УКРАИНСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ
ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА

**Н.И. Данько, С.П. Евсеев, А.А. Кузнецов,
П.Ф. Поляков, С.И. Приходько**

АЛГЕБРАИЧЕСКИЕ СВЕРТОЧНЫЕ КОДЫ

Учебное пособие

Харьков 2007

УДК 621.321

Данько Н.И., Евсеев С.П., Кузнецов А.А., Поляков П.Ф., Приходько С.И. Алгебраические сверточные коды: Учебное пособие. – Харьков: УкрГАЖТ, 2007. – 238 с.

ISBN 966-7593-63-0

Учебное пособие посвящено развитию теории кодирования информации. Разрабатываются алгебраические методы построения сверточных кодов, исследуются методы их декодирования. Исследуется эффективность применения алгебраических сверточных кодов для помехоустойчивой передачи информации. Предлагаются схемы параллельного каскадного кодирования с использованием алгебраических сверточных кодов.

Для преподавателей, научных и инженерно–технических сотрудников, аспирантов в области радиотехники, систем связи, вычислительной техники и автоматизированных систем управления, а так же для студентов старших курсов соответствующих специальностей.

Ил. 58, табл. 23, библиогр.: 60 назв.

Рекомендован Министерством образования и науки Украины как учебное пособие для студентов высших учебных заведений (№ 14/18-Г-677 от 03.05.07).

Рецензенты:

профессора С.В. Смеляков,
О.М. Фоменко (ХУПС)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Сверточные коды были предложены в 1955 году советскими учеными А.М. Финком, В.И. Шляпоберским и американским ученым П. Элайесом.

Основное отличие сверточных кодов от блочных кодов состоит в наличии функциональной зависимости проверочных символов блока от информации, содержащейся не только в данном блоке, но и в других. Это определяется непрерывностью процесса кодирования и декодирования информации сверточными кодами, что в свою очередь хорошо согласуется с последовательной и непрерывной обработкой информации.

Непрерывность процесса кодирования и декодирования информации сверточными кодами предопределила развитие теории сверточных кодов, которая развивалась по трем основным направлениям в соответствии с методами их декодирования.

Наиболее важным является существование простого способа исправления ошибок при мажоритарном декодировании, для которого не требуется использование буферных устройств, что значимо для большинства практических приложений. Кроме этого, существует возможность контроля качества порогового декодирования путем введения процедуры сигнализации об уровне ошибок, что дает возможность развития метода в направлении адаптации.

Вероятностные алгоритмы последовательного декодирования (Зигангирова–Джеленека, Фано) и субоптимальный алгоритм Витерби являются процедурами декодирования, позволяющими в любом канале получить сколь угодно малую вероятность ошибки при скоростях передачи, близких к пропускной способности канала. При этом использование алгоритма Витерби удобно согласуется с многоуровневыми модемами.

Сверточные коды позволяют осуществлять комбинированные схемы декодирования и, в частности, с блочными кодами, в целях улучшения процесса декодирования, в каналах с изменяющимися характеристиками.

Комбинированные схемы декодирования для блочных кодов, позволяющие минимизировать вероятность ошибки при конечной скорости передачи, сложно реализуются и требуют большого объема памяти.

Использование сверточных кодов позволяет снизить требования к процедуре циклового фазирования сравнительно с блочными кодами. Сверточные коды проще применять в системах с обратными связями и переспросом.

Однако при перечисленных положительных свойствах сверточных кодов следует отметить отсутствие единых теоретических положений, объединяющих процедуры их выбора и построения, а также кодирования и декодирования.

Существующие конструктивные методы построения сверточных кодов позволяют получать сверточные коды с ограниченными параметрами, причем также коды не относятся по своим характеристикам к лучшим из известных. Кроме этого, в этих методах построения не определяется взаимосвязь с процедурами декодирования сверточных кодов, за исключением алгоритма порогового декодирования.

Перспективным и недостаточно разработанным применением сверточных кодов является их использование в турбокодах, где преимущества и недостатки сверточных кодов становятся более явными.

В предлагаемой монографии на основании единого теоретического подхода к описанию, построению, кодированию и декодированию сверточных кодов, основанного на математическом аппарате высшей алгебры, решена проблема их практического применения, в том числе и в турбокодах.

РАЗДЕЛ 1
ОБЩЕТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ ИНФОРМАЦИИ

1.1. Помехоустойчивая передача данных по каналу с шумами

Под системой связи (рис 1.1) понимают совокупность технических средств, предназначенных для передачи информации, включая источник данных и получателя данных. Система связи соединяет источник данных с получателем данных посредством канала; примерами каналов являются микроволновые линии, коаксиальные кабели, телефонные сети и т.д.

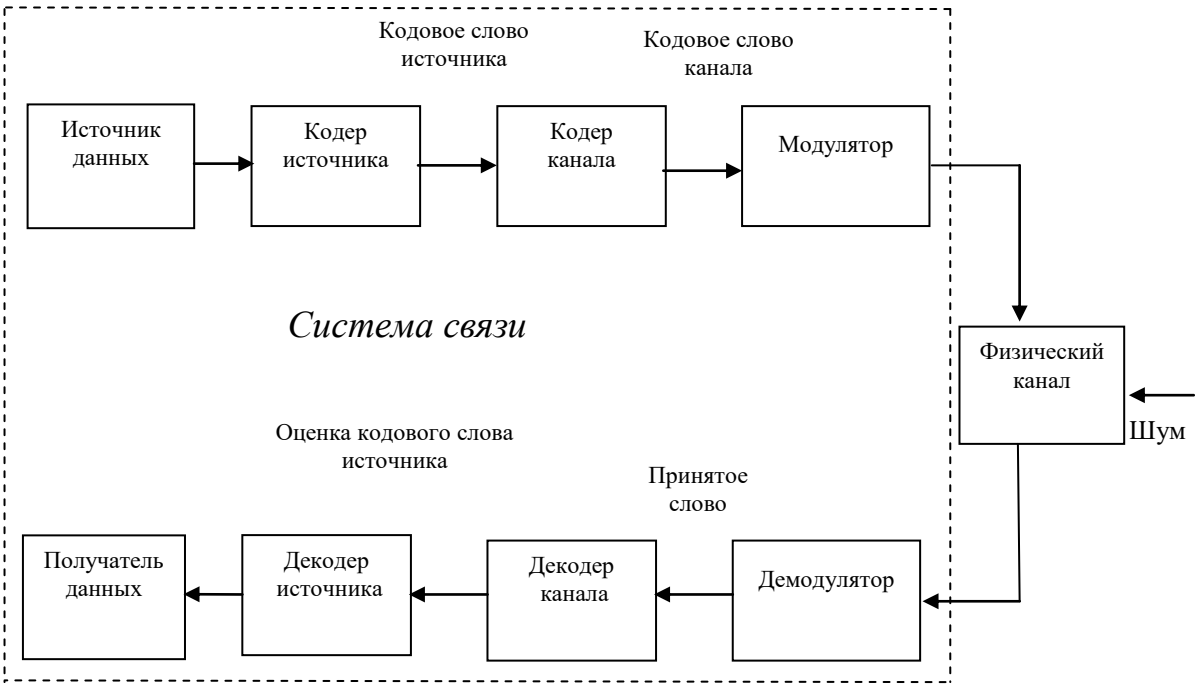


Рис. 1.1. Структурная схема системы связи

Данные, поступающие в систему связи от источника данных, прежде всего обрабатываются кодером источника, предназначенным для более компактного представления данных источника. Это промежуточное представление, закодированное неизбыточным кодом, в частном случае в виде двоичной

последовательности импульсов постоянного тока, является последовательностью символов, которая называется кодовым словом источника. Затем данные обрабатываются кодером канала, преобразующим последовательность символов кодового слова источника в другую последовательность символов, называемую кодовым словом канала. Кодовое слово канала представляет собой новую, более длинную последовательность. Каждый символ кодового слова канала может быть представлен битом или, возможно, группой битов.

Далее модулятор преобразует каждый символ кодового слова канала в соответствующий аналоговый символ из конечного множества допустимых аналоговых символов. Последовательность аналоговых символов передается по каналу.

Совокупность канала связи и устройства преобразования сигнала (модема) называют дискретным каналом. После дискретного канала связи с помощью дискретного приемника и преобразователя информация преобразуется к исходному виду.

Так как в канале возникают различного типа шумы, искажения и интерференция, то выход канала отличается от его входа, т.е. сообщение принимается с ошибкой. Демодулятор преобразует каждый полученный на выходе канала сигнал в последовательность символов одного из кодовых слов канала. Каждый принятый символ является лучшей оценкой переданного символа, но из–за шума в канале демодулятор делает ошибки. Демодулированная последовательность символов называется принятым словом. Из–за ошибок символы принятого слова не всегда соответствуют символам кодового слова канала.

Декодер канала использует избыточность кодового слова канала для того, чтобы исправить ошибки в принятом слове, и затем выдает оценку кодового слова источника. Если все ошибки исправлены, то оценка кодового слова источника совпадает с исходным кодовым словом источника. Декодер источника выполняет операцию, обратную операции кодера источника, результат которой поступает к получателю. При исправлении ошибок стремятся, чтобы ошибки были нейтрализованы полностью.

При оценке параметров системы передачи информации используют математические модели дискретных каналов связи.

Математические модели описывают последовательности ошибок в дискретных каналах связи и предназначены для аналитического решения задач, связанных с определением параметров системы. На основе проведения многочисленных исследований потоков ошибок предложен ряд математических моделей дискретных каналов связи. Наиболее простой и часто используемой моделью является модель двоичного симметричного канала без памяти (ДСК). При этом имеется в виду следующее. По каналу связи (двоичному каналу), описываемому такой моделью, передается только двоичная информация.

Вероятность ошибки в передаваемом символе не зависит от значения передаваемого элементарного символа 0 или 1, т.е. среди неправильно принятых символов одинаково часто встречаются как 1, так и 0 (канал симметричный). И, наконец, вероятности перехода для каждой пары символов постоянны и не зависят от ранее передаваемых символов (канал без памяти). Условно такой канал представлен на рис.1.2.

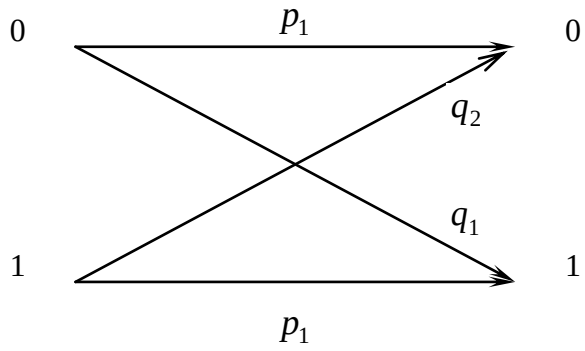


Рис. 1.2. Граф переходов для модели двоичного симметричного канала без памяти

Здесь входными являются символы 0 и 1. Возможными выходными символами будут также 0 и 1. Вероятности p_1 и p_2 определяют вероятность неискаженной передачи символов 0 и 1, а вероятности $q_1 = (1 - p_1)$ и $q_2 = (1 - p_2)$ вероятности трансформации символов. Понятно, что для ДСК $p_1 = p_2$. Таким образом, модель ДСК полностью определяется одной характеристикой – величиной $p_э$ (вероятностью искажения элементарного символа). Используя такую модель, можно достаточно просто определить вероятность возникновения в ДСК ошибок кратности t на длине из n символов.

Действительно, если вероятность ошибочного приема одного элемента равна $p_э = p_0$, то вероятность одновременного искажения элементов последовательности (так как ошибки происходят независимо друг от друга) равна p_0^t. Вероятность того, что в комбинации, имеющей n разрядов, $n-t$ элементов не искажены, составляет величину

$$p_1 = (1-p_0)^{n-t}. \qquad (1.1)$$

Тогда вероятность одновременного искажения t элементов в n–разрядной последовательности составляет величину

$$p_2 = p_0^t(1-p_0)^{n-t}. \qquad (1.2)$$

Общее количество возможных комбинаций t–кратных ошибок в n–разрядной комбинации равно

$$N_t = C_n^t = \frac{n!}{t!(n-t)!}, \qquad (1.3)$$

откуда вероятность наличия в комбинации из n разрядов t–кратной ошибки определяется по формуле

$$p(t,n) = C_n^t p_0^t (1-p_0)^{m-t}. \qquad (1.4)$$

При оценке системы передачи информации необходимо прежде всего исходить из того, какую точность передачи сообщений она обеспечивает и с какой скоростью передается информация.

Верность передачи информации является одной из важнейших характеристик системы передачи дискретной информации. Она может быть охарактеризована вероятностью ошибочного декодирования кодовой комбинации первичного кода. Требования к верности передачи разнообразны и зависят от назначения системы связи. В качестве нормы в общегосударственной системе передачи данных принято, что вероятность ошибки на кодовую комбинацию не должна превышать 10^{-6}. В особо ответственных системах передачи

дискретных сообщений существуют еще более жесткие требования. А при использовании реальных каналов связи и простого (первичного) кода указанная верность не превышает $10^{-2} – 10^{-5}$.

Таким образом, система передачи дискретной информации должна обеспечивать существенное повышение верности передачи.

Способы повышения верности передачи весьма многочисленны и разнообразны, однако все их можно разделить на три группы.

К первой группе относятся меры эксплуатационного и профилактического характера, направленные на улучшение качества показателей каналов связи. Эти меры призваны сократить число и уменьшить интенсивность действия источников помех и искажений, вызывающих ошибки при передаче, что достигается рядом технических и организационных мероприятий: улучшением стабильности работы основных узлов системы передачи, схем резервирования; выявлением и своевременной заменой неисправного оборудования и т.п.

Ко второй группе относятся мероприятия, направленные на уменьшение вероятности ошибочного приема элементарного символа за счет повышения отношения сигнал/помеха. Этого можно достигнуть путем применения более помехоустойчивых методов модуляции, более совершенных методов приема, рационального выбора мощности, длительности или спектра сигнала.

Третья группа способов повышения верности передачи дискретной информации основана на использовании помехоустойчивых кодов, с помощью которых обнаруживаются и исправляются ошибки в принятой информации. Их можно реализовать как в системах без обратной связи, так и в системах с обратной связью, в которых имеются два направления передачи.

Одним из наиболее действенных путей повышения верности в настоящее время является использование специальных процедур, основанных на применении помехоустойчивых (корректирующих) кодов.

Целью помехоустойчивого кодирования является повышение верности передачи информации путем обнаружения и

исправления ошибок. Теоретической основой помехоустойчивого кодирования является теорема Шеннона, в которой утверждается, что и для канала с помехами всегда можно найти такую систему кодирования, при которой сообщения будут переданы со сколь угодно большой степенью верности, если только производительность источника не превышает пропускной способности канала.

Рассмотрим общую постановку задачи помехоустойчивого кодирования сообщений. От источника информации поступает последовательность элементарных сообщений a_i. Кодирование заключается в том, что последовательность символов источника заменяется последовательностью двоичных кодовых символов. Такое преобразование является взаимно однозначным, что и позволяет осуществить декодирование, т.е. восстановить сообщение по принятой кодовой комбинации.

Реальный канал может добавлять к передаваемому сообщению некоторое количество ошибок. Для обнаружения этих ошибок приемнику необходимо просто заметить, что они были добавлены, а для прямого исправления ошибок (т.е. исправление ошибок без требования повторения передачи) существует дополнительное требование – нужно определить их расположение в принятом сообщении.

Проектировщик кода не может контролировать ошибки, но у него есть возможность определить код таким образом, чтобы для большинства возникающих ошибок сообщение можно было восстановить по принятому кодовому слову. Распознавание возможно до тех пор, пока сообщения достаточно отличаются друг от друга, а число ошибок не слишком велико. Задачу формирования достаточно различающихся кодовых слов можно решить путем добавления к сообщению избыточной информации.

Примером добавок такого рода является акустический алфавит, используемый при чтении слов по слогам в радиоканалах. Многие буквы (английского) алфавита, например, *B, C* и *D,* имеют очень близкое звучание, которое можно перепутать при произношении по шумящему каналу. Чтобы было легче отличить друг от друга буквы алфавита, в акустическом алфавите для каждой буквы используются специальные кодовые слова. Кроме того, числа произносят с небольшими изменениями

и гиперболизацией слогов, например, *nine* – как *niner* и *five* – как *fife*, чтобы их было легче произносить по слогам.

При проектировании кода, исправляющего ошибки, избыточность, которая добавляется в форме дополнительных символов (в двоичном коде – это двоичные символы, или биты), следует использовать с осторожностью, чтобы она была эффективной. В частности, избыточность должна делать похожие кодовые слова более разными и давать возможность отличать ошибочную комбинацию от правильной.

Различие между кодовыми словами можно представить в виде расстояния между ними. Коррекцию ошибок следует выполнять путем просмотра расстояний от полученного кодового слова до всех возможных допустимых кодовых слов, и выбора самого близкого кодового слова. Если это сделано, то ошибки можно всегда исправить, если они разрушают кодовые слова на расстоянии, которое меньше, чем половина расстояния между двумя самыми близкими кодовыми словами, или их можно обнаружить, если они разрушают кодовые слова на расстоянии, которое меньше, чем минимальное расстояние между двумя кодовыми словами.

В общем случае для обнаружения t ошибок минимальное кодовое расстояние равно

$$d = t + 1. \tag{1.5}$$

Для кодов, только исправляющих ошибки, минимальное кодовое расстояние равно

$$d = 2r + 1, \tag{1.6}$$

где r – число исправляемых ошибок.

Для определения кодового расстояния между двумя комбинациями двоичного кода достаточно просуммировать эти комбинации по модулю 2 и подсчитать число единиц в полученной комбинации.

Коррекцию и обнаружение ошибок можно выполнять одновременно. Если минимальное расстояние между любой парой кодовых слов обозначить как d_{min} (рис. 1.3), то пока

$d_{min} \geq d_d + d_c$, исправлять ошибки можно в радиусе d_c, а обнаруживать – в радиусе d_d от каждого кодового слова, т.к. если полученное сообщение находится вне изображенных кругов, то можно лишь узнать, что имеется ошибка, хотя ее нельзя "исправить", потому что она не находится в круге радиуса d_c вокруг любого кодового слова. Изменяя размер кругов, можно поменять местами возможности исправления и обнаружения ошибок. Заметим, что для фиксированного d_{min} увеличение d_c уменьшает d_d, и появляется еще одна проблема – возможность неправильного "исправления" полученного сообщения. Если сообщение было так сильно разрушено, что переместилось более чем на d_d и попало в круг радиуса d_c другого кодового слова, то оно будет декодировано неправильно.

Математическим аналогом расстояния является метрика. Самый простой и наиболее общей метрикой для бинарных сигналов является расстояние Хемминга. Расстояние Хемминга между двумя битовыми потоками определяется выражением $a-b$, где a и b – это так называемые веса этих битовых потоков, которые определяются числом их ненулевых компонентов.

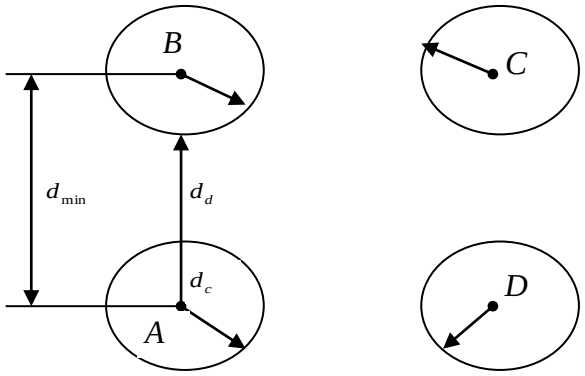


Рис. 1.3. Расстояние между кодовыми словами

Рассмотрим три кода, использующих символы, составленные из трех двоичных цифр (рис. 1.4). Первый код (рис. 1.4,а) использует все возможные комбинации трех битов и поэтому имеет минимальное расстояние 1. Он не может исправлять или обнаруживать никакие ошибки, потому что любая ошибка формировала бы новое кодовое слово.

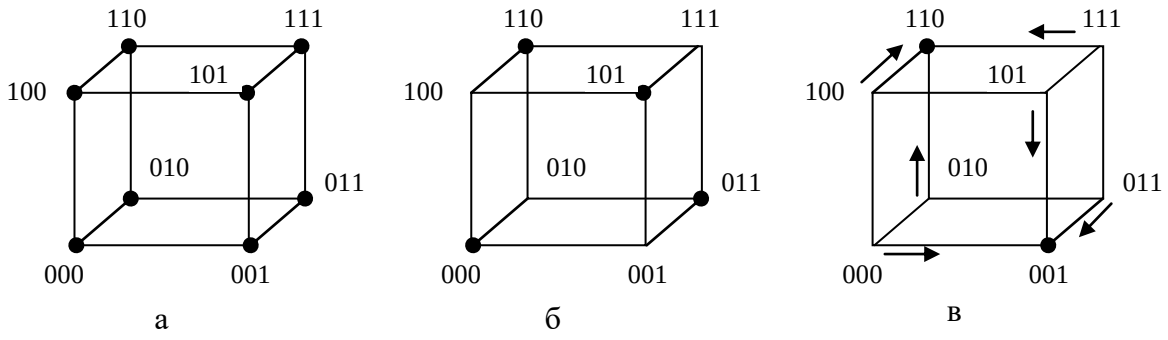


Рис. 1.4. Коды, составленные из трех двоичных цифр

Второй код (рис. 1.4,б) имеет минимальное расстояние 2. Он построен путем удаления всех кодовых слов с расстоянием 1 из кодового слова на рис. 1.4,а. Этот код может обнаруживать одиночную ошибку, хотя не исправляет ее (ошибочный код 010 мог бы сформировать кодовые слова 000, 011 или даже 110). Две ошибки не могут быть обнаружены. Код на рис. 1.4,б – это, в действительности, код проверки на четность, который обнаруживает также и три ошибки. Код (рис. 1.4,в) имеет расстояние 3. Он будет исправлять одну ошибку или обнаруживать 2 (но не одновременно). Если бы он использовался для исправления ошибок, то код 010 был бы исправлен на 000, как было бы и для кодов 100 и 001, что показано маленькими стрелками.

Существование расстояния между кодовыми словами не обязательно означает, что сообщение можно легко извлечь из полученного кодового слова, даже если случается меньше ошибок, чем код может теоретически исправить. В наихудшем случае нужно будет сравнивать полученный битовый поток со всеми возможными кодовыми словами, чтобы увидеть, к какому из них он ближе всего. Если длина потока велика, то существует много возможных кодовых слов, так что этот процесс может сильно затянуться во времени. Поэтому выбор кодирующей функции должен быть сделан так, чтобы существовал простой метод декодирования.

Известно большое количество кодов, систематизация и классификация которых из–за их многочисленных признаков являются довольно затруднительными. Поэтому в основу классификации положим структурные характеристики кодов. Коды можно разделить на две самостоятельные группы. К первой относятся коды, использующие все возможные комбинации – неизбыточные коды. В литературе их еще называют простыми или первичными. Ко второй группе относятся коды, использующие лишь определенную часть всех возможных комбинаций. Такие коды называются избыточными. Оставшаяся часть комбинаций используется для обнаружения или исправления ошибок, возникающих при передаче сообщений. В этих кодах количество разрядов кодовых комбинаций можно условно разделить на определенное число разрядов, предназначенных для информации (информационные разряды), и число разрядов, предназначенных для коррекции ошибок (проверочные разряды).

