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

Часть 3.

Константин Гласман

Линейные коды

Элементы алгебры

Конечные поля

Многие хорошие коды, широко применяемые на практике, основаны на алгебраических структурах. Такие коды имеют особые структурные закономерности, обеспечивающие возможность практической реализации операций кодирования и декодирования без составления таблиц кодирования и декодирования. Первым шагом к пониманию таких кодов является определение арифметических операций, которые могут выполняться с двоичными символами.

Арифметические системы, которые вводятся в современной алгебре, подчиняются определенным правилам. Эти правила часто применимы и к обычным числовым системам и состоят из множеств и операций над элементами этих множеств.

Группа – это система, в которой определены одна основная операция и операция, ей обратная. Например, это могут быть операции сложения и вычитания, умножения и деления.

Кольцо – это система, в которой определены две основные операции – сложение и умножение, и операция, обратная сложению – вычитание.

Поле – это система, в которой определены две основные операции и обратные операции для каждой из основных: сложение, умножение, вычитание, деление.

Вещественные числа образуют поле – множество математических объектов, которые можно складывать, умножать, вычитать и делить по правилам обычной арифметики. Это поле содержит бесконечное множество элементов. Арифметические системы, используемые в теории кодирования, содержат конечное число элементов. Такие поля называют конечными. Правила обычной арифметики вещественных чисел к ним неприменимы.

Минимальное число элементов, образующих конечное поле, равно 2. Это связано с тем, что в поле должны быть два единичных элемента: *0* относительно сложения и *1* относительно умножения. Правила сложения и умножения в поле из двух элементов приведены в табл. 5 и 6.

Табл. 5. Правила сложения в поле GF(2)

+	0	1
0	0	1
1	1	0

Табл.6. Правила умножения в поле GF(2)

×	0	1
0	0	0
1	0	1

Операции в табл. 5 и 6 являются сложением и умножением по модулю 2. Результат операции представляет собой остаток от деления на 2 результата операции в соответствии с правилами поля вещественных чисел, то есть правилами обычной арифметики. Из равенства *1+1=0* следует, что *-1=1*, то есть вычитание эквивалентно сложению. Из равенства *1×1=1* следует, что $1^{-1}=1$, то есть величина, обратная 1, равна 1. Таким образом всегда определены вычитание и деление за исключением деления на нуль. Множество из двух символов вместе со сложением и умножением по модулю 2 называется полем Галуа из двух элементов – ***GF(2)***.

Многие коды основаны на идеях проверки на четность. В ч. 2 статьи, опубликованной в предыдущем номере, используется одна проверка на четность. Слово на выходе кодера *x* формируется путем добавления к информационным символам одного проверочного символа *p* таким образом, чтобы число единиц в каждом кодовом слове было четным. Добавляемый символ *p* был назван битом проверки на четность. Если использовать правила арифметики целых чисел из поля *GF(2)*, то бит *p* можно ввести как сумму символов информационного слова:

$$p = u_1 + u_2.$$

Векторные пространства

Пример векторного пространства дает трехмерное евклидово пространство, хорошо известное даже школьникам. Векторная величина – это направленный отрезок, который зависит от двух элементов разной природы: алгебраического элемента – числа, измеряющего длину, или модуль вектора, и геометрического элемента – направления вектора. Величины являются скалярными (скалярами), если они полностью характеризуются одним числом. Вектор ***a*** в трехмерном пространстве (рис. 3) можно задать с помощью трех алгебраических элементов, или чисел – проекций вектора на оси координат $\boldsymbol{a=(a_x, a_y, a_z)}$.

Выше было введено поле Галуа из двух элементов ***GF(2)***. Элементы из этого поля *0* и *1* являются скалярами. В предыдущих частях уже рассматривались наборы, или последовательности символов, называемые словами. Это упорядоченные последовательности из *n* элементов поля, обозначаемые как $(a_1, a_2, \ldots, a_n)$. Наборы являются векторами, а множество всех наборов образует векторное пространство, если для пар наборов определена операция сложения, а для набора и элемента поля определена операция умножения набора на этот элемент – скаляр. Обязательное требование – результат операций дает элемент из множества наборов. Сложение наборов длины *n* определяется как покомпонентное, или поразрядное сложение:

