=\beta$.

Продолжение следует

Таблица 15. Сложение в поле GF(4)

+	0	1	z	z+1
0	0	1	z	z+1
1	1	0	z+1	z
z	Z	z+1	0	1
z+1	z+1	z	1	0

Таблица 16. Умножение в поле GF(4)

×	0	1	z	z+1
0	0	0	0	0
1	0	1	z	z+1
z	0	z	z+1	1
z+1	0	z+1	1	z+1

Таблица 17. Элементы поля GF(4)

Многочленные обозначения	Двоичные обозначения	Целочисленные обозначения	Степенные обозначения
0	00	0	0
1	01	1	α^0
z	10	2	α^1
z+1	11	3	α^2