Обе группы кодов, в свою очередь, подразделяются на равномерные и неравномерные. Равномерные коды – это коды, все кодовые комбинации которых содержат постоянное количество разрядов. Неравномерные коды содержат кодовые комбинации с различным числом разрядов. Ввиду того что неравномерные избыточные коды не нашли применения на практике из–за сложности их технической реализации, их рассматривают очень редко.

Все избыточные коды разделяются на два класса: непрерывные и блочные.

В непрерывных кодах процесс кодирования и декодирования носит непрерывный характер. Этот класс кодов появился сравнительно недавно и не получил пока широкого развития. В блочных кодах каждому сообщению соответствует кодовая комбинация (блок) из n символов. Блоки кодируются и декодируются отдельно друг от друга.

Избыточные коды, в которых определенные разряды кодовых комбинаций отводятся для информационных и проверочных символов, называются разделимыми. Разделимые блочные коды обозначаются обычно (n, k) – кодами, где n – количество разрядов

кодовой комбинации, k – число разрядов, отводимых для информационных символов. Неразделимые коды не имеют четкого разделения кодовой комбинации на информационные и проверочные символы. К ним относятся коды с постоянным весом и коды Плоткина.

Разделимые блочные коды, в свою очередь, делятся на несистематические и систематические. В несистематических кодах проверочные символы представляют собой суммы подблоков с l разрядами, на которые разделена последовательность информационных символов. К этим кодам относятся коды Бергера.

Самый большой класс разделимых блочных кодов составляют систематические коды, у которых проверочные символы определяются в результате проведения линейных операций над определенными информационными символами. Для двоичных кодов эти операции сводятся к выбору каждого проверочного символа таким образом, чтобы его сумма по модулю два с определенными информационными символами была равной нулю.

К систематическим кодам относятся коды с проверкой на четность, коды с повторением, корреляционный, инверсный, коды Хэмминга, Голея, Рида–Маллера, Макдональда, Варшамова, с малой плотностью проверок на четность, итеративный код.

Разновидностью систематических кодов являются циклические коды. Кроме всех свойств систематического кода, циклические коды имеют следующее свойство: если некоторая кодовая комбинация принадлежит коду, то получающаяся путем циклической перестановки символов новая комбинация также принадлежит данному коду. К наиболее известным циклическим кодам относятся простейшие коды, коды Хэмминга, Боуза–Чоудхури–Хоквингема, мажоритарные, коды Файра, Абрамсона, Миласа–Абрамсона, Рида–Соломона, компаундные коды. Классификация рассмотренных кодов приведена на рис. 1.5.

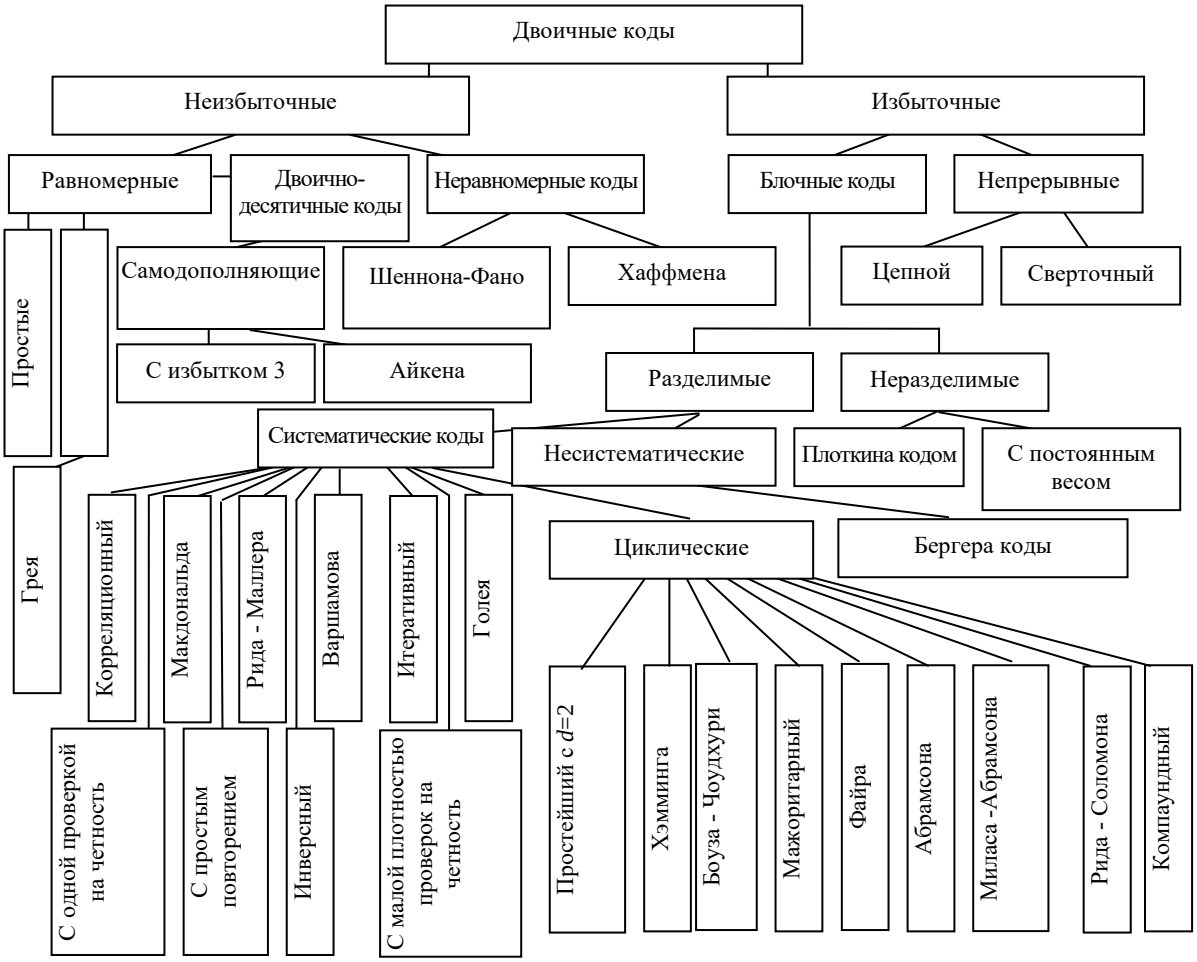


Рис. 1.5. Классификация двоичных кодов

1.2. Теорема кодирования и ее следствия

Дискретный источник информации может быть представлен как марковский процесс. Определим величину, которая будет измерять, как много информации создается таким процессом или с какой скоростью она создается.

Предположим, что имеется некоторое множество возможных событий, вероятности которых суть $p_1, p_2, \ldots, p_n$. Эти вероятности известны, но неизвестно, какое событие произойдет. Вычислим меру, насколько велик “выбор” из такого набора событий или сколь не определен его исход.

Если имеется такая мера, скажем, $H(p_1, p_2, \ldots, p_n)$, то разумно требовать, чтобы она обладала следующими свойствами:

– H должна быть непрерывной относительно p_i;

– если все p_i равны, $p_i = \frac{1}{n}$, то H должна быть монотонно возрастающей функцией от n. В случае равновероятных событий имеется больше возможностей выбора или неопределенности, чем в случае, когда имеются разновероятные события;

– если бы выбор распадался на два последовательных выбора, то первоначальная H должна была бы быть взвешенной суммой индивидуальных значений H, смысл этого иллюстрируется рис. 1.6.

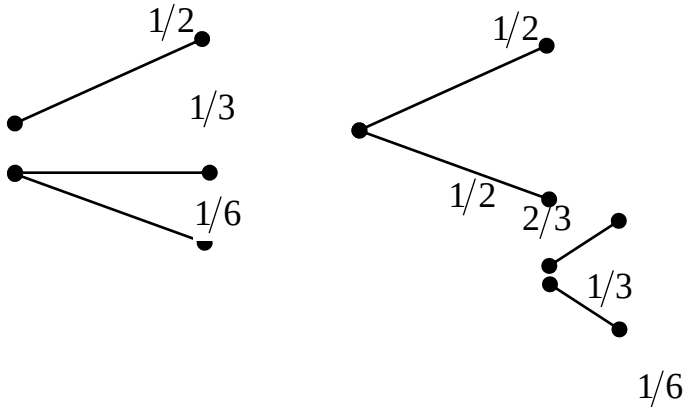


Рис. 1.6. Выбор из трех возможностей

Пример. Имеются три возможности $p_1 = 1/2, p_2 = 1/3, p_3 = 1/6$. Справа производится выбор между двумя возможностями, причем каждая имеет вероятность $1/2$, и в случае осуществления второй возможности производится еще один выбор между двумя возможностями с вероятностями $2/3, 1/3$. Окончательные результаты имеют те же самые вероятности, как и прежде. В этом конкретном случае потребуем, чтобы

$$H(\frac{1}{2},\frac{1}{3},\frac{1}{6}) = H(\frac{1}{2},\frac{1}{2}) + \frac{1}{2}H(\frac{2}{3},\frac{1}{3}).$$

Коэффициент $1/2$ является весовым множителем, введенным из–за того, что второй выбор осуществляется только в половине всех случаев.

Теорема 1.1. Существует единственная функция H, удовлетворяющая трем перечисленным выше свойствам, при этом H имеет вид

$$H = -K\sum_{i=1}^{n} p_i \log p_i , \qquad (1.7)$$

где K – некоторая положительная константа.

Эта теорема и допущения, требуемые для ее доказательства, не являются необходимыми. Они приводятся, главным образом, чтобы обосновать целесообразность некоторых дальнейших определений.

Величины вида $H = -\sum_{i=1}^{n} p_i \log p_i$ (постоянная K определяет выбор единицы измерения) играют центральную роль в теории информации в качестве *меры количества информации.* Назовем величину H энтропией множества вероятностей $p_1, p_2, \ldots, p_n$.

Величина H обладает следующими свойствами:

1. $H=0$ тогда и только тогда, когда все вероятности p_i, кроме одной, равны нулю, а эта единственная вероятность равна 1. Таким образом, H равна нулю только в случае полной определенности исхода опыта. В противном случае H положительна.

2. При заданном n величина H максимальна и равна $\log n$, когда все p_i равны (следовательно $p_i = 1/n$).

Пусть имеются два события x и y с m исходами для первого и n исходами для второго. Пусть $p(i,j)$ означает вероятность совместного осуществления исхода i для x и j для y. Энтропия совместного события равна

$$H(x,y) = -\sum_{i,j} p(i,j) Log p(i,j), \qquad (1.8)$$

где $$H(x) = -\sum_{i,j} p(i,j) \log \sum_{j} p(i,j);$$

$$H(y) = -\sum_{i,j} p(i,j) \log \sum_{i} p(i,j),$$

легко показать, что

$$H(x,y) \le H(x) + H(y). \qquad (1.9)$$

Неопределенность совместного события меньше или равна сумме неопределенностей отдельных событий.

3. Всякое изменение вероятностей $p_1, p_2, \ldots, p_n$ в сторону их выравнивания увеличивает H. В общем виде, если над вероятностями p_i произвести операцию "осреднения" вида

$$p_i' = \sum_j a_{ij} p_i , \qquad (1.10)$$

где $\sum_j a_{ij} = \sum_i a_{ij} = 1$ и все $a_{ij} \ge 0$, то H увеличивается.

4. Пусть имеются два события x и y с m исходами для первого и n исходами для второго, не обязательно независимые. Определим *условную энтропию* $H_x(y)$ величины y как величину, получаемую в результате осреднения энтропии y, вычисленной по всем значениям x. Таким образом

$$H_x(y) = -\sum_{i,j} p(i,j) \log p_i(j). \qquad (1.11)$$

Эта величина показывает, какова в среднем неопределенность значения y, когда известно значение x. Подставляя значение $p_i(j)$, получим

$$H(x,y) = H(x) + H_x(y), \qquad (1.12)$$

5. Из п.3 и 5 имеем

$$H(x) + H(y) \ge H(x,y) = H(x) + H_x(y),$$

отсюда

$$H(y) \ge H_x(y). \qquad (1.13)$$

Неопределенность события y не возрастает оттого, что событие x становится известным. Она уменьшается, если только

события *x* и *y* являются независимыми. В противном случае она не изменяется.

Рассмотрим дискретный источник с конечным числом состояний. Для каждого возможного состояния *i* имеется некоторое множество вероятностей $p_i(j)$ создания различных возможных символов *j*. Энтропия источника определяется как среднее значение величин H_i , которые усреднены в соответствии с вероятностями осуществления соответствующих событий

$$H = \sum_i P_i H_i = -\sum_{i,j} P_i(j) \log p_i(j). \qquad (1.14)$$

Это энтропия источника на символ текста.

Теорема 1.2. Пусть $p(B_i)$ – вероятность того, что источник создает последовательность символов B_i. Пусть

$$G_N = -\frac{1}{N}\sum_i p(B_i) \log p(B_i), \qquad (1.15)$$

где суммирование распространяется на все последовательности B_i, содержащие *N* символов. Тогда G_N – монотонно убывающая функция от N и

$$\lim_{N\to\infty} G_N = H. \qquad (1.16)$$

Теорема 1.3. Пусть $p(B_i, S_j)$ – вероятность того, что появится последовательность B_i и после нее появится символ S_j, а $p_{B_i}(S_j) = p(B_i, S_j)/p(B_i)$ – условная вероятность того, что после B_i появится S_j. Пусть

$$F_N = -\sum_{i,j} p(B_i, S_j) \log p_{B_i}(S_j), \qquad (1.17)$$

где суммирование проводится по всем последовательностям B_i из $(N-1)$ символа и по всем символам S_j. Тогда F_N – монотонно убывающая функция от N,

$$F_N = NG_N - (N-1)G_{N-i}, \tag{1.18}$$

$$G_N = \frac{1}{N}\sum_{i=1}^{N} F_i \ , \tag{1.19}$$

$F_N \le G_N$, и

$$\lim_{N\to\infty} F_N = H . \tag{1.20}$$

Доказательства теорем 1.2, 1.3. F_N монотонно убывает, так как увеличение N увеличивает значение условной энтропии. Простая подстановка значения $p_{B_i}(S_j)$в выражение (1.12) показывает, что

$$F_N = NG_N - (N-1)G_{N-i} .$$

Суммируя по всем N, получим

$$G_N = \frac{1}{N}\sum_{i=1}^{N} F_i .$$

Следовательно, $G_N \ge F_N$ и G_N монотонно убывают. Достоверно, что $\frac{log\frac{1}{p}}{N}$ весьма близко к H, когда N велико. Таким образом

$$\lim_{N\to\infty} F_N = H .$$

Доказательства теорем показывают, что ряд приближений к H может быть получен с помощью рассмотрения только статистической структуры последовательностей, охватывающей *1,2, ...,N* символов. F_N является лучшим приближением. На самом деле F_N является энтропией N–го приближения к источнику. Если условная вероятность появления следующего символа при условии, что значения $(N-1)$ (предшествовавших символов) известны и не изменяются от добавления сведений о любых стоящих ранее символах, то тогда $F_N = H$, тогда G_N – энтропия на символ последовательности из N символов.

Отношение энтропии источника к максимальному значению, которого могла бы достичь энтропия при тех же символах, назовем относительной энтропией источника. Единица минус относительная энтропия есть *избыточность.*

Представим математически операции, выполняемые передатчиком и приемником при кодировании и декодировании информации. И передатчик, и приемник будем называть дискретными преобразователями информации. На вход преобразователя поступает последовательность входных символов, а на выходе получается последовательность выходных символов. Преобразователь может обладать внутренней памятью, так что выходной символ зависит не только от данного входного символа, но и от всех предыдущих.

Предположим, что внутренняя память конечна, т.е. существует конечное число m возможных состояний преобразователя и при этом очередной выходной символ является функцией от соответствующего входного символа и от состояния, в котором находится преобразователь.

Если выходные символы одного преобразователя могут служить входными символами для другого, то преобразователи могут быть соединены последовательно, в результате чего также получится некоторый преобразователь. Если существует второй преобразователь, который при соединении его входа с выходом первого восстанавливает исходный входной сигнал, то первый преобразователь называется *невырожденным,* а второй – *обратным* к первому.

Теорема 1.4. Выход преобразователя с конечным числом состояний, подключенного к статистическому источнику с конечным числом состояний, представляет собой статистический источник с конечным числом состояний, причем энтропия этого источника (в единицу времени) меньше или равна энтропии источника на входе. Если преобразователь невырожденный, то энтропии равны.

Доказательство. Пусть α представляет собой состояние источника, который создает последовательность символов x_i, и пусть β – состояние преобразователя, который создает на выходе

блоки символов y_i. Система, полученная в результате соединения источника с преобразователем , может быть представлена с помощью произведения “пространства состояний”, состоящего из пар (α,β). Две точки в этом пространстве (α_1,β_1) и (α_2,β_2) соединяются линией, если источник может изменить состояние α_1 на α_2, создав при этом такой символ x, который изменит состояние преобразователя β_1 на β_2. Энтропия источника на выходе может быть вычислена как взвешенная сумм по всем состояниям. Если мы просуммируем сначала по β, то каждый получающийся в результате член будет не больше соответствующего члена для α, и, следовательно, энтропия не увеличится. Если преобразователь невырожденный, то к его выходу можно присоединить второй преобразователь, обратный первому. Если при этом H_1',H_2' и H_3' представляют соответственно энтропии источника и выходов первого и второго преобразователей, то $H_1' \geq H_2' \geq H_3' = H_1'$ и, следовательно $H_1' = H_3'$.

Теорема 1.5. Пусть система ограничений, рассматриваемая как канал, имеет пропускную способность $C = \log W$. Если положить

$$p_{ij}^{(s)} = \frac{B_j}{B_i} \cdot W^{-l_{ij}^{(s)}}, \tag{1.21}$$

где $l_{ij}^{(s)}$ – длительность s–го символа, ведущего от состояния i к состоянию j, а B_i удовлетворяет условию

$$B_i = \sum_{i,j} B_j W^{-l_{ij}^{(s)}}, \tag{1.22}$$

то энтропия H максимальна и равна C.

Доказательство. Допустим, что на данные последовательности символов наложен ряд ограничений, задающих систему с конечным числом состояний. Пусть $l_{ij}^{(s)}$ – длительности различных символов. Каково распределение вероятностей для различных состояний P_i и вероятностей $p_{ij}^{(s)}$ выбора символа s при переходе из состояния i в j, для которого

максимизируется скорость создания информации при данных ограничениях? Эти ограничения определяют дискретный канал, а максимальная скорость должна быть меньше или равна пропускной способности C этого канала. Рассматриваемая скорость равна

$$\frac{-\sum_{i,j,s} P_i p_{ij}^{(s)} \log p_{ij}^{(s)}}{\sum_{i,j,s} P_i p_{ij}^{(s)} l_{ij}^{(s)}} = \frac{N}{M}. \qquad (1.23)$$

Пусть

$$p_{ij}^{(s)} = \frac{B_j}{B_i} \cdot W^{-l_{ij}^{(s)}},$$

где B_i удовлетворяет системе уравнений

$$B_i = \sum_{i,j} B_j W^{-l_{ij}^{(s)}}.$$

Эта однородная система имеет ненулевое решение, так как W обращает в нуль детерминант из коэффициентов

$$\left| \sum_s W^{-l_{ij}^{(s)}} - \delta_{ij} \right| = 0.$$

Определенные таким образом $p_{ij}^{(s)}$ могут служить переходными вероятностями, так как

$$\sum_{j,s} p_{ij}^{(s)} = \sum_{j,s} \frac{B_j}{B_i} W^{-l_{ij}^{(s)}} = \frac{B_j}{B_i} = 1, \qquad (1.24)$$

так что сумма вероятностей выхода из любой фиксированной узловой точки равна 1.

Подставляя эти значения $p_{ij}^{(s)}$ в общее выражение для скорости (1.15), получим

$$C = \log W.$$

С помощью надлежащего выбора величин переходных вероятностей энтропия символов на выходе канала может быть доведена до максимума, являющегося пропускной способностью канала.

Подтвердим интерпретацию величин H как скорости создания информации доказательством того факта, что H определяет пропускную способность канала, необходимую при наиболее эффективном кодировании.

Теорема 1.6 (основная теорема для канала без шума) Пусть источник имеет энтропию H (бит на символ), а канал имеет пропускную способность C (бит в секунду). Тогда можно закодировать сообщения на выходе источника таким образом, чтобы передать символы по каналу со средней скоростью C/H – ε символов в одну секунду, где ε – сколь угодно мало. Передавать со средней скоростью, больше чем C/H, невозможно.

Доказательство. Обратная часть теоремы, утверждающая, что нельзя превзойти скорость C/H, может быть доказана, если заметить, что энтропия в секунду на входе канала равна энтропии источника, так как передатчик должен быть невырожденным преобразователем, и что эта энтропия не может превосходить пропускной способности канала.

Докажем первую часть теоремы двумя способами. Первый способ состоит в рассмотрении множества всех последовательностей из N символов, создаваемых источником. При большом N все эти последовательности можно разделить на две группы, одна из которых содержит меньше, чем $2^{(N+\eta)N}$ членов, а вторая содержит больше, чем 2^{RN} членов (где R – логарифм числа различных символов) и имеет суммарную вероятность меньшую, чем μ. С ростом N μ и η стремятся к нулю. Число сигналов в канале, имеющих длительность T, больше, чем $2^{(C-\theta)r}$, где θ мало, когда T велико.

Если выбрать

$$T = \left(\frac{H}{c} + \lambda \right) N , \qquad (1.25)$$

то найдется достаточное число последовательностей символов в канале для высоковероятностной группы, когда N и T достаточно велики (сколь бы малым ни было выбрано λ), и несколько добавочных последовательностей. Высоковероятностная группа произвольным, взаимнооднозначным образом кодируется в это множество последовательностей символов канала. Остающиеся последовательности источника кодируются в более длинные последовательности канала, начинающиеся и заканчивающиеся в одной из добавочных последовательностей, не использованных для высоковероятностной группы, причем эта последовательность действует как начальный и конечный сигналы, указывающие на использование в промежутке иного кода. Между этими сигналами оставляют временной интервал, достаточный для того, чтобы для этого временного интервала в канале существовало достаточно различных последовательностей для всех маловероятностных сообщений. Для этого требуется, чтобы такой временной интервал равнялся

$$T_i = \left(\frac{R}{C} + \varphi\right) N, \tag{1.26}$$

где φ мало. Средняя скорость передачи символов сообщения в 1 секунду будет тогда больше, чем

$$\left[(1-\delta)\frac{T}{N} + \delta\frac{T_i}{N}\right]^{-1} = \left[(1-\delta)\left(\frac{H}{C} + \lambda\right) + \delta\left(\frac{R}{C} + \varphi\right)\right]^{-1}. \tag{1.27}$$

При увеличении N величины δ, λ, φ стремятся к нулю, и скорость стремится к C/N.