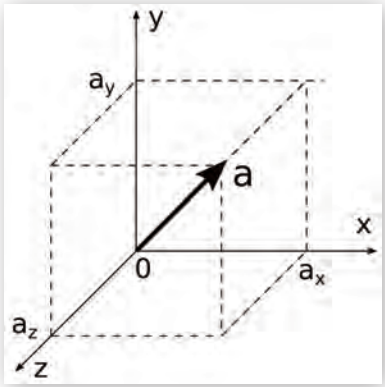

Рис. 3. Вектор в трехмерном пространстве

$$(a_1, a_2, \ldots, a_n) + (b_1, b_2, \ldots, b_n) = (a_1 + b_1, a_2 + b_2, \ldots, a_n + b_n)$$

Умножение набора на скаляр определяется следующим образом:

$$c\,(a_1, a_2, \ldots, a_n) = (ca_1, ca_2, \ldots, ca_n).$$

Для векторов также определено скалярное произведение:

$$\boldsymbol{a*b} = (a_1, a_2, \ldots, a_n)*(b_1, b_2, \ldots, b_n) = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n.$$

Подмножество векторного пространства образует подпространство, если оно удовлетворяет тем же требованиям. Практически достаточно проверить замкнутость подмножества относительно операций сложения и умножения на скаляр.

Ниже приведено несколько важных положений, которые будут использоваться далее.

Линейной комбинацией векторов $\boldsymbol{v_1, v_2, \ldots, v_k}$ называется сумма вида:

$$\boldsymbol{u} = a_1\boldsymbol{v_1} + a_2\boldsymbol{v_2} + \ldots + a_k\boldsymbol{v_k},$$

где $a_1, a_2, \ldots, a_k$ – скаляры, то есть элементы поля.

Совокупность всех линейных комбинаций некоторого набора векторов из векторного пространства ***V*** является подпространством пространства ***V***.

Совокупность векторов $v_1, v_2, \ldots, v_k$ называется линейно зависимой, если существуют такие скаляры $c_1, c_2, \ldots, c_k$ (не все равные нулю), что

$$c_1v_1 + c_2v_2 + \ldots + c_kv_k = 0.$$

Совокупность векторов называется линейно независимой, если она не является линейно зависимой.

Совокупность линейно независимых векторов порождает векторное пространство, если каждый вектор векторного пространства может быть представлен в виде линейной комбинации векторов этой совокупности. Эта совокупность называется базисом пространства. Число линейно независимых векторов, порождающих пространство, называется размерностью пространства.

Последовательность символов u_1, u_2 в слове $u=(u_1, u_2)$ можно рассматривать как компоненты вектора в двухмерном пространстве, а само слово – как вектор в этом пространстве (рис. 4). В этом пространстве существует 4 вектора: *(0,0), (0, 1), (1, 0)* и *(1, 1)*. Четыре информационных слова, соответствующие этим векторам, использовались в разделе о коде с одной проверкой на четность (ч. 2, № 7/2020). Последовательность символов x_1, x_2, x_3 в слове $x=(x_1, x_2, x_3)$ можно рассматривать как компоненты вектора в трехмерном пространстве, а само слово – как вектор в этом пространстве (рис. 5). В трехмерном пространстве существует 8 векторов. В коде раздела о коде с одной проверкой на четность четыре вектора – *(0, 0, 0), (0, 1, 1), (1, 0, 1)* и *(1, 1, 0)* – были выбраны в качестве кодовых векторов. Они находятся на расстоянии 2 друг от друга и представляют собой подпространство в трехмерном пространстве всех векторов.

Линейность кодов

Код называется линейным, если сумма любых кодовых слов дает кодовое слово. Под суммированием кодовых слов понимается сложение векторов, которое выполняется как поразрядное сложение символов слов по правилам арифметики поля **GF(2)**. Нетрудно убедиться, что код с одной проверкой на четность относится к классу линейных кодов. Например, складывая поразрядно второе и третье кодовые слова табл. 4 (011 и 101), получим четвертое слово 110: (0+1)=1, (1+0)=1, (1+1)=0. Из свойства линейности вытекает, что нулевое слово (слово, состоящее из одних нулей) входит в число кодовых слов линейного кода, так как в результате сложения слова с самим собой получается нулевое слово.