Другой способ доказательства теоремы можно описать следующим образом: расположим сообщения длины N в порядке убывания их вероятностей. Пусть эти вероятности будут $p_1 \geq p_2 \geq \ldots \geq p_n$.

Пусть

$$p_s = \sum_{i=1}^{s-1} p_i, \tag{1.28}$$

где p_s – накопленная вероятность.

Закодируем сначала все сообщения в двоичную систему. Двоичный код для сообщения S получается путем разложения p_s как двоичного числа. Разложение проводится до m_s–й позиции, где m_s – целое число, удовлетворяющее соотношению

$$log_2 \frac{1}{p_s} \le m_s < 1 + log_2 \frac{1}{p_s}. \qquad (1.29)$$

Таким образом, высоковероятные сообщения представляются короткими кодами, а маловероятные – длинными. Из этих неравенств (1.23) вытекает следующее неравенство

$$\frac{1}{2^{m_s}} \le p_s < \frac{1}{2^{m_s - 1}}. \qquad (1.30)$$

Код для p_s будет отличаться от всех последующих кодов одной или более из своих m_s позиций, так как все оставшиеся, p_i по крайней мере на 2^{-m_s} больше, поэтому их двоичные разложения отличаются от кода p_s на первых m_s позициях. Следовательно, все эти коды различны и можно восстановить сообщение по его коду. Если последовательности символов канала не являются уже двоичными последовательностями, им можно приписать двоичные числа произвольным образом и преобразовать таким образом, двоичный код в сигналы, используемые для канала.

Среднее число H_i бит, употребляемых на символ первоначального сообщения, легко оценить. Имеем

$$H_i = \frac{1}{N} \sum m_s p_s; \qquad (1.31)$$

$$\frac{1}{N} \sum \left(log_2 \frac{1}{p_s} \right) p_s \le \frac{1}{N} \sum m_s p_s < \frac{1}{N} \sum \left(1 + log_2 \frac{1}{p_s} \right) p_s$$

и поэтому

$$G_N \le H_i < G_N + \frac{1}{N}. \qquad (1.32)$$

С ростом N величина G_N стремится к H – энтропии источника, и H_i также стремится к H.

Отсюда видно, что неэффективность кодирования в случае, когда используется конечное запаздывание на N символов, не должна быть больше, чем $1/N$ плюс разность между истинной энтропией N и энтропией G_N, сосчитанной для последовательностей длины N. Поэтому исходное увеличение времени по сравнению с идеальным в процентах не превысит

$$\frac{G_N}{H} + \frac{1}{HN} - 1. \qquad (1.33)$$

Рассмотрим теперь случай, когда в процессе передачи сигнал дополняется шумом. Это означает, что принятый сигнал не обязательно совпадает с сигналом, посланным передатчиком. Рассмотрим случай, когда сигнал при передаче испытывает не всегда одинаковые изменения. В этом случае можно считать, что принятый сигнал E является функцией переданного сигнала S и второй переменной – шума N:

$$E = f(S,N). \qquad (1.34)$$

Шум рассматривается как случайная переменная, точно так же, как выше рассматривалось сообщение.

Если канал с шумом питается некоторым источником, то имеются два статистических процесса: источник и шум. Поэтому имеется несколько энтропий, которые могут быть вычислены. Во–первых, существует энтропия источника или энтропия входа канала $H(x)$ (они равны, если передатчик невырожденный).

Энтропию выхода канала, т.е. принятого сигнала и выхода, обозначим $H(y)$. В случае отсутствия шума $H(x) = H(y)$. Совместную энтропию входа выхода обозначим $H(x, y)$. Наконец, имеются две условные энтропии $H_x(y)$, $H_y(x)$ (энтропия выхода, когда вход известен, и наоборот). Эти величины связаны соотношениями

$$H(x, y) = H(x) + H_x(y) = H(y) + H_y(x). \qquad (1.35)$$

Исходя из проведенного обсуждения понятия энтропии как меры неопределенности, представляется рациональным использовать условную энтропию сообщения (при известном сигнале) как меру этой недостающей информации. Следуя этой идее, было бы можно получить скорость действительной передачи информации *R*, вычитая из скорости создания информации (т.е. энтропии источника) среднюю скорость условной энтропии

$$R=H(x)-H_y(x). \qquad (1.36)$$

Условная энтропия $H_y(x)$ для удобства будет называться *ненадежностью.*

Пропускная способность канала с шумом должна быть максимально возможной скоростью передачи, т.е. скоростью при надлежащем согласовании источника с каналом. Определим поэтому пропускную способность канала как

$$C = max\left[H(x)-H_y(x)\right], \qquad (1.37)$$

где максимум берется по всем возможным источникам информации, используемым в качестве входа в канал.

При надлежащем кодировании по каналу можно передавать информацию со скоростью *C* со сколь угодно малой частотой ошибок или сколь угодно малой ненадежностью. Это утверждение не верно, если скорость передачи больше *C*. При попытках передавать со скоростью больше *C*, скажем, $C+R_1$, неизбежно появится ненадежность, равная или большая, чем R_1. Изложенное выше показано на рис. 1.7.

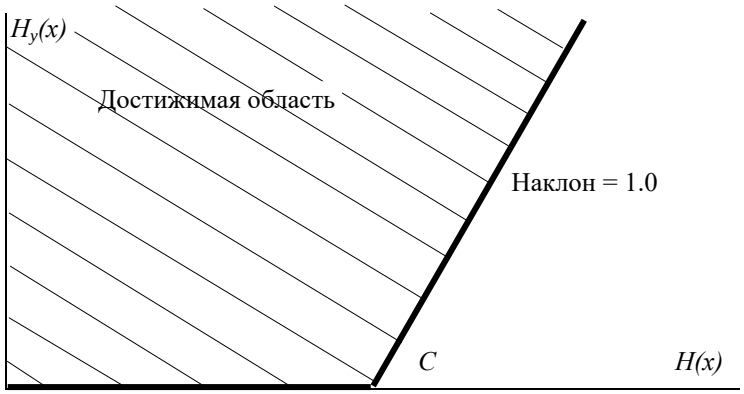


Рис. 1.7. Ненадежность, возможная для энтропии входа канала

Скорость информации в канале отложена по горизонтали, а ненадежность – по вертикали. Любая точка выше жирной линии в заштрихованной области может быть получена, точки ниже линии получены быть не могут. Точки самой линии, вообще говоря, получены быть не могут, за исключением обычно двух точек.

Теорема 1.7 (основная теорема для дискретного канала с шумом). Пусть дискретный канал обладает пропускной способностью *C*, а дискретный источник – энтропией в секунду *H*. Если $H \le C$, то существует такая система кодирования, что сообщения источника могут быть переданы по каналу с произвольно малой частотой ошибок (или со сколь угодно малой ненадежностью). Если $C > H$, то можно закодировать источник таким образом, что ненадежность будет меньше, чем $H–C+\varepsilon$, где ε сколь угодно мало. Не существует способа кодирования, обеспечивающего ненадежность, меньшую чем *H–C*.

Доказательство. Метод доказательства первой части теоремы состоит не в указании способа кодирования, имеющего желаемые свойства, а в доказательстве того, что искомый код должен существовать в определенной группе кодов. Будем усреднять по этой группе частоту ошибок и покажем, что полученное среднее может быть сделано меньше, чем ε. Но если среднее некоторого множества чисел меньше, чем ε, то в этом множестве должно существовать, по крайней мере, одно число, меньшее ε. Это и даст желаемый результат.

Пропускная способность канала с шумом определена выражением (1.31)

$$C = max\left[H(x) - H_y(x)\right],$$

где *x* – сигнал на входе канала, а *y* – на выходе канала.

Максимум берется по всем источникам, которые могут быть использованы на входе такого канала.

Пусть S_0 – источник, на котором достигается максимальная пропускная способность C. Если максимум в действительности не достигается ни на каком источнике (но достигается лишь в пределе), то пусть S_0 – источник, для которого пропускная способность достаточно близка к предельной.

Пусть S_0 используется в качестве входа канала. Рассмотрим возможные передаваемые и принимаемые последовательности большой длительности T. Можно утверждать следующее:

1. Передаваемые последовательности распадаются на два класса: высоковероятная группа, содержащая примерно $2^{NH(x)}$ членов, и остающиеся последовательности с малой суммарной вероятностью.
2. Аналогично имеется высоковероятное множество принимаемых последовательностей, содержащее примерно $2^{NH(y)}$ членов, и маловероятное множество оставшихся последовательностей.
3. Каждая из высоковероятных выходных последовательностей может быть создана одной из входных последовательностей, число которых равно примерно $2^{NH_y(x)}$. Суммарная вероятность всех других случаев мала.
4. Каждая из высоковероятных входных последовательностей может создать примерно $2^{NH_x(y)}$ выходных последовательностей. Суммарная вероятность всех других исходов мала.

В этих утверждениях все “ε” и “δ”, подразумеваемые в словах “малый” и “примерно”, стремятся к нулю, когда T возрастает и S_0 приближается к максимизирующему источнику.

Вышесказанное изображено на рис. 1.8, где входные последовательности являются точками слева, а выходные последовательности – точками справа.

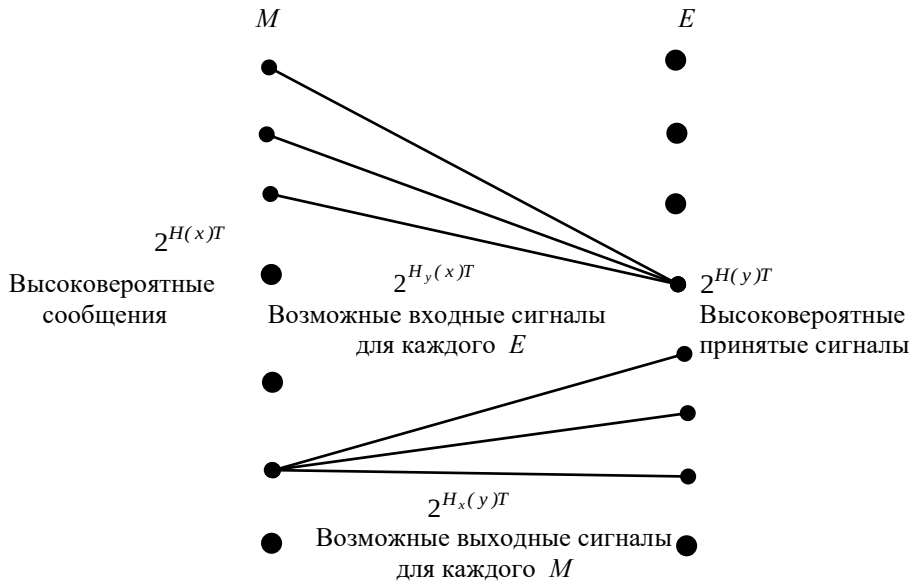


Рис. 1.8. Схематическое представление соотношений между входами и выходами в канале

Верхний веер сходящихся линий изображает ряд возможных входов типичного выхода. Нижний веер изображает ряд возможных результатов, обусловленных типичным входом. В обоих случаях не учитываются маловероятные множества.

Предположим, что имеется другой источник S, создающий информацию со скоростью R, причем $R < C$. За время T этот источник будет создавать 2^{TR} высоковероятных сообщений. Требуется связать их с некоторым выбранным множеством возможных входных сигналов канала таким образом, чтобы получить малую частоту ошибок. Это равносильно вычислению частоты ошибок при случайном связывании сообщений и входных сигналов канала длительности *T*. Предположим, что наблюдается некоторый выходной сигнал y_i. Какова вероятность того, что во множество возможных причин получения этого y_i войдет более одного сообщения от источника S. Имеется 2^{TR} сообщений, распределенных случайно по $2^{TH(x)}$ точкам. Вероятность того, что некоторая конкретная точка будет сообщением, равна

$$2^{T[R-H(x)]}. \tag{1.38}$$

Вероятность того, никакая другая точка веера (кроме действительного исходного сообщения) не будет сообщением, равна

$$P = \left[1 - 2^{T[R-H(x)]}\right] 2^{TH_y(x)}. \tag{1.39}$$

Но $R < H(x) - H_y(x)$, так что

$$R - H(x) = -H_y(x) - \eta, \tag{1.40}$$

где η положительно. Следовательно

$$P = \left[1 - 2^{-TH_y(x)-T_\eta}\right] 2^{TH_y(x)}, \tag{1.41}$$

стремится (при $T \to \infty$) к

$$1 - 2^{T_\eta}. \tag{1.42}$$

Отсюда вероятность ошибки стремится к нулю, и первая часть теоремы доказана.

Вторую часть теоремы легко доказать исходя из того, что можно просто передавать C бит в секунду от источника, совсем пренебрегая остатком создаваемой информации. В приемнике эта неучитываемая часть создаст ненадежность $H(x) - C$, а передаваемая часть должна добавить лишь ε. Эту границу можно достигнуть также многими способами.

Последнее утверждение теоремы является простым следствием нашего определения C. Предположим, что можно закодировать некоторый источник с энтропией $H(x) = C + \alpha$ таким образом, чтобы получить ненадежность, где ε положительно.

Тогда

$$H(x) - H_y(x) = C + \varepsilon, \tag{1.43}$$

где ε положительно. Это противоречит определению C как максимума

$$H(x) - H_y(x).$$

В действительности здесь доказано больше, чем в теореме. Если среднее значение множества положительных чисел заключено между нулем и ε, то только часть из них, доля которой не превышает $\sqrt{\varepsilon}$, может быть больше $\sqrt{\varepsilon}$. Так как ε произвольно мало, то можно сказать, что почти все системы кодирования сколь угодно близки к идеальным.

Доказательство теоремы 7, не будучи чистым доказательством существования, обладает некоторыми недостатками подобных доказательств. Попытка осуществить хорошее приближение к идеальному кодированию по методу, примененному в доказательстве, представляется непрактичной.

Действительно, за исключением нескольких довольно тривиальных случаев и некоторых предельных ситуаций, никакого явного описания последовательных приближений к идеальному методу не найдено.

Приближение к идеальному методу достигается ценой введения определенной избыточности в кодировании. Избыточность должна быть введена способом, приспособленным для борьбы против действующих на канал шумов определенной структуры. Впрочем, обычно будет помогать любая избыточность источника информации, если она используется в точке приема.

Как и в случае отсутствия шума, для приближения к идеальному кодированию требуется некоторая временная задержка, которая позволяет воздействовать на сигнал большой выборкой шума до того, как будет сделано какое–либо суждение в точке приема относительно исходного сообщения.

Содержание теоремы 7 и ее доказательство могут быть сформулированы несколько иным способом, который яснее выявляет связь со случаем отсутствия шума. Рассмотрим сигналы длительности T и предположим, что из них выбирается для использования некоторое подмножество. Пусть все сигналы из этого подмножества используются с одинаковой вероятностью, и допустим что приемник сконструирован так, что он выбирает в качестве исходного сигнала тот элемент из подмножества, для

которого наиболее вероятно перейти в искаженный сигнал. Обозначим через $N(T,q)$ максимальное число сигналов, которые можно выбрать для нашего подмножества так, чтобы вероятность неправильной интерпретации была меньше или равна q.

Теорема 1.8. Если q не равно нулю или единице, то

$$\lim_{T\to\infty}\frac{\log N(T,q)}{T}=C, \qquad (1.44)$$

где C – пропускная способность канала.

Доказательство. Независимо от требований надежности можно за время T уверенно различить достаточное количество сообщений, соответствующее примерно CT битам, если T достаточно велико. Теорему 8 можно сравнить с определением пропускной способности канала без шума.

Рассмотрим пример дискретного канала и его пропускной способности.

Простой пример дискретного канала показан на рис. 1.9. Имеются три возможных символа. Первый символ не подвергается воздействию шума. Второй и третий имеют вероятность p пройти неискаженными и вероятность q превратится в другой символ той же пары.

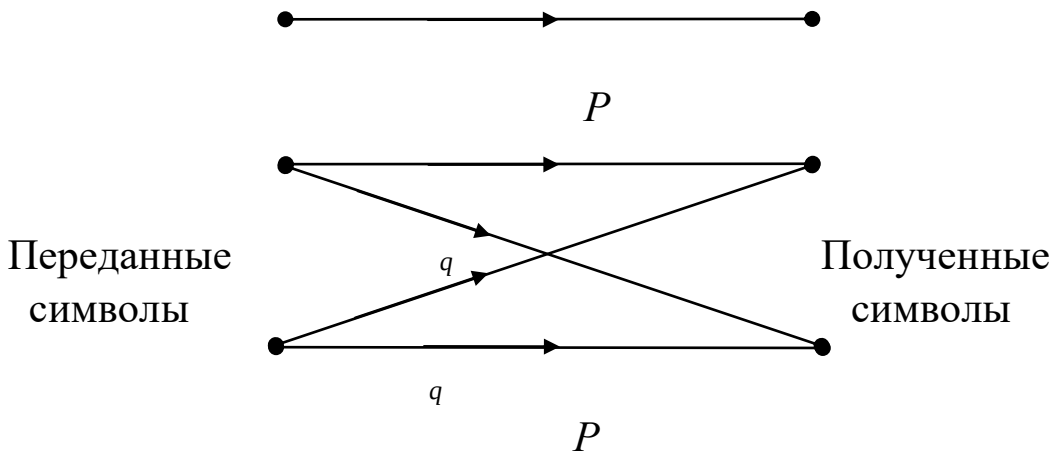


Рис. 1.9. Пример дискретного канала

Пусть $\alpha = -[p \log p + q \log q]$ и пусть P,Q – вероятности передачи соответственно первого, второго и третьего символов (при этом две последние вероятности равны по соображениям симметрии). Имеем

$$H(x) = -P \log P - 2Q \log Q, \tag{1.45}$$

$$H_y = 2Q_\alpha .$$

Требуется выбрать P,Q так, чтобы максимизировать $H(x) - H_y(x)$, соблюдая при этом условие $P + 2Q = 1$, поэтому рассмотрим

$$U = -P \log P - 2Q \log Q - 2Q\alpha + \lambda(P + 2Q), \tag{1.46}$$

$$\frac{\partial U}{\partial P} = -1 - \log P + \lambda = 0,$$

$$\frac{\partial U}{\partial Q} = -2 - 2 \log Q - 2\alpha + 2\lambda = 0 .$$

Исключая λ, получим

$$\log P = \log Q + \alpha,$$

$$P = Q \cdot 2^{\alpha} = Q\beta,$$

$$P = \frac{\beta}{\beta + 2},\ Q = \frac{1}{\beta + 2} .$$

Пропускная способность канала равна

$$C = \log \frac{\beta + 2}{\beta} . \tag{1.47}$$

Заметим, что полученные выражения дают очевидные ответы в случаях p=1 и p=1/2. В первом случае β=1 и $C = \log 3$, что правильно, так как в этом случае имеется канал без шума с тремя возможными символами. Если p=1/2, то β=2 и $C = \log 2$. Здесь

второй и третий символы могут быть отличны друг от друга и они воспринимаются как один символ. Подается первый символ и второй с третьим вместе с вероятностью 1/2. Эта вероятность может быть распределена между вторым и третьим символами произвольным образом, и при этом будет достигаться максимальная пропускная способность.

При промежуточных значениях p пропускная способность канала будет заключена между $log\,2$ и $log\,3$. Различие между вторым и третьим символами несет некоторое количество информации, но не так много, как в случае отсутствия шума. Первый символ передается чаще, чем второй и третий, так как на него не воздействует шум.

Следующий пример, хотя и несколько искусственный, представляет случай, когда возможно точное согласование передатчика информации с каналом. В канале имеются два символа 0 и 1, а шум воздействует на блоки из семи символов. Блок из семи символов либо передается без ошибок, либо в нем оказывается ошибочным ровно один символ из семи. Все эти восемь возможностей равновероятны.

Имеем

$$C = Max\left[H(y) - H_x(y)\right] = \frac{1}{7}\left[7 + \frac{8}{8} log\,\frac{1}{8}\right] = \frac{4}{7}\ \text{бит/символ.} \quad (1.48)$$

Эффективный код, обеспечивающий полную коррекцию ошибок и передачу со скоростью C, представляет собой следующее (он найден по методу, предложенному Хэммингом).

Пусть блок из семи символов будет $X_1, X_2, \ldots, X_7$. Из них X_3, X_5, X_6, X_7 – символы сообщения и выбираются произвольно источником. Остальные три символа являются избыточными и вычисляются следующим образом:

X_4 выбирается так, чтобы $\alpha = X_4 + X_5 + X_6 + X_7$;

X_2 выбирается так, чтобы $\beta = X_2 + X_3 + X_6 + X_7$;

X_1 выбирается так, чтобы $\gamma = X_1 + X_3 + X_5 + X_7$ были четными.

Когда принят блок из семи символов, вычисляются α, β, γ и, если какое–либо из них окажется четным, то считаем его нулем, если же нечетным, то единицей. Двоичное число $\alpha\beta\gamma$ даст тогда индекс того x_i, которое оказалось ошибочным (если получится 0, то блок принят без ошибок).

1.3. Кодовые границы блоковых и непрерывных кодов

Предельные возможности кодов, исправляющих ошибки, необходимо знать, во–первых, при оценке того, насколько реально используемые коды хуже «идеальных» кодов, во–вторых, при определении характеристик систем в целом и, в–третьих, для сравнения систем различных типов. В данном разделе получим простейшие верхние границы для минимального расстояния.

Вначале получим границу Чернова, которая будет необходима в дальнейшем.

Лемма 1.1. Пусть $t < (q-1)n/q$. Тогда

$$\frac{q^{n\varphi(t/n)}}{n+1} \le C_t^n (q-1)^t \le \sum_{i=0}^{t} C_i^n (q-1)^i \le q^{n\varphi(t/n)},$$

где $\varphi(x) = x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x).$

Доказательство. Пусть $A(z) = (1+(q-1)z)^n = \sum_{i=0}^{n} A_i z^i$; ясно, что $A_i = C_i^n (q-1)^i$. Если $0 < z \le 1$, то

$$\sum_{i=0}^{t} C_i^n (q-1)^i = \sum_{i=0}^{t} A_i z^{i-t} = z^{-t} \sum_{i=0}^{t} A_i z^i \le z^{-t} A(z). \qquad (1.49)$$

С другой стороны, поскольку отношение $A_i / A_{i+1} = (i+1)/[(n-i)(q-1)]$ является монотонно возрастающей функцией i при $0 \le i < n$ и, кроме того, $t < (q-1)n/q$, то найдется такое число $\bar{z}$, $0 < \bar{z} \le 1$,

$$\frac{A_{t-1}}{A_t} \le \bar{z} \le \frac{A_t}{A_{t+1}}.$$

Так как $A_{i-1}/A_i \le A_{t-1}/A_t \le \bar{z}$ при $0 \le i < t$ и $\bar{z} \le A_t/A_{t+1} \le A_i/A_{i+1}$ при $t < i$, то для всех i

$$A_t \bar{z}^t \ge A_i \bar{z}^i$$

и, следовательно, $A(\bar{z}) = \sum_{i=0}^{n} A_i \bar{z}^i \le (n+1) A_t \bar{z}^t$. Отсюда и из формулы (1.49) получаем

$$\min_{0<z\le 1} \frac{z^{-t} A(z)}{n+1} \le A_t < \sum_{i=0}^{t} A_i \le \min_{0<z\le 1} z^{-t} A(t). \qquad (1.50)$$

Положим $t/n = \tau$. Значение z, при котором достигается экстремум функции $z^{-n\tau} A(z) = z^{-n\tau}(1+(q-1)z)^n$, находится из уравнения $-n\tau z^{-n\tau-1}(1+(q-1)z)^n + nz^{-n\tau}(q-1)(1+(q-1)z)^{n-1} = 0.$

Отсюда следует, что экстремум достигается в точке $z_e = \tau/[(1-\tau)(q-1)]$. Поскольку исследуемая функция является монотонно убывающей при $0 < z < z_e$ и монотонно возрастающей при $z_e < z$, то в точке z_e она достигает минимума. Так как $t < (q-1)n/q$, то $z_e < 1$. В точке z_e рассматриваемая функция равна

$$\tau^{-n\tau}[(1-\tau)(q-1)]^{n\tau}(1-\tau)^{-n} = q^{n\varphi(\tau)}.$$

Отсюда и из формулы (1.49) получаем доказательство леммы 1.

Введенная выше функция $\varphi(x)$ при $q = 2$ совпадает с энтропией $H(x)$.

Далее найдем *верхнюю границу Хэмминга*.

Теорема 1.9. Если существует q–ичный блоковый код длины n со скоростью R и минимальным расстоянием Хэмминга $2t+1$ или более, то

$$\sum_{i=0}^{t} C_i^n (q-1)^i \le q^{n(1-R)}.$$

Доказательство. Для каждого кодового слова ϖ рассмотрим следующее множество $D(\varpi)$ последовательностей длины n:

$$D(\varpi)=\{\varpi' \mid d(\varpi,\varpi')\le t, \varpi'\in V_n\},$$

т.е. $D(\varpi)$ включает все последовательности ϖ', отличающиеся от ϖ не более чем в t компонентах. Следовательно, число последовательностей длины n, входящих в $D(\varpi)$, равно

$$\sum_{i=0}^{t} C_i^n (q-1)^i$$

Так как множества $D(\varpi)$, соответствующие различным кодовым словам, не имеют общих элементов, то

$$K\sum_{i=0}^{t} C_i^n (q-1)^i \le q^n,$$

где K – общее число кодовых слов.

Отсюда и из равенства $K=q^{nR}$ следует справедливость теоремы.

Если для некоторого кода верхняя граница Хэмминга выполняется со знаком равенства, то код называется *совершенным.* Как видно из вывода границы Хэмминга, для совершенных кодов введенные выше множества $D(\varpi)$, или, другими словами, «сферы» радиуса t, центрами которых являются кодовые слова, попарно не пересекаются и заполняют все пространство V_n. В силу этого совершенные коды иногда называют также плотноупакованными кодами.

Кроме тривиальных двоичных кодов, кодовые слова которых получаются повторением одного и того же символа некоторое фиксированное число раз, совершенными кодами являются также следующие:

1) q–ичные совершенные коды, исправляющие одиночные ошибки и имеющие длину $n=(q^l-1)/(q-1), q=p^m$,

где p – простое;

2) двоичный (23,12) – код Голея, исправляющий тройные ошибки;
3) троичный (11,6) – код Голея, исправляющий двойные ошибки.

Утверждение 1.1. Если существует q–ичный систематический совершенный код длины n, то

$$n = (q^r - 1)/(q-1),$$

где r – число проверочных символов.

Идея доказательства. Поскольку код систематический, то $R = k/n$. Далее следует воспользоваться верхней границей Хэмминга в частном случае $t = 1$.

Если q является степенью простого числа, q–ичные совершенные коды длины $n = (q^r - 1)/(q-1)$, всегда существуют.

Утверждение 1.2. Если q–ичный блоковый код длины $n = q^m - 1$ со скоростью $R \geq 1 - mt/n$ такой, что

$$\frac{(t+1)!}{(q-1)^{t+1}} + \frac{(t+1)(t+2)}{2} \leq q^m,$$

то $d_{min} \leq 2t + 2$.

Краткое доказательство:

$$C_{t+1}^{q^m-1}(q-1)^{t+1} = \frac{[(q-1)q^m]^{t+1}}{(t+1)!}\prod_{i=1}^{t+1}(1-iq^{-m}) \geq \frac{[(q-1)q^m]^{t+1}}{(t+1)!}\left(1-\sum_{i=1}^{t+1} iq^{-m}\right) =$$

$$= \frac{[(q-1)q^m]^{t+1}}{(t+1)!}\left(1-\frac{q^{-m}(t+2)(t+1)}{2}\right) \geq q^{mt} \geq q^{n(1-R)}.$$

Отсюда и из теоремы 1.9 непосредственно следует, что неравенство $d_{min} \leq 2t + 3$ не может иметь места. Это утверждение используется для получения верхней оценки минимального расстояния некоторых двоичных БЧХ–кодов.

Верхнюю границу Хэмминга можно записать следующим образом:

$$R \le 1 - \frac{1}{n} log_q \left[\sum_{i=0}^{t} C_i^n (q-1) \right].$$

Используя неравенство Чернова, можно показать, что при фиксированном отношении t/n и достаточно больших n граница Хэмминга имеет вид

$$R \le 1 - \varphi\left(\frac{t}{n}\right). \qquad (1.51)$$

Так как $\varphi'(x) = log_q(q-1) - log_q x + log_q(1-x),$ то в интервале $0 \le x \le (q-1)/q$ функция $\varphi(x)$ монотонно возрастает от 0 до 1. Поэтому для любого $R, 0 < R < 1$ существует единственное значение x, при котором $\varphi(x) = 1 - R$. Обозначим это значение x через $\delta(R)$. Из формулы (1.1) следует справедливость следующего утверждения.

Следствие 1.1. Для любого блокового кода со скоростью R и минимальным расстоянием d_{min} при достаточно больших n

$$\frac{d_{min}}{2n} \le \delta(R). \qquad (1.52)$$

В области малых значений R верхняя граница Хэмминга является довольно грубой. При малых R более точной, чем граница Хэмминга, является верхняя граница Плоткина, определяемая следующей теоремой.

Теорема 1.10. Если существует q–ичный блоковый код длины n с общим числом кодовых слов K и минимальным расстоянием d_{min}, то

$$d_{min} \le \frac{(q-1)nK}{q(K-1)}.$$

Доказательство. Эта граница получается путем оценки сверху среднего расстояния между кодовыми словами. Пусть $\varpi^{(1)}, \varpi^{(2)}, \ldots, \varpi^{(K)}$ – кодовые слова кода, существование которого

предполагается в теореме, и пусть $\varpi_m^{(i)} - m - я$ $(1 \le m \le n)$ компонента кодового слова $\varpi^{(i)}$. Сумма попарных расстояний Хэмминга между кодовыми словами определяется следующими равенствами:

$$D_t = \sum_{i=1}^{K}\sum_{j=1}^{K} d(\varpi^{(i)},\varpi^{(j)}) = \sum_{i=1}^{K}\sum_{j=1}^{K}\sum_{m=1}^{n} d(\varpi_m^{(i)},\varpi_m^{(j)}) = \sum_{m=1}^{K}\sum_{i=1}^{K}\sum_{j=1}^{K} d(\varpi_m^{(i)},\varpi_m^{(j)}),$$

где функция $d(\varpi_m^{(i)},\varpi_m^{(j)})$ равна 1, если $\varpi_m^{(i)} \neq \varpi_m^{(j)}$, и равна 0, если $\varpi_m^{(i)} = \varpi_m^{(j)}$. Пусть $J_i^{(m)}$ – число равных i символов канала среди компонент $\varpi^{(1)},\varpi^{(2)},\ldots,\varpi^{(K)}$. Тогда имеют место следующие равенства:

$$\begin{aligned} &\sum_{i=1}^{q} J_i^{(m)} = K; \\ &\sum_{i=1}^{K}\sum_{j=1}^{K} d(\varpi_m^{(i)},\varpi_m^{(j)}) = \sum_{i=1}^{q}\sum_{j=1}^{q} \delta_{ij} J_i^{(m)} J_j^{(m)}, \end{aligned} \qquad (1.53)$$

где $\delta_{ij} = 1$, если $i \neq j$, и $\delta_{ij} = 0$, если $i = j$. Обозначим через Δ_q максимум квадратичной формы $\sum_{i=1}^{q}\sum_{j=1}^{q} \delta_{ij} x_i x_j$ при следующих условиях: $\sum_{i=1}^{q} x_i = K$ и $x_i \ge 0$ $(1 \le i \le m)$ (здесь будем считать, что переменные x_i принимают действительные значения). Методом индукции по q покажем, что $\Delta_q = K^2(q-1)/q$. В случае $q-2$ справедливость последнего равенства проверяется легко. Полученные с использованием метода неопределенных множителей Лагранжа значения $x_{1,} \ldots, x_q$, при которых достигается экстремум рассматриваемой квадратичной формы, должны удовлетворять следующим условиям:

$$\sum_{j=1}^{q} \delta_{ij} x_j = \lambda; \quad 1 \le i \le q.$$

Отсюда следует, что $x_i = K/q \ \ (1 \le i \le q)$. В этой точке значение квадратичной формы равно $K^2(q-1)/q$. Поскольку рассматриваемая область значений переменных x_i является ограниченной и замкнутой, то максимум квадратичной формы достигается либо в этой точке, либо на границе. На границе, по крайней мере, одна из переменных оказывается равной нулю, так что мы переходим к случаю, когда число переменных равно $q-1$ или меньше. По предположению индукции значения функции в таких точках не превосходят $K^2(q-2)/(q-1)$. Следовательно $\Delta_q = K^2(q-1)/q$. С другой стороны, минимальное расстояние d_{min} должно удовлетворять неравенству

$$(K^2 - K)d_{min} \le D_t \le nK^2(q-1)/q,$$

так как число пар различных кодовых слов равно $K^2 - K$.

Из вывода границы Плоткина можно видеть, что кодами, минимальное расстояние которых достигает границы Плоткина, могут быть лишь коды, в которых расстояние между любыми двумя различными кодовыми словами одно и то же; такие коды называются *эквидистантными.* Примерами двоичных эквидистантных кодов являются коды, состоящие из последовательностей максимальной длины, и коды, получающиеся из матриц Адамара.

Используя идеи доказательства верхних границ Хэмминга и Плоткина, Элайс получил новую границу для минимального расстояния, которая оказалась лучше обеих исходных границ, по крайней мере, при средних скоростях. Эта граница в дальнейшем получила название *границы Элайса.* Рассмотрим q–ичный блоковый код C длины n с K кодовыми словами, скоростью R и минимальным расстоянием Хэмминга d_{min}.

Предположим, что все последовательности длины n из V_n перенумерованы каким–либо образом числами от 1 до q^n. Пусть s – произвольное целое положительное число и K_i – число кодовых слов кода C, находящихся на расстоянии Хэмминга s или менее от i–й последовательности длины n. Поскольку

каждое кодовое слово находится на расстоянии s или менее от $\sum_{i=0}^{s} C_i^n (q-1)^i$ последовательностей длины n, то

$$\sum_{i=1}^{q^n} K_i = KC_{n,q}(s), \text{ где } C_{n,q}(s) = \sum_{i=0}^{s} C_i^n (q-1)^i.$$

Учитывая, что $K = q^{nR}$, получаем из последнего равенства следующую оценку снизу для $K' = \max\limits_i K_i$:

$$K' \geq KC_{n,q}(s)/q^n = C_{n,q}(s)/q^{n(1-R)}. \qquad (1.54)$$

Предположим, что $K_j = K'$. Без ограничения общности можно считать, что $V_1 = \{0,1,\ldots,q-1\}$. Пусть $(a_1, a_2, \ldots, a_n)$ – j–я последовательность длины n. Рассмотрим далее вместо блокового кода C код

$$C' = \{(b_1 - a_1),(b_2 - a_2),\ldots,(b_n - a_n) \mid (b_1, b_2, \ldots, b_n) \in C\},$$

где знак – означает вычитание по модулю q. Если каждому кодовому слову $(b_1, b_2, \ldots, b_n)$ кода C сопоставить кодовое слово $(b_1 - a_1),(b_2 - a_2),\ldots,(b_n - a_n)$ кода C', то расстояние Хэмминга между кодовыми словами кода C будет совпадать с расстоянием Хэмминга между соответствующими словами кода C'. Следовательно, код C' имеет те же параметры n, K, d_{min}, что и код C, и содержит K' кодовых слов, находящихся на расстоянии s или менее от последовательности $(0,0,\ldots,0)$ длины n. Поэтому вместо кода C с самого начала можно было рассматривать код C', так что мы просто будем считать, что указанные K' кодовых слов существуют в коде C. Обозначим через $J_i^{(m)}$ число таких кодовых слов среди K' кодовых слов кода C, находящихся на расстоянии более или менее от последовательности $(0,0,\ldots,0)$ длины n, у которых m–я компонента равна i. Но тогда, как и при определении границы Плоткина, получаем, что сумма $D_t^{'}$ попарных расстояний Хэмминга между указанными выше кодовыми словами определяется равенством

$$D_t' = \sum_{m=1}^{n} \sum_{i=1}^{q-1} \sum_{j=0}^{q-1} \delta_{ij} J_i^{(m)} J_j^{(m)},$$

где

$$K' = \sum_{i=0}^{q-1} J_i^{(m)}; \quad 1 \le m \le n.$$

Так как расстояние Хэмминга между любым из K' кодовых слов и кодовым словом $(0, 0, \ldots, 0)$ не превышает s, то

$$\sum_{m=1}^{n} \sum_{i=1}^{q-1} J_i^{(m)} \le K' s.$$

Поскольку D_t' и d_{min}, как и раньше, связаны между собой неравенством

$$(K'^2 - K')d_{min} \le D_t', \tag{1.55}$$

то следует оценить сверху D_t'. Возьмем произвольное действительное число $y^{(m)}$, $0 \le y^{(m)} \le K'$ и найдем максимум $I = \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} \delta_{ij} x_i x_j$ при следующих ограничениях:

$$\begin{aligned} &\sum_{i=0}^{q-1} x_i = K'; \\ &\sum_{i=1}^{q-1} x_i = y^{(m)}; \\ &x_i \le 0, \quad 0 \le i < q. \end{aligned} \tag{1.56}$$

Поскольку, как следует из формул (1.56), $x_0 = K' - y^{(m)}$, то

$$I = \sum_{i=1}^{q-1} \sum_{j=1}^{q-1} \delta_{ij} x_i x_j + 2(K' - y^{(m)}) \sum_{i=1}^{q-1} x_i = \sum_{i=1}^{q-1} \sum_{j=1}^{q-1} \delta_{ij} x_i' x_j' - \frac{q-1}{q-2} \left(K' - y^{(m)}\right)^2,$$

где $x_i' = x_i + (K' - y^{(m)})/(q-2)$. Ограничения на x_i' имеют следующий вид:

$$\sum_{i=1}^{q-1} x_i' = y^{(m)} + \frac{q-1}{q-2}\left(K' - y^{(m)}\right).$$

Таким образом, задача свелась к той же самой задаче нахождения максимума квадратичной формы, что решалась и при определении границы Плоткина. Искомый максимум $I(y^{(m)})$ квадратичной формы I определяется равенством

$$I\left(y^{(m)}\right) = \frac{\left[(q-1)K' - y^{(m)}\right]^2}{(q-2)(q-1)} - \frac{q-1}{q-2}\left(K' - y^{(m)}\right)^2 = \left(2K' - \frac{q}{q-1}y^{(m)}\right)y^{(m)}.$$

Так как $\sum_{m=1}^{n} I(y^{(m)})$ является выпуклой вверх функцией $y^{(1)},\ldots,y^{(n)}$, то с помощью метода множителей Лагранжа можно показать, что максимум $\sum_{m=1}^{n} I(y^{(m)})$ при условии $\sum_{m=1}^{n} y^{(m)} \le K's$ достигается в точке

$$y^{(m)} = K's/n; \quad 1 \le m \le n$$

и равен

$$K'^2 s\left(2 - \frac{qs}{(q-1)n}\right).$$

Отсюда и из неравенств (1.37) и (1.38) получаем

$$d_{min} \le \frac{K's}{K'-1}\left(2 - \frac{qs}{(q-1)n}\right) \le \frac{sC_{n,q}(s)q^{-n(1-R)}}{C_{n,q}(s)q^{-n(1-R)} - 1}\left(2 - \frac{qs}{(q-1)n}\right).$$

Далее параметр s следует выбрать так, чтобы он минимизировал правую часть последнего неравенства. Однако, поскольку в общем случае это сделать сложно, рассмотрим здесь случай больших n. Из леммы 1 имеем

$$q^{n\varphi(s/n)}/(n+1) \le C_{n,q}(s).$$

Если выбрать отношение s/n несколько большим, чем $\delta(R)$, и фиксировать, то при $n \to \infty$ произведение

$$C_{n,q}(s)q^{-n(1-R)}$$

будет стремиться к бесконечности, так что при больших n отношение $K'/(K'-1)$ можно считать равным единице.

Теорема 1.11. Для любой скорости передачи R и любого положительного числа ε при достаточно большой длине кода n

$$\frac{d_{min}}{n} < \delta(R)\left(2 - \frac{q\delta(R)}{q-1}\right) + \varepsilon,$$

где $\delta(R)$ – функция, введенная выше в следствии 1.

Утверждение 1.3. При фиксированном отношении t/n и достаточно больших n полученная выше верхняя граница лучше верхней границы (1.35).

Утверждение 1.4. При малых скоростях R и достаточно больших n полученная выше верхняя граница и граница Плоткина почти совпадают.

Доказательство. Как указывалось выше (при введении функции $\delta(R)$), $\delta(R) \to (q-1)/q$ при $R \to 0$. С другой стороны, $K = q^{nR} \to \infty$ при $n \to \infty$.

Таким образом, получают верхние границы для произвольных блоковых кодов.

Теорема 1.12. Минимальный вес линейного (n,k) – кода C равен d тогда и только тогда, когда любые $(d-1)$ столбцов проверочной матрицы этого кода линейно независимы, но некоторые d столбцов проверочной матрицы линейно зависимы.

Прямым следствием этой теоремы является следующее.

Следствие 2. $d \le n - k + 1$.

Далее получим нижнюю границу Варшамова–Гилберта, которая гарантирует существование линейного кода с заданным числом проверочных символов $r = n - k$ и заданным минимальным расстоянием d, число информационных символов которого больше определенной величины (вначале Гилберт получил эту границу для блоковых кодов; Варшамов улучшил результат Гилберта, получив аналогичную границу для линейных кодов).

Теорема 1.13. Пусть q – степень простого числа, а r и d – некоторые целые положительные числа. Тогда существует q–ичный линейный код длины n с r проверочными символами и кодовым расстоянием, не меньшим d, параметры которого удовлетворяют неравенству

$$q^r \le \sum_{i=0}^{d-2} C_i^n (q-1)^i. \qquad (1.57)$$

Используя функцию φ из неравенства Чернова, неравенство (1.40) можно заменить следующим:

$$\frac{r}{n} \le \varphi\left(\frac{d-2}{n}\right). \qquad (1.58)$$

В частности, при $q = 2$ последнее неравенство имеет вид

$$\frac{r}{n} \le H\left(\frac{d-2}{n}\right). \qquad (1.59)$$

Доказательство. Как следует из теоремы 10, построение матрицы H, любые $d-1$ столбцов которой линейно независимы, эквивалентно построению кода с минимальным расстоянием, не меньшим d. Построим матрицу H, удовлетворяющую указанному выше свойству, следующим образом. Вначале произвольным образом выберем в множестве $V_r' = V_r$ – {нулевой вектор} 1–й столбец h_1 матрицы H.

В качестве 2–го столбца h_2 матрицы H возьмем произвольный вектор из $V_r^{'}$, не являющийся произведением ah_1, $a \in GF(q)$. Если в $V_r^{'}$ существует хотя бы один вектор, не являющийся линейной комбинацией h_1 и h_2, то выберем один из этих векторов (произвольный) в качестве h_3. Предположим, что таким образом мы выбрали j векторов h_i, $0 \le i \le j$; по построению h_i не является линейной комбинацией никаких $d-2$ или менее столбцов из $h_1, \ldots, h_{i-1}$. Поскольку i $(0 \le i \le d-2)$ векторов из $h_1, \ldots, h_j$ можно выбрать C_i^j способами, а число способов выбора i ненулевых коэффициентов линейной комбинации равно $(q-1)^i$, то число векторов, являющихся линейными комбинациями $d-2$ или менее столбцов из $h_1, \ldots, h_j$, не больше

$$\sum_{i=1}^{d-2} C_i^j (q-1)^i.$$

Если это число меньше общего числа векторов в $V_r^{'}$, то в $V_r^{'}$ существует вектор, не совпадающий ни с одной из указанных выше линейных комбинаций. Этот вектор можно выбрать в качестве h_{j+1}. К тому моменту, когда новый вектор h_{n+1} выбрать уже нельзя, общее число n выбранных ранее векторов h_i удовлетворяет неравенству (n является длиной кода)

$$\sum_{i=1}^{d-2} C_i^n (q-1)^i \ge q^r - 1.$$

Из этого неравенства и границы Чернова получаются неравенства (1.58) и (1.59).

На рис. 1.10 показано поведение верхних границ Плоткина, Хэмминга и Элайса, а также нижней границы Варшамова–Гилберта при достаточно больших n (здесь по вертикальной оси откладывается скорость кода $R = k/n$, а по горизонтальной – отношение минимального расстояния $d_{мин}$ к $2n$).