Минимальный вес

Вес Хэмминга кодового слова равен числу его ненулевых символов. Минимальный вес кода – минимальное количество ненулевых символов в ненулевом слове. Для нахождения минимального расстояния линейного кода не надо сравнивать все пары кодовых слов. Достаточно найти слова, ближайшие к нулевому слову. Для линейного кода минимальное расстояние равно минимальному весу ненулевого слова. Это означает, что надо подсчитать число ненулевых символов в кодовых словах, ближайших к нулевому. Нетрудно убедиться, что минимальный вес кода с одной проверкой на четность равен 2, следовательно, минимальное расстояние $d^*=2$. С помощью кода с одной проверкой на четность можно обнаружить одну ошибку в каждом кодовом слове. Как видно, оценка метрических свойств линейных кодов проще, чем нелинейных.

Граница Синглтона

Для линейных кодов можно получить простое неравенство, которое связывает параметры кода. Оно известно как граница Синглтона. Минимальное расстояние для любого линейного *(n, k)*-кода удовлетворяет неравенству:

$$d^* \le n - k + 1.$$

Это неравенство легко объяснить для систематического кода. Существуют слова систематического кода с одним ненулевым информационным символом и *(n-k)* проверочными символами. Если даже все проверочные символы равны единице, то вес такого кодового слова не может быть больше, чем *(n-k+1)*. Следовательно, минимальный вес кода не может быть больше, чем *(n-k+1)*. Как отмечалось выше, минимальное расстояние линейного кода равно минимальному весу. Поэтому минимальное расстояние кода не может быть больше, чем *(n-k+1)*. Каждый линейный код эквивалентен систематическому линейному коду. Следовательно, для любого линейного кода минимальное расстояние $d^* \le n - k + 1$.

Из границы Синглтона следует, что для исправления *t* ошибок код должен предусматривать не менее *2t* проверочных символов. На исправление одной ошибки надо иметь не менее двух проверочных символов. Граница Синглтона – это верхняя граница. Многие коды, признанные хорошими, имеют минимальное расстояние, значительно меньшее, чем дает граница. Но есть коды, параметры которых удовлетворяют границе Синглтона с равенством. Такой код называют кодом с максимальным расстоянием. Код с максимальным расстоянием предусматривает ровно *2t* проверочных символов для исправления *t* ошибок. К кодам с максимальным расстоянием относятся коды Рида-Соломона.

Стандартное расположение

Описать стандартное расположение можно на примере нового кода. Коды с одной проверкой на четность (ч. 2, № 7/2020) и линейные коды являются в определенном смысле антиподами. Код с повторением, определяемый в общем виде как *(n, 1)*-код, обладает минимальным расстоянием, равным *n*, и отличными возможностями для исправления ошибок в канале связи, но его скорость минимальна и равна *1/n*. Код с одной проверкой на четность, определяемый в общем виде как *(n, n-1)*-код, обладает максимальной скоростью, равной *n/(n-1)*. Но его минимальное расстояние равно *2* и он может только обнаруживать одну ошибку. При *n=3* код с повторением обладает минимальным расстоянием $d^*=3$ и скоростью $R=1/3$. Код с одной проверкой на четность при *n=3* обладает минимальным расстоянием $d^*=2$ и скоростью $R=2/3$. При *n=7* код с повторением обладает минимальным расстоянием $d^*=7$ и скоростью $R=1/7$. Код с одной проверкой на четность при *n=7* обладает минимальным расстоянием $d^*=2$ и скоростью $R=6/7$.

Как отыскать «хороший» код, находящийся где-то посередине и обладающий одновременно и неплохими возможностями для исправления ошибок и не очень низкой скоростью? Можно увеличивать число проверочных символов, подбирая их таким образом, чтобы кодовые слова отличались друг от друга в максимальном числе позиций. Такой подход реализован в систематическом коде мощности *M=4* с параметрами *(n, k)=(5, 2)*, который используется для представления двухбитовых информационных слов с помощью пятибитовых кодовых слов. Информационное слово ***u*** поступает на вход канального кодера. Слово на выходе кодера ***x*** формируется с помощью добавления трех проверочных символов (табл. 7).