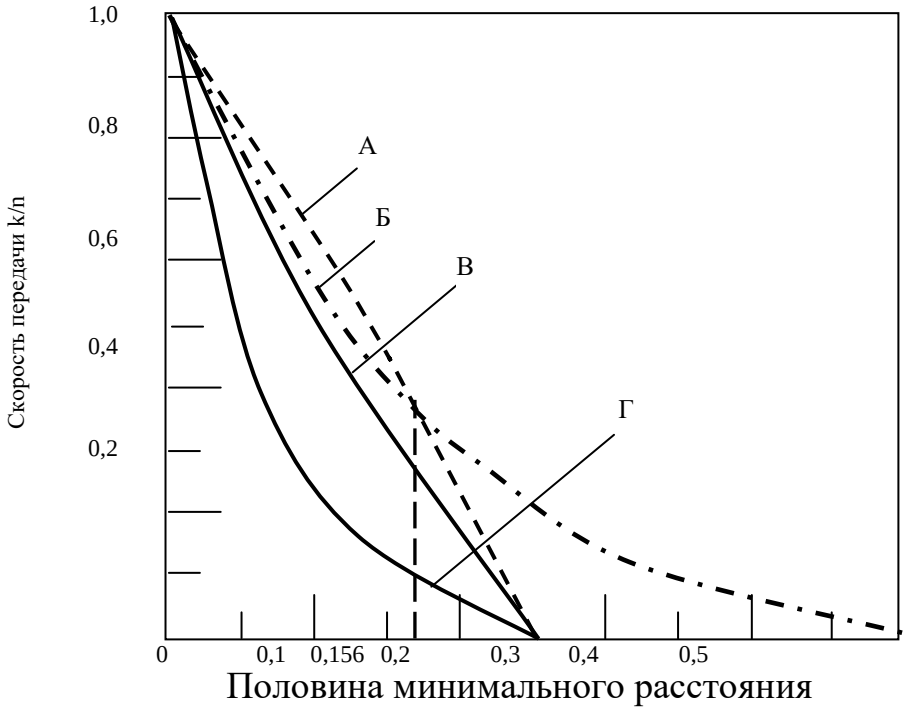


Рис. 1.10. А – верхняя граница Плоткина; Б – верхняя граница Хэмминга; В – верхняя граница Элайса; Г – нижняя граница Варшамова–Гилберта

Вывод неравенства (1.58), как мы видели выше, допускает любой способ выбора векторов $h_1,\ldots,h_i,\ldots,$, удовлетворяющих условиям теоремы 1.10. Поэтому если задан линейный (n,k)– код с минимальным весом d, параметры n,k и d которого удовлетворяют неравенству (1.58), то к проверочной матрице этого кода всегда можно добавить еще один столбец, не нарушая при этом условий теоремы 1.11. Другими словами, сохраняя постоянным и число проверочных символов, и минимальный вес, и увеличивая число информационных символов, можно расширить исходный код. Как показывают несложные расчеты, минимальный вес почти всех линейных кодов близок к величине, гарантируемой неравенством (1.58). Из неравенства (1.59) следует, что при фиксированном отношении d/n $(<1/2)$ и $n\to\infty$ существуют линейные коды, для которых отношение k/n больше $1-H(d/n)$. Здесь следует заметить, что для БЧХ–кодов, обладающих хорошими свойствами с точки зрения кодирования и декодирования, при фиксированном d/n и $n\to\infty$ отношение

k/n стремится к нулю. За исключением кодов Юстесена, все известные к настоящему времени алгебраические коды обладают этим недостатком. Однако и для кодов Юстесена отношение k/n значительно меньше величины, гарантируемой границей Варшамова–Гилберта. Построение лежащих на границе Варшамова–Гилберта (или близко к ней) кодов, способ построения которых можно было бы задать явно (конструктивно) и которые обладали бы хорошими процедурами кодирования и декодирования, является одной из нерешенных проблем теории кодирования.

Таким образом, из рассмотренного выше можно сделать вывод, что границы Хэмминга и Плоткина являются необходимыми условиями существования кода, а граница Варшамова-Гильберта – достаточным.

1.4. Некоторые простейшие коды

Групповые коды часто называются линейными или кодами с обобщенными проверками на четность. Описание кодов задается указанием алгоритма для его построения. Рассмотрим некоторые общие свойства групповых кодов.

Будем предполагать, что каждое кодовое слово группового кода разбито на две части. Первая часть, состоящая из k символов, всегда совпадает с передаваемой информационной последовательностью. Каждый из $n–k$ символов второй части вычисляется как линейная комбинация фиксированного подмножества информационных символов. Поэтому эти символы называются символами обобщенных проверок на четность или просто символами четности. Коды такого типа, в которых информационные символы при кодировании не изменяются, выше назывались *систематическими.* Можно показать, что любой групповой код можно сделать систематическим на некотором множестве из k позиций, выбрав подходящее соответствие между входными последовательностями и кодовыми словами. Это утверждение станет более ясным в дальнейшем. Его значение состоит в том, что, ограничиваясь лишь рассмотрением систематических кодов, мы не исключаем

никаких важных групповых кодов. Однако средняя вероятность ошибки при использовании систематического и эквивалентного ему несистематического кодов не должна быть одной и той же. Это также станет более ясным при дальнейшем рассмотрении.

Коды с обобщенными проверками на четность

Очень простым двоичным групповым кодом является *(n, n*–1) код, построенный с помощью одной общей проверки на четность. Например, кодовое слово (4,3)–кода может быть записано в виде вектора–столбца

$$a^T = (a_1, a_2, a_3, a_1 + a_2 + a_3), \tag{1.60}$$

где a_i принимают значения 0 или 1, а + означает сложение д$_0$ модулю 2. Отметим, что, если посимвольно прибавить к первому кодовому слову второе кодовое слово

$$b^T = (b_1, b_2, b_3, b_1 + b_2 + b_3),$$

то получим

$$c^T = a^T + b^T = (a_1 + b_1, a_2 + b_2, a_3 + b_3, a_1 + b_1 + a_2 + b_2 + a_3 + b_3) = \\ = (c_1, c_2, c_3, c_1 + c_2 + c_3). \tag{1.61}$$

Таким образом, проверочный символ в слове *c* получается точно такой же процедурой, как проверочные символы в словах *a* и *b*. Поэтому с также является кодовым словом. Этот пример показывает наиболее важное свойство групповых кодов, которое называется замкнутостью: сумма двух кодовых слов также является кодовым словом. Этот результат очевидным образом обобщается на любой групповой код.

В качестве второго примера определим кодовое слово (6,3) – кода равенством

$$a^T = (a_1, a_2, a_3, a_1 + a_2, a_2 + a_3, a_1 + a_2 + a_3). \tag{1.62}$$

Если посимвольно прибавить к *a* второе кодовое слово *b*, то получим слово *c*, в котором

$$
\begin{aligned}
c_1 &= a_1 + b_1;\\
c_2 &= a_2 + b_2;\\
c_3 &= a_3 + b_3;\\
c_4 &= a_1 + b_1 + a_2 + b_2 = c_1 + c_2;\\
c_5 &= a_2 + b_2 + a_3 + b_3 = c_2 + c_3;\\
c_6 &= a_1 + b_1 + a_2 + b_2 + a_3 + b_3 = c_1 + c_2 + c_3.
\end{aligned}
\qquad (1.63)
$$

Таким образом, три проверочных символа в слове *c* определяются точно так же, как в *a* и в *b*, поэтому с также является кодовым словом.

Из указанного свойства вытекают два важных следствия. Одно из них – существование простой процедуры для построения групповых кодов. Второе состоит в существовании связи между расстояниями в групповом коде и его весовым спектром. Этот факт позволяет значительно упростить задачу построения хороших групповых кодов, а также задачу вычисления их характеристик. Временно отложим задачу построения групповых кодов и рассмотрим свойства расстояния между кодовыми словами.

Расстояние d(a, b) между двумя кодовыми словами *a* и *b* определяется как число позиций, в которых эти слова различаются. *Вес w(c)* кодового слова определяется как число ненулевых элементов этого слова. Легко видеть, что, если рассмотреть посимвольную сумму по модулю 2 двух кодовых слов, то ее ненулевые символы соответствуют несовпадающим символам двух кодовых слов. Поэтому для любых двух кодовых слов *a* и *b* имеем

$$d(a, b)=w>(a+b). \qquad (1.64)$$

Отсюда вытекает, что множество расстояний от фиксированного кодового слова до всех других кодовых слов совпадает с множеством всех весов этого кода. Другая формулировка этого свойства состоит в том, что расстояние между двумя кодовыми словами совпадает с расстоянием от нулевого кодового слова до некоторого кодового слова. Таким образом, при построении группового кода с хорошим набором расстояний нужно стремиться к тому, чтобы веса ненулевых

кодовых слов были, возможно, большими. Кроме того, при вычислении характеристик группового кода достаточно рассматривать лишь передачу нулевого кодового слова, поскольку расстояния между любыми другими кодовыми словами будут такими же.

Своим названием групповые коды обязаны тому, что множество кодовых слов вместе с нулевым словом, снабженное операцией посимвольного сложения по модулю 2, образует математическую структуру, называемую группой. Основные свойства группы таковы:

1) сумма двух элементов группы всегда лежит в группе (замкнутость);

2) выполняется закон ассоциативности, так что $(a+b)+c= = a+(b + c);$

3) группа всегда содержит единичный элемент (нулевое слово);

4) каждый элемент группы обладает обратным (в случае двоичного кода каждое слово совпадает со своим обратным), для которого $a+ (–a) =0.$

Ясно, что коды с обобщенными проверками на четность, определенные в этом подразделе, характеризуются всеми четырьмя указанными свойствами.

Некоторые коды настолько просты, что их можно описать в самом начале.

Простые коды с проверкой на четность. Это высокоскоростные коды с плохими корректирующими характеристиками. К заданным k информационным битам дописывается $(k+1)$–й бит так, чтобы полное число единиц в кодовом слове было четным (в дальнейшем этот дополнительный бит называется битом проверки на четность).

Таким образом, например, для $k = 4$

0000↔00000;
0001↔00011;
0010↔00101;
0011↔00110

и т.д. Этот код является *(k+1, k)*–кодом или *(n, n–1)*–кодом. Минимальное расстояние кода равно двум, и, следовательно, никакие ошибки не могут быть исправлены. Простой код с проверкой на четность используется для обнаружения (но не исправления) одной ошибки.

Простые коды с повторением. Это низкоскоростные коды с хорошими корректирующими характеристиками. Один заданный информационный символ повторяется *n* раз (обычно *n* нечетно).

Таким образом,

0↔00000,
1↔11111.

Это *(n, 1)* – код. Для него минимальное расстояние равно *n*, и при предположении, что большинство принятых битов совпадает с переданным информационным битом, может быть исправлено *(m–1)/2* ошибок.

Коды Хемминга. Эти коды позволяют исправлять одну ошибку. Сейчас мы введем эти коды с помощью непосредственного описания. Для каждого *m* существует $(2^m–1, 2^m–1–m)$ – код Хемминга. При больших *m* скорость кода близка к 1, но доля общего числа битов, которые могут быть искажены, очень мала.

(7, 4) – код Хемминга можно описать с помощью приведенной на рис. 1.11 реализации. При заданных четырех информационных битах (i_1, i_2, i_3, i_4) полагаем первые четыре бита кодового слова равными этим четырем информационным битам. Дополняем тремя *проверочными битами*, задавая равенства

$$p_1 = i_1 + i_2 + i_3,$$
$$p_2 = i_2 + i_3 + i_4,$$
$$p_3 = i_1 + i_2 + i_4.$$

Здесь + обозначает сложение по модулю 2:

0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 0.

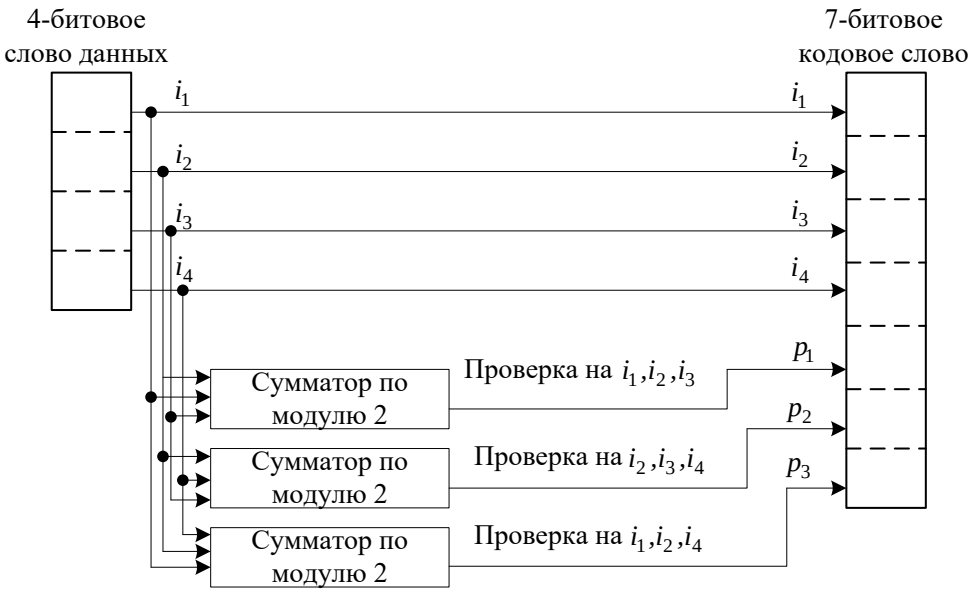


Рис. 1.10. Кодер для простого (7, 4) – кода Хемминга

Шестнадцать кодовых слов (7, 4) – кода Хемминга выписаны в табл. 1.1.

Таблица 1.1

Кодовые слова (7, 4) – кода Хемминга

0 0 0 0 0 0 0
0 0 0 1 0 1 1
0 0 1 0 1 1 0
0 0 1 1 1 0 1
0 1 0 0 1 1 1
0 1 0 1 1 0 0
0 1 1 0 0 0 1
0 1 1 1 0 1 0
1 0 0 0 1 0 1
1 0 0 1 1 1 0
1 0 1 0 0 1 1
1 0 1 1 0 0 0
1 1 0 0 0 1 0
1 1 0 1 0 0 1
1 1 1 0 1 0 0
1 1 1 1 1 1 1

Декодер принимает семи–битовое слово $v=(i_1^{'},i_2^{'},i_3^{'},i_4^{'},p_1^{'},p_2^{'},p_3^{'})$. При передаче в этом слове произошло не более одной ошибки. Изображенный на рис. 1.12 декодер вычисляет биты

$$\begin{aligned} S_1 &= p_1^{'} + i_1^{'} + i_2^{'} + i_3^{'}; \\ S_2 &= p_2^{'} + i_2^{'} + i_3^{'} + i_4^{'}; \\ S_3 &= p_3^{'} + i_1^{'} + i_2^{'} + i_4^{'}. \end{aligned} \qquad (1.65)$$

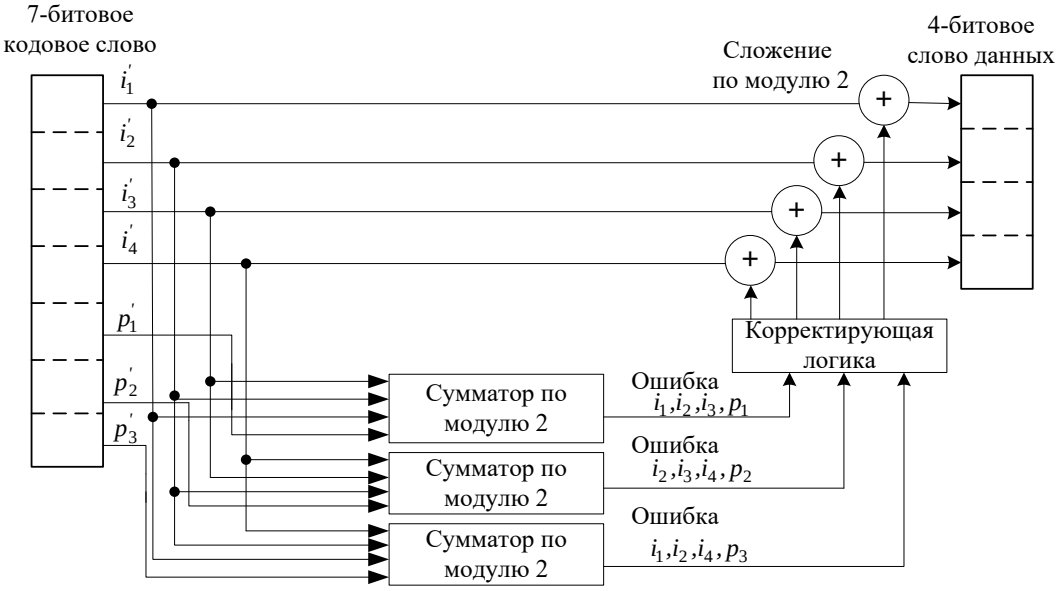


Рис. 1.12. Декодер для простого (7, 4) – кода Хемминга

Трехбитовая последовательность *(S_1, S_2, S_3)* называется *синдромом*. Она зависит не от истинных информационных битов, а только от конфигурации ошибок. Всего имеется восемь возможных синдромов: один для случая отсутствия ошибки и по одному для каждой из семи возможных одиночных ошибок. Простая проверка показывает, что каждая из этих ошибок имеет свой единственный синдром. Таким образом, не составляет труда сконструировать цифровую логику, которая по синдрому локализует соответствующий бит. После внесения исправления проверочные символы можно опустить. Две или более ошибки превышают возможности кодовой конструкции, и код будет ошибаться. Это означает, что он будет вносить неправильные исправления и выдавать искаженные информационные биты.

Простой итеративный код

Простая система кодирования, которая, как, оказывается, имеет различные применения, может быть построена следующим образом. Предположим, что нужно передать девять информационных символов. Эти девять символов можно расположить в виде квадратной матрицы, как показано в табл. 1.2, с проверочными символами, добавленными к каждой строке и к каждому столбцу. Один оставшийся символ P_7 является общим проверочным символом на четность.

Таблица 1.2

Простой итеративный блоковый код с проверкой на четность по строкам и по столбцам

A_1	A_2	A_3	P_1
A_4	A_5	A_6	P_2
A_7	A_8	A_9	P_3
P_4	P_5	P_6	P_7

Легко показать, что кодовое расстояние этого кода равно 4. Прежде всего одиночная ошибка приведет к тому, что проверки на четность, соответствующие строке и столбцу, содержащие ошибку, не будут выполняться. Таким образом, координаты одиночной ошибки однозначно определяются по номерам строки и столбца, в которых не выполняется проверка на четность. Поскольку эти номера различны для различных одиночных ошибок, кодовое расстояние будет не меньше 3. Поскольку наличие проверки P_7 гарантирует, что все кодовые слова имеют четный вес, кодовое расстояние будет не меньше 4. Вместе с тем кодовое расстояние в точности равно 4, поскольку код с кодовым расстоянием 5 позволяет однозначно распознавать все двукратные ошибки. В рассматриваемом коде этого сделать нельзя, поскольку ошибки в символах A_5 и A_9 приводят в точности к тем же невыполненным проверкам (синдрому), что и ошибки в символах A_6 и A_8. Проверочная матрица этого кода имеет вид

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Следует заметить, что матрица H может быть записана в нескольких эквивалентных формах. Одна из этих форм, имеющая определенные преимущества при реализации в некоторых ситуациях, может быть получена добавлением четвертой, пятой и шестой строк к последней строке, в результате чего последняя строка становится равной

(0 0 0 0 0 0 0 0 0 1 1 10 0 0 1),

и P_7 вычисляется теперь как сумма P_1, P_2 и P_3. Аналогично P_7 можно вычислять как сумму P_4, P_5 и P_6. Удобство, достигаемое при вычислении P_7 как суммы P_1, P_2 и P_3, состоит в том, что при таком способе все столбцы вычисляются абсолютно одинаково. Этот принцип справедлив при любых размерах массива. Таким образом, описанный метод оказывается полезным в случае, когда данные естественно формируются в виде массива, как, например, знаки в шине для передачи данных или на бумажной ленте, когда длина серии знаков не фиксирована. Во многих практических случаях, связанных с пересылкой данных, обычно вводится проверка на четность для каждого знака. Нетрудно при этом вычислить знак общей проверки, записав серию знаков в регистре, использующем сложение по модулю 2.

Полиномиальные коды

В предыдущих примерах кодовое слово *(n, k)*–кода представлялось в виде набора длиной *n:*

$$(a_0, a_1, \ldots, a_{n-1}).$$

Другой способ представления того же кодового слова состоит в том, чтобы считать элементы $a_0, a_1, \ldots, a_{n-1}$ коэффициентами многочлена от x. Таким образом,

$$f(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}. \qquad (1.66)$$

Используя это обозначение, можно определить полиномиальный код как множество всех многочленов степени, не большей $n-1$, содержащих в качестве множителя некоторый фиксированный многочлен $g(x)$. Многочлен $g(x)$ называется *порождающим многочленом* кода. Для того чтобы иметь возможность умножать такие кодовые многочлены, разлагать их на множители и производить над ними другие операции, нужно уметь складывать, вычитать, умножать и делить их коэффициенты. Этого легко добиться, если потребовать, чтобы все коэффициенты были элементами некоторого конечного поля.

1.5. Арифметика полей Галуа, основные теоремы и свойства

При решении задачи синтеза и анализа дискретных сигналов и кодов широко используется аппарат алгебры, и, в частности, теории конечных полей. Решение этих задач основывается на выполнении алгебраических операций над совокупностью произвольных элементов множеств, образующих конечные поля.

Многие коды, исправляющие ошибки, основаны на структурах колец многочленов и полей Галуа. Кроме того, эти алгебраические понятия и методы являются необходимым рабочим инструментом для конструирования кодеров и декодеров.

Изучаемые в современной алгебре арифметические системы классифицируются в соответствии с усложнением их математической структуры.

Дадим следующие формальные понятия:

- *абелева группа* – множество математических объектов, которые можно складывать и вычитать;

- *кольцо* – множество математических объектов, которые можно складывать, вычитать, умножать;
- *поле* – множество математических объектов, которые можно складывать, вычитать, умножать и делить.

В поле, состоящем из 2 элементов, выполним простейшие операции:

0 + 0 = 0; 0 + 1 = 1; 1 + 0 = 1; 1 + 1= 0; 0 * 0 = 0; 0 * 1 = 0; 1 * 0 = 0; 1 * 1 = 1.

Операции сложения и умножения в поле называются *сложением и умножением по модулю 2.*

При этом из 1 + 1 = 0, следует, что – 1 = 1, а из равенства $1 \cdot 1$ – что $1^{-1} = 1$.

Поле из двух элементов имеет алфавит 0 и 1.

Введем понятия *подгруппы и смежных классов.*

Подгруппа – некоторое подмножество группы.

Пусть G – группа, и пусть H – некоторое подмножество в G. H называется *подгруппой* группы G, если оно является группой относительно ограничения операции "*" на H.

Для доказательства, что непустое множество H является подгруппой группы G, необходимо проверить, что для всех a и b из H элемент $a * b$ принадлежит H (замкнутость) и что элемент обратный, a из H, также принадлежит H. Остальные групповые свойства наследуются из группы G.

Один из путей построения подгруппы H конечной группы G состоит в выборе произвольного элемента h и формировании H как множества элементов, образованных умножением h на самое себя произвольное число раз. Таким образом, строим последовательность h, h^2, h^4, ... Так как G конечна, то только конечное число элементов различно. Так, с некоторого момента последовательность начнет повторяться. Первым повторяющимся элементом будет сам элемент h. Множество H называется подгруппой, порожденной элементом h.

Порядком элемента h называется число элементов c в H. Множество элементов h, $h^{2,}$, h^4 ,..., $h^c = 1$ называется *циклом;* группа, состоящая из всех степеней одного из ее элементов называется *циклической.*

Для заданных G и H существует важная операция, которая устанавливает некоторые взаимосвязи между ними и называется *разложением группы G на смежные классы по H.* Обозначим через $h_1, h_2, h_3, \ldots, h_n$ элементы из H.

Построим таблицу следующим образом: первая строка состоит из элементов подгруппы H, причем первым слева выписан единичный элемент h_1 и каждый элемент из H записан в строке один и только один раз. Выберем произвольный элемент группы G, не содержащийся в первой строке. Назовем его g_2 и используем его в качестве первого элемента второй строки.

Остальные элементы второй строки получаются умножением слева элементов подгруппы на этот первый элемент. Аналогично строим третью, четвертую и пятую строки. Построение заканчивается тогда, когда после некоторого шага оказывается, что каждый элемент группы записан в некотором месте таблицы. Процесс обрывается в силу конечности G.

Таблица 1.3

Разложения на смежные классы

$h_1=1$	h_2	h_3	...	h_n
$g_2*h_1=g_2$	g_2*h_2	g_2*h_3	...	g_2*h_n
$g_3*h_1=g_3$	g_3*h_2	g_3*h_3	...	g_3*h_n
...	...	...	...	...
$g_m*h_1=g_m$	g_m*h_2	g_m*h_3	...	g_m*h_n

Первый элемент слева в каждой строке называется *лидером смежного класса.*

Каждая строка таблицы называется *левым смежным классом.*

Теорема 1.14. В разложении группы G на смежные классы каждый элемент из G встречается один и только один раз.

Доказательство. Каждый элемент появится хотя бы один раз, так как в противном случае процесс не остановится. Предположим, что два элемента одной и той же строки, $g_i * h_j$ и $g_i * h_k$ равны. Тогда умножение каждого из них на g_i^{-1} дает равенство $h_j = h_k$. Это противоречит тому, что каждый элемент подгруппы выписан в первой строке только один раз.

Предположим, что два элемента одной и той же строки, $g_i * h_j$ и $g_i * h_k$ равны и что $k < i$. Умножение справа на h_i^{-1} дает равенство $g_i = g_k * h_i * h_j^{-1}$. Тогда g_i порождает *к*–й смежный класс, так как элемент $h_i * h_j^{-1}$ принадлежит подгруппе. Это противоречит указанному выше правилу выбора лидеров смежных классов.

Следствие 3. Если *H* – подгруппа группы *G*, то число элементов в *H* делит число элементов в группе *G*. Таким образом

(Порядок *H*)* (Число смежных классов в *G* по *H*) = (Порядок *G*). (1.67)

Доказательство. Следует непосредственно из прямоугольности табл. 1.3 разложения на смежные классы.

Теорема 1.15. Порядок конечной группы делится на порядок любого из ее элементов.

Доказательство. Группа содержит циклическую подгруппу, порожденную любым из ее элементов; таким образом, утверждение теоремы вытекает из следствия 3.

Рассмотрим понятия и основные свойства произвольных множеств, образующих группы, кольца или поля.

Группа – множество элементов с определенной для каждой пары элементов операцией (обозначаемой “*”), обладающее следующими свойствами:

1) *замкнутость*: *для каждой пары a и b из множества элемент c принадлежит множеству*

$$c = a * b; \qquad (1.68)$$

2) *ассоциативность*: для всех *a*, *b*, *c* из множества

$$a * (b * c) = (a * b) * c; \tag{1.69}$$

3) *существование единицы:* во множестве существует элемент e, называемый *единичным элементом* и такой, что для любого элемента a из множества

$$a * e = e * a = a; \tag{1.70}$$

4) *существование обратных элементов*: для любого a из множества существует некоторый элемент b, называемый обратным элементу a и такой, что

$$a * b = b * a = e. \tag{1.71}$$

Некоторые группы обладают дополнительным свойством – *коммутативностью*:

$$a * b = b * a\,. \tag{1.72}$$

Такие группы называются *коммутативными* или *абелевыми группами.*

В этом случае операция “*” записывается “+”, единичным элементом является 0, а обратный элементу a элемент записывается в виде $-a$, так что

$$a + (-a) = (-a) + a = 0\,. \tag{1.73}$$

Групповая операция, обозначаемая символом “×”, называется умножением. В этом случае единичный элемент называется единицей и обозначается “1”, а обратный элементу элемент a записывается в виде a^{-1}, так что

$$a \times a^{-1} = a^{-1} \times a = 1. \tag{1.74}$$

Теорема 1.16. Единичный элемент в каждой группе является единственным. Для каждого элемента группы обратный элемент также является единственным и

$$(a^{-1})^{-1} = a\,. \tag{1.75}$$

Доказательство. Предположим, что e и e' – единичные элементы группы, тогда $e = e * e' = e'$.

Далее предположим, что b и b' – элементы, обратные элементу a, тогда

$$b = b * (a * b') = (b * a) * b' = b' \quad . \tag{1.76}$$

В соответствии с (1.74) $a \times a^{-1} = a^{-1} \times a = 1$, так что a – обратный элементу a^{-1}.

Кольцом R называется множество с двумя определенными на нем операциями: *сложением* и *умножением*, причем имеют место следующие аксиомы:

- относительно "+" R является абелевой группой;
- *замкнутость*: для любых a и b произведение ab принадлежит R;
- *закон ассоциативности*:

$$a\,(b\,c) = (a\,b)c; \tag{1.77}$$

- *закон дистрибутивности*:

$$a(b + c) = ab + bc, \ (b + c\,)a = ba + ca. \tag{1.78}$$

Теорема 1.17. Для произвольных a и b в кольце R:

$$a0 = 0a = 0;$$

$$a\,(–b) = (–a)\,b = –(ab). \tag{1.79}$$

Доказательство. $a0 = a(0+0) = a0 + a0.$
Вычитая из обеих частей равенство $a0,$ получаем $0 = a0.$
Вторая часть утверждения доказывается аналогично:

$$0 = a0 = a(b–b\,)= ab + a(–b).$$

Следовательно, $a(–b\,) = –(ab).$

Операция сложения в кольце имеет единичный элемент:

$$1a = a1 = a. \tag{1.80}$$

Теорема 1.18. В кольце с единицей:

единица единственна, если элемент a имеет как правый обратный элемент $ab = 1$, так и левый $ac = 1$, то элемент a называется *обратимым*, причем обратный ему элемент является единственным и обозначается через a^{-1}:

$$(a^{-1})^{-1} = a. \tag{1.81}$$

Доказательство. Рассуждения аналогичны проведенным при доказательстве теоремы 1.14.

Обратимый элемент кольца называется *единицей*. Множество всех единиц в кольце замкнуто относительно умножения, так как если a и b – единицы, то $c = ab$ имеет обратный элемент, равный $c^{-1} = b^{-1}a^{-1}$.

Теорема 1.19. Множество “1” кольца образует группу относительно умножения в кольце.

Если $c = ab$ и c – единица, то a имеет правый обратный, а b – левый обратный элемент.

Доказательство. Непосредственная проверка.

Полем называется множество с двумя определенными над ним операциями: *сложением* и *умножением,* причем имеют место аксиомы:

- множество образует абелеву группу по сложению;
- поле замкнуто относительно умножения, множество ненулевых элементов образует абелеву группу по умножению;
- закон *дистрибутивности*:

$$(a+b)c = ac + bc \tag{1.82}$$

для любых a, b, c из поля.

Единичный элемент для сложения принято обозначать через 0, аддитивный обратный элементу a элемент через $–a$.

Единичный элемент относительно умножения обозначается через 1 и называется единицей, мультипликативный обратный к элементу a элемент – через a^{-1}. Под вычитанием понимается $(a–b) = a = (–b)$, под делением (a/b) понимается $b^{-1}a$.

Широко известны следующие примеры полей:

R – множество вещественных чисел;
C – множество комплексных чисел;
Q – множество рациональных чисел.

Все эти поля имеют бесконечное множество элементов. Мы интересуемся полями, имеющими конечное число элементов. Поле с q элементами, если оно существует, называется *конечным полем* или *полем Галуа.*

Пример. Рассмотрим конечное поле $GF(4)$ с элементами поля $\{0, 1, 2, 3\}$, тогда результат операций сложения и умножения элементов поля можно представить в виде

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

*	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Поле обладает всеми свойствами кольца, а также важным свойством: в нем всегда возможно *сокращение*. Сокращение представляет слабую форму деления и означает, что, если

$$ab = ac, \quad \text{то } b = c. \tag{1.83}$$

Теорема 1.20. Если в произвольном поле $ab = ac$ и $a \neq 0$, то $b = c$.

Доказательство. Умножить на a^{-1}.

Для поля вещественных и комплексных чисел широко используется линейная алгебра, в частности теория матриц, однако большинство известных операций справедливо и для произвольного поля.

Кратко рассмотрим основные операции справедливые для полей Галуа:

матрицей A над кольцом R называется прямоугольная таблица, состоящая из n строк и m столбцов.

$(n \times m)$,

при этом множество a_{ii} называется *главной диагональю.*

Матрица $(n \times n)$ называется *квадратной матрицей.*

Пример. Рассмотрим простейшие операции: сложение матриц, умножение матрицы на число, перемножение матриц в конечном поле Галуа *GF(5)* с элементами поля {0, 1, 2, 3, 4}.

Пусть даны две матрицы:

$$A = \begin{vmatrix} 0 & 1 & 3 \\ 4 & 0 & 2 \\ 0 & 3 & 1 \end{vmatrix}, B = \begin{vmatrix} 1 & 1 & 2 \\ 3 & 0 & 0 \\ 0 & 4 & 1 \end{vmatrix}.$$

Рассмотрим *операцию сложения* матрицы A с матрицей B, при этом каждый элемент строки матрицы A складывается с элементом строки матрицы B, после чего полученное значение суммы записывается в строку результирующей матрицы C.Операции проводятся в конечном поле Галуа *GF(5)* поэтому сложение чисел следует производить по *mod5.*

$$A + B = \begin{vmatrix} 0 & 1 & 3 \\ 4 & 0 & 2 \\ 0 & 3 & 1 \end{vmatrix} + \begin{vmatrix} 1 & 1 & 2 \\ 3 & 0 & 0 \\ 0 & 4 & 1 \end{vmatrix} = \begin{vmatrix} 1 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 2 & 2 \end{vmatrix}.$$

Результирующая матрица $C = \begin{vmatrix} 1 & 2 & 0 \\ 2 & 0 & 2 \\ 0 & 2 & 2 \end{vmatrix}$.

Рассмотрим *операцию умножения матрицы на число.* Пусть дана матрица A

$$A = \begin{vmatrix} 2 & 4 & 3 \\ 4 & 3 & 2 \\ 2 & 3 & 0 \end{vmatrix}$$ и число $в$=3.

Операция умножения производится следующим образом: каждый элемент матрицы *A* умножается на число *в* и полученные произведения образуют результирующую матрицу *C*, при этом, как и в предыдущем случае, операцию умножения необходимо рассматривать с учетом *mod5*.

$$b \times |A| = 3 \times \begin{vmatrix} 2 & 4 & 3 \\ 4 & 3 & 2 \\ 2 & 3 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 2 & 4 \\ 2 & 4 & 1 \\ 1 & 4 & 0 \end{vmatrix}.$$

Результирующая матрица $C = \begin{vmatrix} 1 & 2 & 4 \\ 2 & 4 & 1 \\ 1 & 4 & 0 \end{vmatrix}$.

Рассмотрим *умножение матрицы на матрицу* в конечном поле Галуа.

Пусть даны матрицы *A* и *B*.

$$A = \begin{vmatrix} 1 & 0 & 3 \\ 0 & 4 & 2 \\ 1 & 3 & 0 \end{vmatrix}, \quad B = \begin{vmatrix} 2 & 1 & 0 \\ 3 & 4 & 0 \\ 0 & 1 & 2 \end{vmatrix}.$$

Операция умножения матрицы на матрицы производится по правилу “строка на столбец”, при этом элемент строки матрицы *A* умножается на соответствующий элемент столбца матрицы *B,* полученные суммы складываются между собой и число записывается в строку результирующей матрицы *C,* после чего эта же строка умножается на следующий столбец и т.д. Все операции умножения и сложения проводятся с учетом *mod5*.

$$A \times B = \begin{vmatrix} 1 & 0 & 3 \\ 0 & 4 & 2 \\ 1 & 3 & 0 \end{vmatrix} \times \begin{vmatrix} 2 & 1 & 0 \\ 3 & 4 & 0 \\ 0 & 1 & 2 \end{vmatrix} = \begin{vmatrix} 2 & 4 & 1 \\ 2 & 3 & 4 \\ 1 & 3 & 0 \end{vmatrix}.$$

Результирующая матрица $C = \begin{vmatrix} 2 & 4 & 1 \\ 2 & 3 & 4 \\ 1 & 3 & 0 \end{vmatrix}$.

Транспонированной матрицей $(n \times m)$ к матрице *A* называется матрица A^T $(m \times n)$.

Обратной матрицей A^{-1} называется такая матрица, что

$$AA^{-1} = 1.$$

Матрица, имеющая обратную матрицу, называется *невырожденной.*

Рассмотрим основные свойства определителя матрицы.

Число $detA = \Delta = \Delta_n$ называется *определителем* и вычисляется так:

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21} \quad . \tag{1.84}$$

Теорема 1.21. Если все элементы некоторой строки квадратной матрицы равны нулю, то:

- определитель этой матрицы равен 0;
- определитель матрицы равен определителю транспонированной матрицы;
- если две строки поменять местами, то определитель матрицы поменяет знак;
- если две строки матрицы равны, то определитель матрицы равен 0;
- если все элементы одной строки умножить на элемент поля, то определитель новой матрицы будет равен определителю исходной матрицы, умноженной на это число;
- если к элементам некоторой строки матрицы k раз прибавить соответствующие элементы некоторой другой ее строки, то определитель матрицы не изменится;
- определитель матрицы отличен от 0 тогда и только тогда, когда ее строки (столбцы) линейно независимы.

Доказательство. Непосредственная проверка.

Если в квадратной матрице удалить строку и столбец, содержащие a_{ij}, то определитель оставшейся квадратной матрицы размера n–1 называется *минором* элемента a_{ij} и обозначается через $M_{ij.}$.

Алгебраическое дополнение, обозначаемое через C_{ij}, определяется равенством

$$C_{ij} = (-1)^{i+j} M_{ij}.$$

Таким образом, определитель равен

$$det(A) = \sum_{k=1}^{n} a_{ik} C_{ik}. \qquad (1.85)$$

Строки $(n \times m)$–матрицы A над $GF(q)$ можно рассматривать как множество m–мерных векторов над $GF(q)$. *Пространство строк* матрицы A определяется как множество всех линейных комбинаций векторов–строк матрицы A. Размерность пространства строк называется *рангом матрицы по строкам.*

Рангом матрицы называется наивысший порядок отличных от нуля миноров этой матрицы.

Элементарными преобразованиями матрицы являются:

- перемена мест двух строк (столбцов);
- умножение строки (столбца) на произвольное, отличное от нуля число;
- прибавление одной строки (столбца) к другой строке (столбцу);
- транспонирование матрицы;
- вычеркивание строки (столбца) нулей.

Структура конечных полей и их свойства

Введем основные понятия и определения. Конечное поле, называемое также *полем Галуа* и обозначаемое через $GF(q)$, – это конечное множество, состоящее из q элементов, в котором определены правила для выполнения арифметических операций. Эти правила не очень отличаются от тех, которые используются при арифметических операциях с обычными числами.

Основное отличие состоит в том, что в конечном поле все операции производятся над конечным полем элементов. Введем некоторые изменения:

1. Существует две операции, используемые для комбинирования элементов: умножение и сложение.
2. Результатом умножения или сложения двух элементов является третий элемент, лежащий в этом поле.

3. Поле всегда содержит мультипликативную группу *{1, 2, 3, ..., q}* и мультипликативную единицу 1, аддитивную группу *{0, 1, ..., q–1}* и аддитивную единицу 0; таким образом, $a+0=a,\ a\times 1=a$ для любого элемента *a*.

4. Для любого элемента *a* существует обратный элемент по сложению *(–a)* и обратный элемент по умножению a^{-1} такие, что $a + (-a) = 0$ и $a \times a^{-1} = 1$; существование этих элементов определяет операции вычитания и деления.

Таким образом, для выполнения операции вычитания необходимо найти аддитивно обратный элемент и выполнить операцию сложения $a + (-a) = 0$ ($-a$ – аддитивно обратный элемент, $a + (-a) = 0$).

Для выполнения операции деления необходимо вычислить мультипликативно обратный элемент, $a\times a^{-1}=1$ (мультипликативно обратный элемент a^{-1}, $a\times a^{-1}=1$).

5. Выполняются обычные правила:

- ассоциативности

$$a+(b+c)=(a+b)+c, \quad a\times(b\times c)=(a\times b)\times c; \tag{1.86}$$

- коммутативности

$$a+b=b+a, \quad a\times b=b\times a; \tag{1.87}$$

- дистрибутивности

$$a\times(b+c)=a\times b+a\times c. \tag{1.88}$$

Конечные поля существуют не при любом числе элементов, а только в том случае, когда число элементов является *простым числом*. Для каждого допустимого значения q существует ровно одно поле. Если q – простое число, то элементами поля являются числа 0, 1, 2, …, $q–1$, а сложение и умножение являются обычными сложением и умножением по модулю q.

Теорема 1.22 (алгоритм деления). Для каждой пары целых чисел c и d при отличном от нуля d найдется единственная пара целых чисел Q (частное) и s (остаток) таких, что

$$c = dQ + s, \tag{1.89}$$

где $0 \le s \langle |d|$.

Доказательство. Непосредственная проверка.

Обычно больше интересует не частное, а остаток. Часто частное записывается в виде

$$s = R_d[c]. \tag{1.90}$$

Другим обозначением является

$$s \equiv c (mod\ d). \tag{1.91}$$

Теорема 1.23 (алгоритм Евклида). Наибольший общий делитель двух различных ненулевых целых чисел r и s может быть вычислен итеративным применением алгоритма деления. Предположим, что $r < s$ и оба эти числа положительны; тогда алгоритм состоит в следующем:

$$\begin{aligned} s &= Q_1 r + r_1; \\ r &= Q_2 r + r_2; \\ r &= Q_3 r + r_3; \\ &\vdots \\ r_{n-1} &= Q_{n+1} r_n. \end{aligned} \tag{1.92}$$

и процесс заканчивается, когда полученный остаток равен нулю. Последний ненулевой остаток r_n равен наибольшему общему делителю.

Следствие 4. Для любых целых чисел r и s существуют целые числа a и b такие, что наименьший общий делитель (НОД)

$$\text{НОД}\ (r,s) = ar + bs. \tag{1.93}$$

Доказательство. Последний остаток в теореме равен наименьшему общему делителю (r,s). Воспользуемся множеством выписанных в этой теореме уравнений, чтобы

исключить все остальные остатки. Это даст выражение для r_n в виде линейной комбинации r и s с целочисленными коэффициентами.

Пример. Рассмотрим конечное простое поле *GF(5)* при $q = 5$ с элементами поля {0, 1, 2, 3, 4}.

Определим операции сложения и умножения, вычитания и деления, результаты операций представим в виде табл. 1.4 – 1.7.

Таблица 1.4

Операция сложения по *mod q*

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Таблица 1.5

Операция умножения по *mod q*

×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Для выполнения операции *вычитания* вычислим аддитивно обратные элементы поля *GF(5)* по выражению

$$a + (-a) = 0. \qquad (1.94)$$

Таблица 1.6

Операция вычитания по *mod q*

+	–0	–1	–2	–3	–4
0	0+0=0	0+4=4	0+3=3	0+2=2	0+1=1
1	1+0=1	1+4=0	1+3=4	1+2=3	1+1=2
2	2+0=2	2+4=1	2+3=0	2+2=4	2+1=3
3	3+0=3	3+4=2	3+3=1	3+2=0	3+1=4
4	4+0=4	4+4=3	4+3=2	4+2=1	4+1=0

Для выполнения операции *деления* вычислим мультипликативно обратные элементы поля *GF(5)* по выражению

$$a \times a^{-1} = 1. \qquad (1.95)$$

Таблица 1.7

Операция деления по *modq*

$\times$	1^{-1}	2^{-1}	3^{-1}	4^{-1}
1	$1\times1=1$	$1\times3=3$	$1\times2=2$	$1\times4=4$
2	$2\times1=2$	$2\times3=1$	$2\times2=4$	$2\times4=3$
3	$3\times1=3$	$3\times3=4$	$3\times2=1$	$3\times4=2$
4	$4\times1=4$	$4\times3=2$	$4\times2=3$	$4\times4=1$

Рассмотрим конечные поля, основанные на кольцах многочленов, введем основные понятия и определения**.**

Многочленом над полем *GF(q)* называется математическое выражение

$$f(x) = f_{n-1}x^{n-1} + f_{n-2}x^{n-2} + \ldots + f_1x + f_0, \qquad (1.96)$$

где символ x называется *неопределенной переменной,* коэффициенты $f_{n-1}\ldots f_0$ принадлежат полю *GF(q)* и являются целыми числами.

Нулевым многочленом называется многочлен, для которого $f(x)= 0$.

Приведенным многочленом называется многочлен, старший коэффициент которого f_{n-1} равен 1.

Степенью ненулевого многочлена f(x) называется индекс коэффициента f_{n-1} и обозначается *deg f (x).*

Неприводимым многочленом называется многочлен *p(x),* делящийся только на многочлены $\alpha p(x)$ или α, где α – произвольный ненулевой элемент поля *GF(q)* .

Простым многочленом называется приведенный неприводимый многочлен.

Кольцо многочленов по модулю приведенного многочлена *p(x)* является полем тогда и только тогда, когда многочлен *p(x)* прост.

Для построения полей Галуа используют *примитивные элементы поля GF(q)*: *элемент поля α* – такой, что все остальные элементы поля, за исключением нуля, могут быть представлены в виде степени *α*.

Примитивным многочленом p(x) над полем *GF(q)* называется простой многочлен, такой, что в расширении поля, построенном по модулю *p(x),* соответствующий многочлену *x* элемент поля является примитивным.

Конечное поле, построенное над кольцом многочленов, называется *расширенным полем Галуа* и обозначается *GF(q)*, где $q=p^m$: *p* (простое целое число) называется *характеристикой расширенного поля*; *m* (ненулевое положительное целое число по *mod q(x)*) называется *степенью расширения примитивного поля.*

В общем виде *порождающий многочлен* равен

$$f(x) = a_0 + a_1x + a_2x^2 + \ldots + a_{n-1}x^{n-1}. \qquad (1.97)$$

Элементами поля являются все многочлены, полученные путем подстановки характеристики поля в *порождающий многочлен.*

Правила умножения и сложения таких многочленов получаются из обычного умножения и сложения многочленов и последующего приведения результата по модулю некоторого специального многочлена *q(x)* к степени *m.* Этот многочлен обладает тем свойством, что его нельзя разложить на множители, используя только многочлены с коэффициентами из *GF(p)*. Такие многочлены называются *неприводимыми,* они аналогичны простым числам.

Как и простые числа, они обычно находятся методом перебора; таблицы неприводимых многочленов имеются в нескольких книгах.

Рассмотрим множество всех многочленов от формальной переменной *x* в степени *m–1* с коэффициентами из *GF (p),* где *p* – простое число.

Пример. Рассмотрим расширенное поле *GF(4),* построенное через порождающий многочлен $q(x)= x^2+x+1$. Многочлен *q(x)* неприводим над *GF(2)* и может быть использован для построения $GF(2^2)$. Элементами поля являются *{0, 1, x, x+1}.*

Поле всегда содержит мультипликативную группу *{1, 2, ..., q}* с мультипликативной единицей 1 такую, что $a \times a^{-1} = 1$ и аддитивную группу *{0, 1, ..., q–1}* такую, что $a + (-a) = 0$.