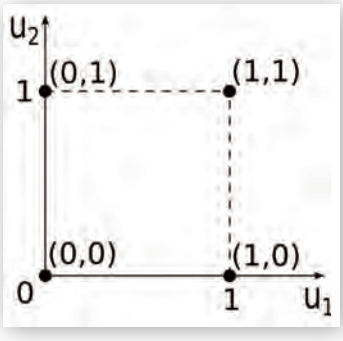

Рис. 4. Информационные слова как векторы в двухмерном пространстве

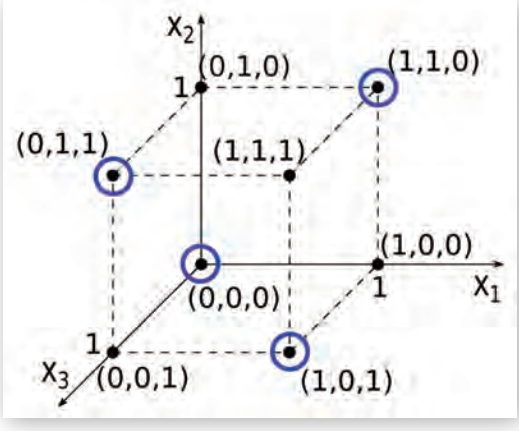

Рис. 5. Подпространство кодовых векторов в трехмерном пространстве

Табл.7. Кодирование для (5, 2)-кода

Информационные слова u	00	01	10	11
Кодовые слова x	00000	01011	10101	11110

Минимальное расстояние кода находится путем сравнения пары кодовых слов:

d(00000, 01011)=3, d(00000, 10101)=3, d(00000, 11110)=4,
d(01011, 10101)=4, d(01011, 11110)=3, d(10101, 11110)=3.

Как видно, минимальное расстояние $d^*=3$, и код способен исправлять одну ошибку. Можно было бы подойти к определению минимального расстояния другим способом. Сложение кодовых слов, как это было описано выше, позволяет оценить линейность рассматриваемого кода. Например, сумма кодовых слов второго и третьего столбцов табл. 7 дает кодовое слово четвертого столбца. Аналогичные результаты дает сложение других кодовых слов. Сложение кодового слова с самим собой дает нулевое слово, которое есть в таблице кодирования. Убедившись в линейности кода, можно было бы оценить минимальный вес ненулевого кодового слова, который тоже равен 3.

Табл. 8. Стандартное расположение для (5, 2)-кода

Кодовые слова	00000	01011	10101	11110
Смежные классы	00001	01010	10100	11111
	00010	01001	10111	11100
	00100	01111	10001	11010
	01000	00011	11101	10110
	10000	11011	00101	01110
	11000	10011	01101	00110
	01100	00111	11001	10010

На вход декодера будут поступать слова длиной 5 символов. Среди этих слов будут и кодовые слова, и другие слова, которые трансформировались из кодовых вследствие шумов и искажений. Надо описать решения, которые должен принимать декодер в ответ на каждое из 32 слов, которые могут появиться на входе декодера, то есть составить таблицу декодирования. Целесообразным способом составления таблицы является стандартное расположение [10].

В первой строке стандартного расположения (табл. 8) находятся все кодовые слова, начиная с нулевого (в общем случае это q^k кодовых слов (n, k)-кода, обозначаемые как 0, $x_1, x_2, \ldots, x_{q^k}$). Из оставшихся слов выбираем любое слово (обозначим его y_1), имеющее единичный вес и находящееся на расстоянии 1 от нулевого, и записываем его в первом столбце второй строки. В остальных столбцах второй строки записываем суммы слова y_1 и кодовых слов каждого столбца. Таким же образом строятся следующие строки. На каждом шаге выбираем слово, которое является одним из ближайших к нулевому и отсутствует в предыдущих строках. Когда после очередного шага не останется незаписанных слов, процедура закончится. Каждое слово записывается в стандартное расположение только один раз.

Множество кодовых слов (первая строка табл. 8, выделенная красным цветом) можно рассматривать как подгруппу всех слов. Тогда остальные строки таблицы представляют собой смежные классы по этой подгруппе. Слова из первого столбца называют лидерами смежных классов.