Построим таблицы умножения, сложения, вычитания и деления для расширенного поля $GF(p^m)$ при $p=2$ (поле задается только 0 и 1), $m=2$ и примитивного многочлена $q(x) = x^2 + x + 1$.

Элементами поля будут все возможные варианты многочленов, полученных путем подстановки характеристики поля *p* в *порождающий многочлен {0, 1, x, x+1}.*

Для определения элементов поля используют *регистры сдвига с обратными связями.* Регистр сдвига с обратными связями состоит из *m* ячеек (см. табл. 1.8).

Таблица 1.8

Регистр сдвига с обратными связями

x^2	x^1	x^0	Выполняемые операции
0	0	0	На начальном этапе регистр обнулен
0	0	1	Введем регистр 1 в правую ячейку "x^0"
0	1	0	На следующем этапе содержимое сдвигается на одну ячейку вправо
1 ⊕ 1	0 1	0 1	Если в ячейке "x^2" появляется 1, то ко всему регистру добавляется порождающий (примитивный) многочлен x^2+x+1 (111)
0	1	1	На следующем этапе содержимое сдвигается на одну ячейку вправо
1 ⊕ 1	1 1	0 1	Если в ячейке "x^2" появляется 1, то ко всему регистру добавляется порождающий (примитивный) многочлен x^2+x+1 (111)
0	0	1	Такой набор уже есть, произошел цикл

Таким образом, элементами поля являются {0, 1, *x*, *x*+1}.

Результаты операций сложения, умножения, вычитания и деления для расширенного поля *GF(4)* приведены в табл. 1.9– 1.12.

При этом необходимо учитывать, что все операции вычисляются в конечном расширенном поле по *modq(x).*

Таблица 1.9

Операция сложения по *modq(x)*

+	0	1	x	x+1
0	0	1	x	x+1
1	1	0	x+1	x
x	x	x+1	0	1
x+1	x+1	x	1	0

Таблица 1.10

Операция сложения по *modq(x)*

×	0	1	x	x+1
0	0	0	0	0
1	0	1	x	x+1
x	0	x	x+1	1
x+1	0	x+1	1	x

Для выполнения операции *вычитания* вычислим аддитивно обратные элементы поля *GF(4)*. Во всех полях $GF(2^m)$ аддитивно обратные элементы равны самим элементам.

Таблица 1.11

Операция вычитания по *modq(x)*

+	–0	–1	–x	–x+1
0	0+0=0	0+1=1	0+x=x	0+x+1=x+1
1	1+0=1	1+1=0	1+x=1+x	1+x+1=x
x	x+0=x	x+1=1	x+x=0	x+x+1=1
x+1	x+1+0=x+1	x+1+1=x	x+1+x=1	x+1+x+1=0

Для выполнения операции *деления* вычислим мультипликативно обратные элементы поля *GF(4)*.

Таблица 1.12

Операция деления по *modq(x)*

×	1^{-1}	x^{-1}	$(x+1)^{-1}$
1	1×1=1	1×(x+1)=x+1	1×x=x
x	x×1=x	x×(x+1)=1	x×x=x+1
x+1	(x+1)×1=x+1	(x+1)×(x+1)=x	(x+1)×x=1

Рассмотрим основные свойствами *полей Галуа*:

- число элементов любого поля Галуа равно степени простого числа;
- для любого простого p и целого положительного m наименьшим подполем поля $GF(p^m)$ является поле $GF(p)$. Элементы поля $GF(p)$ называются *целыми числами* поля $GF(p^m)$, а число p – его *характеристикой.*
- в $GF(2)$ для каждого β поля справедливо равенство $-\beta = \beta$;
- для любого простого p и целого положительного m существует $GF(p^m)$ с p^m элементами;
- каждое поле Галуа содержит хотя бы один примитивный элемент;
- над каждым полем Галуа существует хотя бы один примитивный многочлен любой положительной степени;
- каждый примитивный элемент имеет над любым подполем простой минимальный многочлен;
- два поля Галуа с одним и тем же числом элементов изоморфны.

Между многочленами и элементами конечных полей имеется дополнительная связь – связь между сомножителями и корнями многочленов с вещественными коэффициентами.

Например, заданный многочлен может быть представлен в виде

$$f(x)= (x–\beta_1)(x–\beta_2)\dots(x–\beta_j),$$

где j значений $\beta_1,\dots,\beta_j$ являются корнями многочлена $f(x)$.

Многочлены и элементы поля Галуа обладают рядом свойств, которые оказываются полезными при описании кодов и построении декодеров.

*Свойство 1***.** Неприводимые многочлены. Многочлен $f(x)$ с элементами из некоторого конечного поля называется *неприводимым,* если его нельзя разложить на множители, используя лишь элементы поля. Однако этот многочлен всегда

можно разложить на множители, используя элементы из некоторого расширения. Поэтому многочлен всегда имеет корни в некотором расширении.

Если $f(x)$ – неприводимый многочлен с коэффициентами из $GF(p)$ и α – его корень, то α^p, α^{p2}, α^{p3} ,… также будут корнями. Кроме того, все корни $f(x)$ могут быть найдены таким способом. Многочлен $f(x)$ называется *минимальной функцией от α.* Если α *– примитивный элемент* (генератор – элемент, обладающий тем свойством, что любой другой элемент поля является некоторой степенью этого элемента), то $f(x)$ называется *примитивным многочленом.*

Пример. Многочлен $f(x)=1+x+x^3$ неприводим над полем $GF(2)$. Элемент α является корнем, поскольку $f(\alpha) =1+\alpha+\alpha^3 =$ $= (100)+(010)+(110)=(000)=0$.

Используя свойство, получаем, что α^2, α^4 тоже являются корнями.

$$f(\alpha^2) =1+\alpha^2+\alpha^6 =(100)+(001)+(101)=(000)=0;$$

$$f(\alpha^4) =1+\alpha^4+\alpha^{12} = 1+\alpha^4+\alpha^5 =(100)+(011)+(111)=(000)=0.$$

И наоборот, если α, α^2, α^4 являются корнями $f(x)$, то $f(x)$ можно записать в виде

$$f(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^4) =x^3 -x^2(\alpha+ \alpha^2+ \alpha^4)+x(\alpha^3+ \alpha^6+ \alpha^5)-\alpha^7= =x^3+x+1.$$

Свойство 2. $f(x^p)=f^p(x)$. Если $f(x)$ – произвольный многочлен, коэффициенты которого лежат в $GF(p)$, то $f(x^p)=f^p(x)$.

Справедливость этого свойства вытекает из того, что все попарные или многократные произведения в $f^p(x)$ появляются с коэффициентом, который делится на p и, значит, равен 0 в $GF(p)$.

Пример. $(1+x+x^2)^2=1+x^2+x^4+2(x+x^2+x^3)=1+x^2+x^4$.

Свойство 3. Сомножители $x_m–1$. *Порядком m* элемента β конечного поля называется наименьшее значение *m,* для которого $\beta^m=1$. По определению β является корнем многочлена $x^m–1$. Если β является корнем некоторого неприводимого многочлена *f(x),* то *f(x)* должен быть делителем $x^m–1$.

Свойство 4. Поле $GF(p^m)$ и корни $x^{p^m-1}-1$. Корни многочлена $x^{p^m-1}-1$ совпадают с ненулевыми элементами $GF(p^m)$.

Пример свойства 3 и 4. Многочлен $f(x)=x^7–1$ разлагается на множители следующим способом: $x^7–1=(x^3+x^2+1)(x^3+x+1)(x+1)$.

Используя табл. 2.3, легко показать, что корнями x^3+x^2+1 являются $\alpha, \alpha^2, \alpha^4$, а корнями x^3+x+1 являются $\alpha^3, \alpha^5, \alpha^6, ax+1$. Эти семь корней являются семью ненулевыми элементами *GF(8).*

Свойство 5. Делимость $x^m–1$ на *f(x)*. Наименьшее значение *m,* для которого произвольный многочлен *f(x)* без кратных корней делит $x^m–1$, совпадает с *наименьшим общим кратным* порядков корней *f(x)*. Поэтому *m* является длиной самого короткого цикла, порожденного регистром с обратными связями, определяемыми многочленом *f(x)* .

Пример. Порядок корней $1+x+x^3$ равен 7, а порядок корней $1+x^3+x^4$ равен 15. НОК, кратное 7 и 15, равно 105.

Поэтому $g(x)=(1+x+x^3)(1+x^3+x^4)=1+x+x^5+x^6+x^7$ делит $x^{105}–1$ и порождает цикл длиной 105.

Свойство 6. Делимость $x^n–1$ на $x^m–1$. Многочлен $x^n–1$ делится на $x^m–1$ в том и только том случае, если *n* делится на *m*. Это вытекает из того, что если корни $x^m–1$ являются также корнями $x^n–1$, то *n* должно делиться на *m*.

Пример. Используя простые вычисления, получим, что на $x^3–1$ делится $x^6–1$, но не делится $x^5–1$ и $x^7–1$.

Описанные выше свойства конечных полей нашли широкое применение в циклических кодах.

Циклический код длины *n* над полем *GF(q)* существует для каждого многочлена *g(x)* над *GF(q),* делящего многочлен $x^n–1$.

Тогда *порождающий многочлен* состоит в разложении многочлена $x^n–1$ на простые множители.

$$x^n–1 = f_1(x)\, f^2(x)\, f^3(x) \ldots fs(x) = g(x),$$

где *s*– число простых множителей.

Предположим, многочлен $g(x)$ – порождающий многочлен. Он делит многочлен $x^n–1$ и, следовательно,

$$g(x) = \prod f_i(x). \qquad (1.98)$$

Порождаемый многочленом циклический код состоит из многочленов, которые делятся на каждый из $f_i(x)$.

Многочлен $g(x)$ можно найти, найдя все его простые делители.

Для кода над $GF(q)$ длина $n=q^m–1$ называется *примитивной.* Циклический код примитивной длины над $GF(q)$ называется *примитивным циклическим кодом.*

Поле $GF(q^m)$ называется расширением поля $GF(q)$. Разложение

$$x^{q^m 1} - 1 = \prod_j (x - \beta_j),$$

где β_j – все ненулевые элементы $GF(q^m)$. Тогда $f_j(x)$ – минимальные многочлены элемента β_j.

Два элемента из поля $GF(q^m)$, являющиеся корнями одного и того же минимального многочлена над $GF(q)$, называются *сопряженными* (относительно $GF(q)$).

Классом сопряженных элементов называется множество $\left\{\beta, \beta^q, \beta^{q^2}, ..., \beta^{q^{r-1}}\right\}$, где r – наименьшее целое число такое, что

$$\beta^{q^r} = \beta. \qquad (1.99)$$

Минимальный многочлен элемента β равен

$$f(x)= (x–\beta)(x–\beta^q) \ldots \left(x - \beta^{q^{r-1}}\right). \qquad (1.100)$$

q–ым следом элемента β поля $GF(q^m)$ называется сумма

$$tr(\beta)=\sum_{i=0}^{m-1}\beta^{q^i}\ . \qquad (1.101)$$

Если класс сопряженных элементов, которому принадлежит β, содержит m элементов, то $tr(\beta)$ равен сумме всех элементов этого класса сопряженных элементов

$$tr(\beta+\gamma)=tr(\beta)+tr(\gamma)\,. \qquad (1.102)$$

Все сопряженные элементы имеют один и тот же след.

Над $GF(q^m)$ q–ый *след* принимает в качестве своего значения каждое из чисел поля $GF(q)$ одинаково часто, а именно q^{m-1} раз.

Квадратное уравнение $x^2+x+a=0$, где a – элемент поля $GF(2^m)$, имеющий корни в поле $GF(2^m)$ тогда и только тогда, когда двоичный след элемента a равен 0, т.е.

$$tr(a)=0.$$

Пример. Рассмотрим конечное поле, построенное по кольцу многочленов с коэффициентами из $GF(2)$, примитивный многочлен $g(x)=x^4+x+1$, тогда элементы поля $\{0, 1, x, x^2, x^3, x+1, x^2+x, x^3+x^2, x^2+1, x^3+x, x^2+x+1, x^3+x^2+x, x^3+x^2+x+1, x^3+x^2+1, x^3+1\}$. Найдем классы сопряженных элементов и определим минимальные многочлены для каждого класса.

1. Определим *классы сопряженных элементов* через выражение

$$\beta^{q^r}=\beta\,,$$

где $q=2$, $r = 0, 1, 2, \ldots, n-1$.

$n = q^m-1$; $n=15$:

1 класс – по α^0 – $f_{\alpha^0}=\{\alpha^0\}$;

2 класс – по α^1 $=(\alpha^1)^{2^0}=\alpha^1, (\alpha^1)^{2^1}=\alpha^2, (\alpha^1)^{2^2}=\alpha^4, (\alpha^1)^{2^3}=\alpha^8, (\alpha^1)^{2^4}=\alpha^1$,
$f_{\alpha^1}=\{\alpha^1,\alpha^2,\alpha^4,\alpha^8\}$;

3 класс – по α^3 $=(\alpha^3)^{2^0}=\alpha^3,(\alpha^3)^{2^1}=\alpha^6,(\alpha^3)^{2^2}=\alpha^{12},(\alpha^3)^{2^3}=\alpha^9,$
$(\alpha^3)^{2^4}=\alpha^3$;
$f_{\alpha^3}=\{\alpha^3,\alpha^6,\alpha^9,\alpha^{12}\}$;

4 класс – по $\alpha^5 = (\alpha^5)^{2^0} = \alpha^5, (\alpha^5)^{2^1} = \alpha^{10}, (\alpha^5)^{2^2} = \alpha^5$;

$f_{\alpha^5} = \{\alpha^5, \alpha^{10}\}$;

5 класс – по $\alpha^7 = (\alpha^7)^{2^0} = \alpha^7, (\alpha^7)^{2^1} = \alpha^{14}, (\alpha^7)^{2^2} = \alpha^{13}, (\alpha^7)^{2^3} = \alpha^{11}$, ;

$(\alpha^7)^{2^4} = \alpha^7$;

$f_{\alpha^7} = \{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$.

2. Определим *след*

$f_{\alpha^0} = \{\alpha^0\} = 0$;

$f_{\alpha^1} = \{\alpha^1, \alpha^2, \alpha^4, \alpha^8\} = 0$;

$f_{\alpha^3} = \{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\} = 1$;

$f_{\alpha^5} = \{\alpha^5, \alpha^{10}\} = 1$;

$f_{\alpha^7} = \{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\} = 1$.

3. Определим порядок по выражению

$$(\alpha^i)^p = 1, \qquad (1.103)$$

где p – порядок, при этом

$(\alpha^0)^p = 1$, $p=15$;
$(\alpha^1)^p = (\alpha^2)^p = (\alpha^4)^p = (\alpha^8)^p = 1$, $p=15$;
$(\alpha^3)^p = (\alpha^6)^p = (\alpha^9)^p = (\alpha^{12})^p = 1$, $p=5$;
$(\alpha^5)^p = (\alpha^{10})^p = 1$, $p=3$;
$(\alpha^7)^p = (\alpha^{11})^p = (\alpha^{13})^p = (\alpha^{14})^p = 1$, $p=15$.

Таким образом, *каждый класс сопряженных элементов* имеет один и тот же порядок.

4. Определим *минимальные многочлены* для каждого класса:

$f(\alpha^0) = (x+\alpha^0) = x+1$;
$f(\alpha^1) = (x+\alpha^1)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8) = x^4 + x + 1$;
$f(\alpha^3) = (x+\alpha^3)(x+\alpha^6)(x+\alpha^9)(x+\alpha^{12}) = x^4 + x^3 + x^2 + x + 1$;
$f(\alpha^5) = (x+\alpha^5)(x+\alpha^{10}) = x^2 + x + 1$;
$f(\alpha^7) = (x+\alpha^7)(x+\alpha^{11})(x+\alpha^{13})(x+\alpha^{14}) = x^4 + x^3 + 1$.

Спектральные преобразования в конечных полях

Исследование сигналов с непрерывным временем, принимающих вещественные и комплексные значения, связано с преобразованием Фурье; в случае сигналов с дискретным временем аналогичную роль играет дискретное преобразование Фурье. Для многих значений *n* существуют также преобразования Фурье на векторном пространстве последовательностей длины *n* над полем Галуа *GF(q)*. Преобразования Фурье в поле Галуа могут играть важную роль в исследовании и обработке *GF(q)*–ых сигналов, т.е. кодовых слов. Основываясь на преобразовании Фурье, можно определить циклические коды как коды, в которых некоторые спектральные компоненты слов равны нулю. Декодирование кодов БЧХ и кодов Рида–Соломона также может быть описано на спектральном языке.

В поле комплексных чисел дискретное преобразование Фурье вектора $p = \{p_i, i = 0,\ldots,N-1\}$с комплексными компонентами определяется как вектор, $P = \{P_k, k = 0,\ldots,N-1\}$ задаваемый равенствами

$$P_k = \sum_{i=0}^{N-1} e^{-2\pi j N^{-1} ik} p_i , \qquad (1.104)$$

где $k = 0,\ldots,N-1$, $j = \sqrt{-1}$.

Ядро преобразования Фурье $exp\left(-2\pi j N^{-1}\right)$ равно степени N из единицы в поле комплексных чисел. В конечном поле $GF(q^m)$ элемент α порядка *n* равен корню степени *n* из единицы.

Поскольку кодовое слово *c* является последовательностью, то преобразование является дискретным по времени. Кроме того, каждый элемент этой последовательности лежит в некотором поле Галуа, так что преобразование также должно принимать значения в некотором конечном поле.

Обозначаемое через *C* дискретное преобразование Фурье над конечным полем вектора *c* определяется следующим образом.

Пусть $c = (c_0, c_1, \ldots, c_{n-1})$ последовательность n элементов поля $GF(q)$ (где n делит $q^m - 1$ для некоторого m) и пусть $\alpha \in GF(q^m)$ элемент порядка n. *Преобразованием Фурье над конечным полем* вектора *c* называется последовательность $C = (C_0, C_1, \ldots, C_{n-1})$ элементов поля $GF(q^m)$, задаваемых равенством

$$C_j = \sum_{i=0}^{n-1} \alpha^{ij} c_i \ , \ \ j = 0,1,\ldots,n-1 . \qquad (1.105)$$

Дискретный индекс i принято называть временем, а c – *временной функцией* или *сигналом*. Аналогично индекс j можно назвать *частотой*, а C – *частотной функцией* или *спектром.*

В качестве длины преобразования Фурье можно выбрать произвольный делитель числа $q^m - 1$, но наиболее важную роль играют примитивные длины $n = q^m - 1$.

В последнем случае α является примитивным элементом поля $GF(q^m)$. В отличие от поля комплексных чисел в поле Галуа преобразование Фурье существует не для любой длины n, так как не для любого n в поле существует элемент этого порядка. Если m – наименьшее целое, такое, что делит $q^m - 1$, то над полем $GF(q)$ существует преобразование Фурье длины n и компоненты этого преобразования лежат в поле $GF(q^m)$.

Эти два вектора образуют пару $c \Leftrightarrow C$, связанную между собой следующим образом.

Теорема 1.24. Пусть характеристика поля $GF(q)$ равна p. Тогда вектор и его спектр связаны между собой равенствами

$$C_j = \sum_{i=0}^{n-1} \alpha^{ij} c_i \ ; \qquad (1.106)$$

$$c_i = \frac{1}{n \bmod p} \sum_{j=0}^{n-1} \alpha^{-ij} C_j \ . \qquad (1.107)$$

Доказательство

В любом поле $x^n - 1 = (x-1)(x^{n-1} + x^{n-2} + x^{n-3} + \ldots + x + 1)$.

По определению α элемент α^r при всех r является корнем многочлена в левой части. Следовательно, для всех по модулю n элемент α^r является корнем последнего многочлена.

Но это эквивалентно равенству

$$\sum_{i=0}^{n-1} \alpha^{rj} = 0 \ , \qquad (1.108)$$

$r \neq 0(mod\ n)$, если же $r = 0(mod\ n)$, то

$$\sum_{i=0}^{n-1} \alpha^{rj} = n(mod\ p), \tag{1.109}$$

что всегда отлично от нуля, если n не кратно характеристике p.

Комбинируя эти равенства, получаем

$$\sum_{j=0}^{n-1} \alpha^{-ij} \sum_{i=0}^{n-1} \alpha^{kj} c_i = \sum_{i=0}^{n-1} c_k \sum_{i=0}^{n-1} \alpha^{(k-i)j} = (n\ mod\ p) c_i\ . \tag{1.110}$$

Наконец $q^m - 1 = p^M - 1$ кратно n и поэтому не кратно p. Следовательно $n \neq 0(mod\ p)$.

Преобразование Фурье обладает многими сильными свойствами, которые переносятся на случай конечных полей. Примером является свойство свертки.

Теорема 1.25 (теорема свертки). Предположим, что компоненты c_i вектора c являются произведениями компонентов двух других векторов f и g, т.е.

$$c_i = f_i g_i, i = 0,1,2,\ldots,n-1. \tag{1.111}$$

Тогда преобразование Фурье вектора c можно записать

$$C_j = \frac{1}{n} \sum_{i=0}^{n-1} F((j-k)) G_k\ ; \qquad j = 0,1,2,\ldots,n-1, \tag{1.112}$$

где двойные скобки обозначают, что индексы вычисляются в арифметике по модулю n.

Доказательство. Найдем преобразование Фурье вектора с компонентами $c_i = f_i g_i$:

$$C_j = \sum_{i=0}^{n-1} \alpha^{ij} f_i \left(\frac{1}{n} \sum_{i=0}^{n-1} \alpha^{-ik} G_k \right) \left(\frac{1}{n} \right) \sum_{k=0}^{n-1} G_k \left(\sum_{i=0}^{n-1} \alpha^{i(j-k)} f_i \right) = \frac{1}{n} \sum_{i=0}^{n-1} F((j-k)) G_k\ . \tag{1.113}$$

Заметим также, что выбор $j = 0$ в формуле свертки

$$C_j = \sum_{i=0}^{n-1} \alpha^{ij} f_i g_i = \frac{1}{n} \sum_{i=0}^{n-1} F((j-k))G_k \qquad (1.114)$$

приводит к формуле типа равенства Парсеваля:

$$\sum_{i=0}^{n-1} \alpha^{ij} f_i g_i = \frac{1}{n} \sum_{i=0}^{n-1} F((j-k))G_k \,. \qquad (1.115)$$

Таким образом, умножение во временной области эквивалентно свертке в частотной области (полная аналогия с теорией линейных систем), верно также и обратное утверждение.