Если декодер принимает слово, равное кодовому (например, 00000), то в соответствии с методом максимального правдоподобия декодер принимает решение, что было передано именно это кодовое слово (в соответствии с табл. 7 это означает, что на вход канального кодера было подано слово источника 00). Эта ситуация описывается первой строкой таблицы декодирования (табл. 8), в которой символы отмечены красным цветом.

В последующих пяти строках таблицы, отмеченных синим цветом, показаны возможные принимаемые слова, которые отличаются от кодовых слов, находящихся в первой строке соответствующего столбца, значением одного символа. Например, во втором столбце второй строки находится слово 01010, которое отличается от кодового слова во втором столбце первой строки 01011 значением последнего символа. Минимальное расстояние кода равно 3, от всех остальных кодовых слов 01010 отличается значениями двух символов. Декодер принимает решение, что на вход канала было передано слово 01011, а на вход канального кодера – информационное слово 01. Все пять слов, которые получаются в результате ошибки в одном символе, располагаются под соответствующим кодовым словом. Они декодируются в это кодовое слово.

Выше была отмечена связь между минимальным расстоянием d^* и числом исправляемых ошибок t: $d^* \geq 2t+1$. Множество слов, находящихся в столбце стандартного расположения таблицы декодирования (табл. 8) в первых шести строках и отмеченных красным и синим цветом, можно интерпретировать как сферу декодирования. Сфера декодирования – сферическая область радиуса t в пространстве слов с центром в точке, в которой располагается кодовое слово этого столбца. Расстояние в этом пространстве является расстоянием Хэмминга и измеряется в числе несовпадающих символов в соответственных позициях. Несколько условная геометрическая интерпретация сфер декодирования показана на рис. 6 в виде двумерного среза пространства слов. Кодовые слова отмечены красными точками. Остальные слова отмечены как синие точки. Сферы декодирования выделены синим цветом. Они полупрозрачны. Остаются видными слова, попавшие в сферы декодирования.

Если в канале произошло не более t ошибок, то принятое слово всегда лежит внутри некоторой сферы и декодируется правильно. Множество всех возможных принятых слов содержит сферы декодирования всех кодовых слов. Но некоторые принятые слова, содержащие более чем t ошибок, не попадут ни в одну сферу декодирования и будут находиться в промежуточных областях. В стандартном расположении (табл. 8) эти слова располагаются в двух последних строках таблицы декодирования, отмеченных черным цветом. Как декодировать эти принятые слова? На этот вопрос можно дать два ответа.

Неполный декодер декодирует только те слова, которые находятся внутри одной из сфер декодирования. Остальные принятые слова не декодируются и считаются нераспознанными словами, содержащими ошибки. Такой результат также является важным. Он означает, что декодер исправляет одну ошибку и обнаруживает некоторые конфигурации двух и более ошибок. Декодер сигнализирует, что в принятом слове обнаружена ошибка, которая может быть исправлена другими средствами.

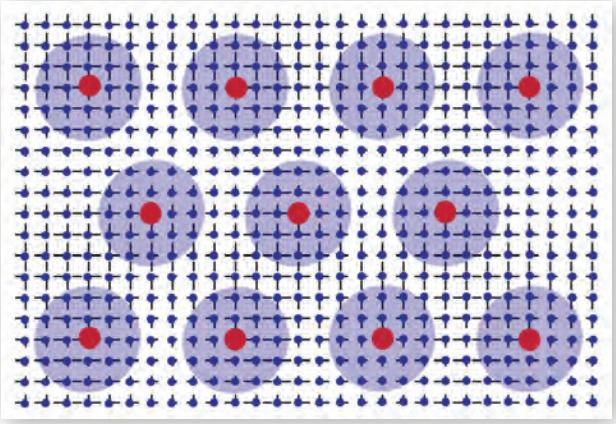

Рис. 6. Сферы декодирования (кодовые слова отмечены красными точками)

Полный декодер декодирует все слова в ближайшее кодовое слово. Если слова находятся в промежуточных областях на равных расстояниях от нескольких сфер декодирования, то одна из этих сфер объявляется ближайшей произвольно. В такой ситуации, когда происходит более t ошибок, декодер может иногда декодировать правильно, иногда – неправильно. Режим полного декодера применяется тогда, когда лучше угадывать сообщение, чем не давать никакой оценки.

Продолжение следует