Иногда векторы c и C можно представить многочленами

$$c(x) = c_{n-1}x^{n-1} + \ldots + c_1 x + c_0, \quad C(z) = C_{n-1}z^{n-1} + \ldots + C_1 z + C_0 \,. \qquad (1.116)$$

Из этого представления видно, что j–й спектральный компонент

$$C_j = \sum_{i=0}^{n-1} c_i \alpha^{ij} = c\left(\alpha^j\right). \qquad (1.117)$$

Таким образом, $C_j = 0$ тогда и только тогда, когда α^j – корень $c(x)$. Аналогично i–й временной компонент

$$c_i = \frac{1}{n} \sum_{j=0}^{n-1} C_j \alpha^{-ij} = \frac{1}{n} C(\alpha^{-I}) \,. \qquad (1.118)$$

Таким образом, задание корней многочлена в одной области эквивалентно выбору нулевых соответствующих компонентов в другой. Это свойство оказывается очень полезным при выяснении корректирующих способностей циклических кодов.

Теорема 1.26. (i) элемент α^j является корнем многочлена $c(x)$тогда и только тогда, когда j–я частотная компонента C_j равна нулю;

(ii) элемент α^{-j} является корнем многочлена $C(x)$ тогда и только тогда, когда i–я временная компонента c_iравна нулю.

Доказательство. Утверждение (i) очевидно, так как

$$c(\alpha^j) = \sum_{i=0}^{n-1} c_i \alpha^{ij} = C_j \ . \qquad (1.119)$$

Утверждение (ii) доказывается тем же путем.

Пример. Рассмотрим вектор $C = (0,0,1,3)$ с элементами из $GF(5)$. Преобразование Фурье над конечным полем для этого вектора также определено над $GF(5)$ и его компоненты согласно (1.107)

$$C_j = \sum_{j=0}^{3} 2^{ij} c_i = 1 \cdot 2^{2j} + 3 \cdot 2^{3j}, \quad j = 0,1,2,3 .$$

В качестве элемента порядка 4 в поле $GF(5)$ выбран элемент 2. Таблицы умножения и сложения для этого поля приведены в табл. 1.3, 1.4. Произведя указанные операции, получаем $C = (4,3,3,0)$ в качестве искомого преобразования Фурье. Заметим, что характеристика поля $GF(5)$ равна 5 и что $4^{-1} = 4$.

Таким образом, обратное преобразование C, задаваемое формулой (1.108),

$$c_i = 4 \sum_{j=0}^{3} 2^{-ij} C_j = 4(4 + 3 \cdot 2^{-i} + 3 \cdot 2^{-2i}), \quad i = 0,1,2,3 .$$

Снова произведя указанные операции, получаем исходный вектор c. Заметим, что в этот вектор входят два нулевых компонента c_0 и c_1.

Теорема 1.27 (свойство сдвига). Если $\{c\} \leftrightarrow \{C\}$ является парой преобразования Фурье, то парами преобразования Фурье является также

$$\{\alpha^i c_i\} \leftrightarrow \{C_{((j+1))}\} \quad \text{и} \quad \{c_{((i-1))}\} \leftrightarrow \{\alpha^j C_j\}. \qquad (1.120)$$

Доказательство. Получается непосредственной подстановкой.

РАЗДЕЛ 2
ИССЛЕДОВАНИЕ И РАЗРАБОТКА АЛГЕБРАИЧЕСКИХ МЕТОДОВ ПОСТРОЕНИЯ НЕРЕКУРСИВНЫХ СВЕРТОЧНЫХ КОДОВ

Исследуются методы сверточного кодирования. Разрабатывается алгебраический метод сверточного кодирования, отличающийся от известных представлением порождающих многочленов сверточного кода через порождающий многочлен недвоичного циклического кода, ограниченного на произвольное подполе, и позволяющий строить коды с наперед заданными конструктивными свойствами. Разрабатываются алгоритмы формирования порождающих многочленов сверточных кодов, алгоритмы построения несистематических сверточных кодов с заданными конструктивными характеристиками. Исследуются свойства несистематических сверточных кодов, построенных с использованием разработанного метода.

2.1. Исследование и анализ известных методов сверточного кодирования

Одним из перспективных направлений в развитии теории помехоустойчивого кодирования является разработка методов непрерывного (древовидного) кодирования. Суть этих методов состоит в представлении информационного потока данных блоками (кадрами) длины k^0 и сопоставлении с каждым из них блока кодовых символов. При этом каждый полученный кадр кодовых символов формируется с учетом предыдущих r кадров информационных символов. На рис. 2.1 представлена обобщенная схема непрерывного кодера.

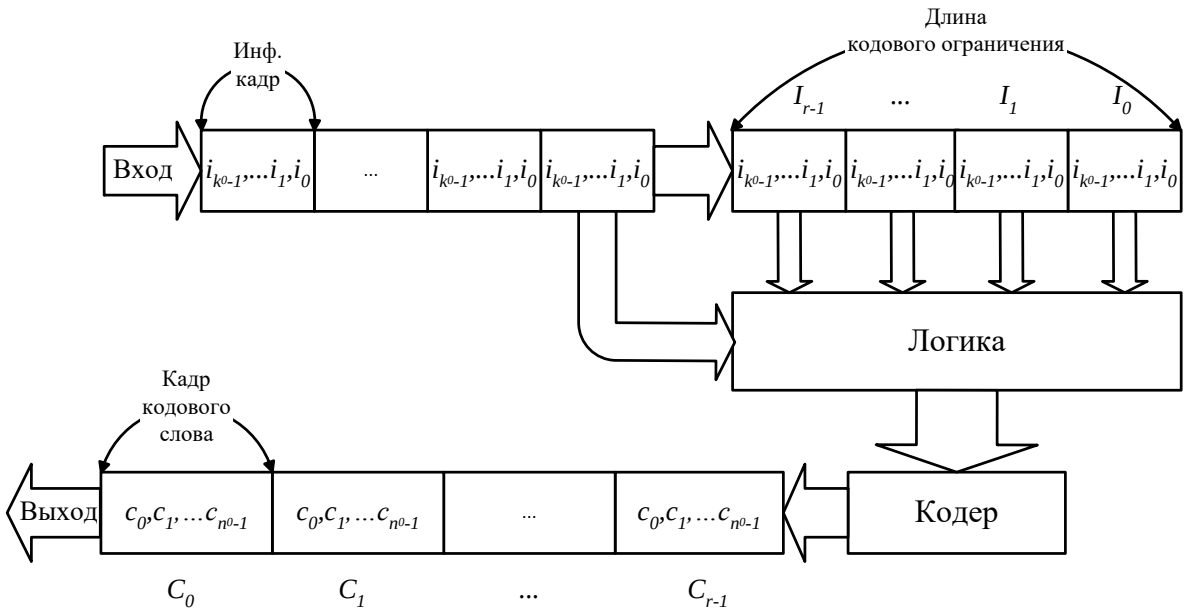


Рис. 2.1. Схема непрерывного кодера в виде регистра сдвига

Работа кодера состоит в следующем. Информационная последовательность вводится в кодер, начиная с нулевого момента времени и до бесконечности. Поток входящих информационных символов разбивается на информационные кадры по k^0 символов каждый. Кадр может в частности состоять из одного символа, в кодере хранится r кадров. В течение каждого сдвига в регистр сдвига вводится новый кадр информационных символов. Кодер по введенному кадру и r хранящихся в нем кадрах вычисляет один кадр кодового слова, имеющий длину n^0 символов. Этот кадр кодового слова выводится из кодера, как только следующий кадр информационных символов поступает в него. Следовательно, каждым k^0 информационным символам соответствуют n^0 кодовых символов.

Бесконечное множество всех бесконечно длинных кодовых слов, получаемых при поступлении в кодер всех возможных входных последовательностей, называется древовидным (непрерывным) (n^0, k^0) – кодом. Скорость кода R определяется как

$$R = k^0 / n^0.$$

Важной характеристикой непрерывного кода является длина кодового ограничения

$$\nu = r \cdot k^0.$$

Минимальным кодовым расстоянием непрерывного кода d называется минимальное расстояние для любых различных кодовых слов, соответствующих $r + 1$ различным информационным кадрам с ненулевым начальным кадром.

Если непрерывный код линеен, то минимальное расстояние равно минимальному весу из всех ненулевых кодовых слов, соответствующих произвольной входной последовательности с ненулевым начальным кадром. Набор минимальных весов d_l, $l = 1, 2, 3, \ldots$ произвольных кодовых слов, соответствующих l различным информационным кадрам с ненулевым начальным кадром, называется дистанционным профилем непрерывного кода.

Свободным расстоянием непрерывного кода называется $d_\infty = max(d_l)$. Очевидно, что $d = d_{r+1} \leq d_{r+2} \leq \ldots \leq d_\infty$. При декодировании непрерывных кодов синдромными и пороговыми методами пользуются величиной d. При декодировании методом Витерби пользуются величиной d_∞.

При построении непрерывных кодов используют также другие параметры кода. Так, величина

$$k = (r + 1) \cdot k^0$$

непосредственно связана с длиной кодового ограничения и называется информационной длиной слова непрерывного кода. Соответствующая ей мера кодовых последовательностей называется длиной кодового блока:

$$n = (r + 1) \cdot n^0 = k \cdot n^0 / k^0.$$

Кодовая длина блока – длина кодового слова, на которой сохраняется влияние одного кадра информационных символов. В большинстве известных практических примеров значения k^0 и n^0 выбираются равными небольшим целым числам, как правило, $k^0 = 1$.

Это означает, что выбор скорости ограничен: $R = 1 / m$, $m \in \{1, 2, \ldots\}$ – в большинстве известных примеров невозможно построить непрерывный код со скоростью, достаточно близкой к единице, как это делается для большинства блоковых кодов (циклические коды БЧХ, Рида–Соломона (РС) и др.).

На практике нашли применение несколько классов непрерывных кодов. Их общая классификация представлена на рис. 2.2.

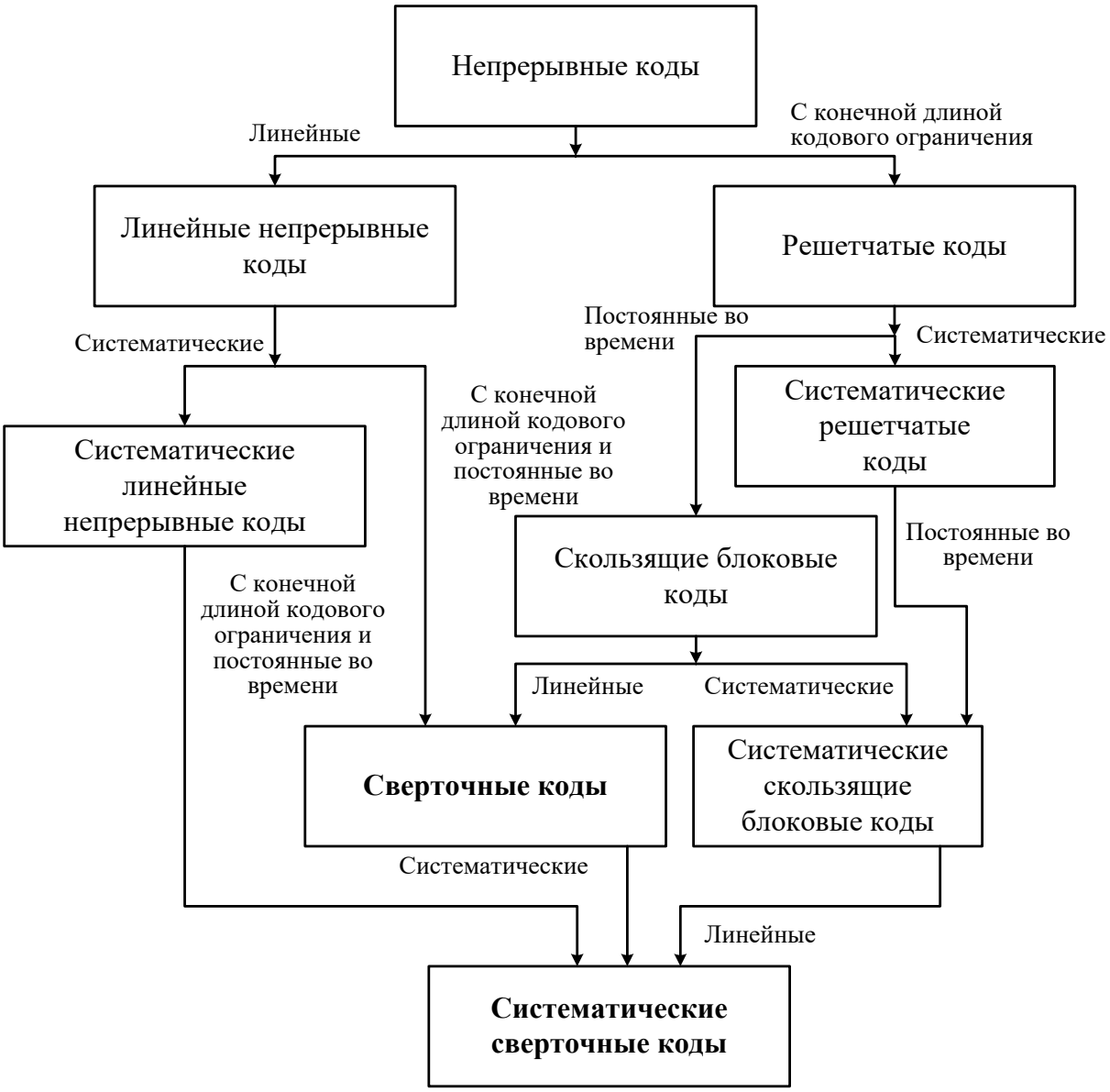


Рис. 2.2. Классификация непрерывных кодов

Частные примеры древовидных кодов получаются различными комбинациями следующих свойств:

1. *Конечность длины кодового ограничения.* Практические непрерывные коды всегда имеют конечную длину кодового ограничения. Древовидный (n^0, k^0) код с конечной длиной кодового ограничения ν, длиной слова $\nu + k^0$ называется также решетчатым кодом.

2. *Постоянство во времени.* Если две различные входные последовательности совпадают во всем, но с временным сдвигом на целое число кадров, то соответствующие им кодовые последовательности также совпадают во всем, но с временным сдвигом на то же самое число кадров.

3. *Линейность*. Кодовая последовательность любой линейной комбинации двух информационных последовательностей совпадает с такой же линейной комбинацией кодовых последовательностей этих двух информационных последовательностей. Иначе говоря, если i_1 и i_2 являются двумя информационными последовательностями с кодовыми словами $c(i_1)$ и $c(i_2)$, то $a \cdot i_1 + b \cdot i_2$ соответствует кодовая последовательность $c \cdot (a \cdot i_1 + + b \cdot i_2) = a \cdot c \cdot (i_1) + b \cdot c \cdot (i_2)$.

4. *Систематичность*. Каждый кадр информационных символов составляет первые k^0 символов первого из тех кадров кодовой последовательности, на которые влияет данный кадр информационной последовательности.

Наиболее важным классом непрерывных кодов являются сверточные коды, обладающие свойством линейности и постоянства во времени. Сверточный код, удовлетворяющий условию систематичности, называется систематическим сверточным кодом (см. рис. 2.2).

Развитие методов сверточного кодирования существенно отличается от известных методов блочного кодирования. При построении хороших классов блочных кодов развитие получили алгебраические методы. Они позволяют строить блочные коды с наперед заданными свойствами.

Подавляющее большинство хороших и наиболее употребимых сверточных кодов получено переборным методом. Путем просмотра большого числа кодов и последующего выбора найдены сверточные коды с хорошими свойствами. Основным недостатком переборного метода построения сверточных кодов является быстрый рост вычислительных затрат для его реализации. Так, выходная последовательность произвольного сверточного кода зависит от $\nu = r \cdot k^0$ входных символов (см. рис. 2.1). Т.е. произвольный сверточный код над полем *GF(q)* можно однозначно задать только путем определения “логики преобразований входных символов”, которая может быть представлена регистром сдвига с ν ячейками. Следовательно, для полного перебора всех возможных сверточных кодов с кодовым ограничением ν следует перебрать, как минимум,

$$N = \sum_{i=0}^{\nu} (q-1)^i C_\nu^i = q^\nu = q^{rk^0}$$

вариантов возможных устройств. Так, для перебора всех двоичных сверточных кодов с кодовым ограничением 100 бит необходимо выполнить перебор $2^{100} \approx 10^{30}$ различных кодеров и выбрать из них лучший, что является практически неразрешимой задачей. На практике переборными методами реализован поиск хороших двоичных сверточных кодов до $\nu \le 14$. Очевидно, что с практической точки зрения этот подход неконструктивен. Переборный метод построения сверточных кодов малоэффективен по причине своей низкой производительности.

Возникает противоречие между необходимостью разработки хороших сверточных кодов с наперед заданными конструктивными свойствами и возможностями существующих методов их построения.

Для разрешения выявленного противоречия предлагается алгебраический метод построения сверточных кодов. Он является ограничением недвоичного циклического кода на произвольное поле и позволяет алгебраически задавать несистематические коды с требуемыми свойствами.

2.2. Исследование и разработка алгебраического метода построения сверточных кодов передаваемой информации

Алгебраический подход к построению сверточных кодов состоит в представлении сверточного кода через порождающий многочлен недвоичного циклического кода и позволяет алгебраически задавать его параметры для скорости кодирования $R = 1 / m$. Этот подход позволяет использовать мощный математический аппарат циклического кодирования в целях алгебраического построения сверточных кодов. В ряде случаев рассмотрены многочисленные примеры построения хороших сверточных кодов с использованием этого метода. Рассмотрим более подробно.

Алгебраический метод построения сверточных кодов для $R = 1/m$

Рассмотрим несистематический сверточный (n, k) – код над $GF(q)$ с параметрами: $k^0 = 1$, $n^0 = m \cdot k^0 = m$, $k = r + 1$, $n = (r + 1) \cdot n^0 = k \cdot m$ и скоростью $R = 1 / m$, построенный с помощью несистематического сверточного кодера (см. рис. 2.3).

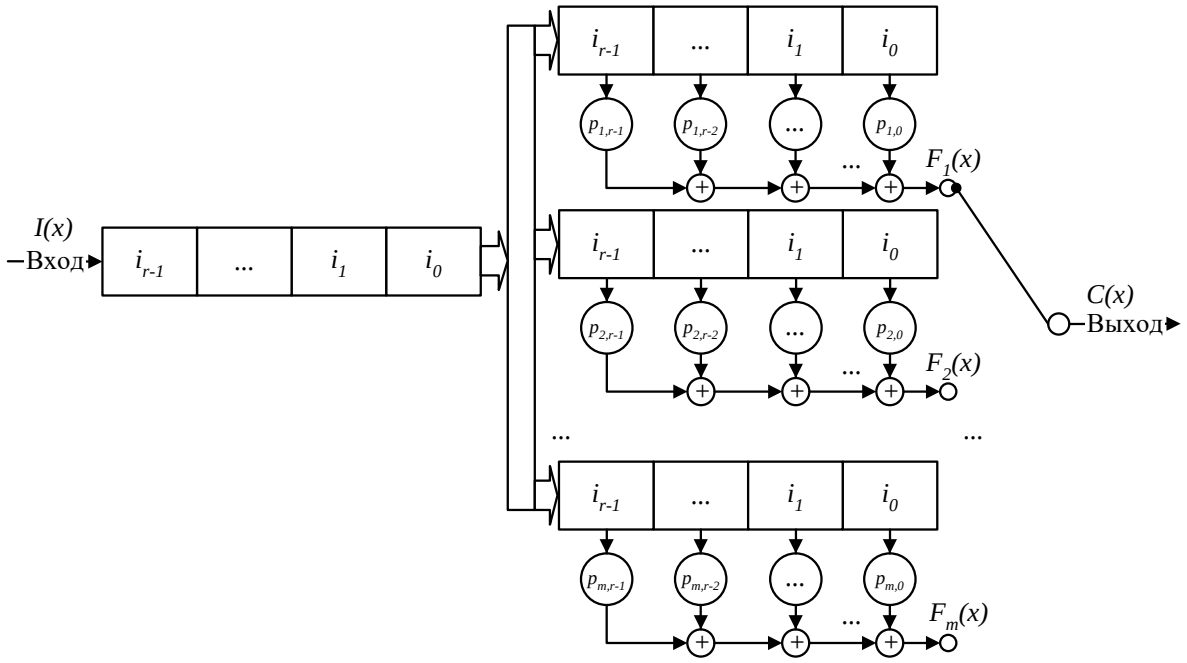


Рис. 2.3. Схема несистематического сверточного кодера при $R = 1/m$

Для описания процесса кодирования информации несистематическим сверточным кодером использован подход, состоящий в формальном определении сверточного кода через недвоичный циклический код над $GF(q^m)$, где степень расширения поля m соответствует числу q – ичных многочленов сверточного кода.

Пусть многочлен

$$I(x) = i_{r-1}x^{r-1} + i_{r-2}x^{r-2} + ... + i_1 x + i_0 \qquad (2.1)$$

является информационной последовательностью, подлежащей кодированию (в общем случае многочлен $I(x)$ может быть бесконечной длины), а многочлены

$$\begin{aligned} P_1(x) &= p_{1,r-1}x^{r-1} + p_{1,r-2}x^{r-2} + ... + p_{1,1}x + p_{1,0}; \\ P_2(x) &= p_{2,r-1}x^{r-1} + p_{2,r-2}x^{r-2} + ... + p_{2,1}x + p_{2,0}; \qquad (2.2) \\ &\dots \\ P_m(x) &= p_{m,r-1}x^{r-1} + p_{m,r-2}x^{r-2} + ... + p_{m,1}x + p_{m,0}, \end{aligned}$$

будут порождающими многочленами данного сверточного кода. Коэффициенты при x в выражениях (2.1) и (2.2) являются элементами $GF(q)$. Если один из многочленов в выражении (2.2) имеет меньший показатель степени, то добавим в этот многочлен необходимое (до большего) количество нулевых коэффициентов при старших степенях формальной переменной x.

Процесс кодирования информации рассматриваемым сверточным кодером опишем следующим образом. Информационная последовательность $I(x)$ вида (2.1) поступает в кодер сверточного кода, где происходит ее умножение на многочлены $P_1(x)$ … $P_m(x)$ вида (2.2) и получение последовательностей $F_1(x)$ … $F_m(x)$ соответственно:

$$\begin{aligned} F_1(x) &= I(x)P_1(x) = s_{1,2r-2}x^{2r-2} + s_{1,2r-3}x^{2r-3} + ... + s_{1,1}x + s_{1,0}; \\ F_2(x) &= I(x)P_2(x) = s_{2,2r-2}x^{2r-2} + s_{2,2r-3}x^{2r-3} + ... + s_{2,1}x + s_{2,0}; \qquad (2.3) \\ &\dots \\ F_m(x) &= I(x)P_m(x) = s_{m,2r-2}x^{2r-2} + s_{m,2r-3}x^{2r-3} + ... + s_{m,1}x + s_{m,0}, \end{aligned}$$

где $s_{i,j}$ – коэффициент в многочлене $F_i(x)$ при x^j в результате перемножения многочленов $I(x)$ и $P_i(x)$.

Кодовое слово $C(x)$ формируется путем последовательного считывания символов при одинаковых степенях многочленов $F_1(x)$ … $F_m(x)$, т.е.:

$$C(x)=(s_{1,2r-2},s_{2,2r-2},\ldots,s_{m,2r-2})x^{2r-2}+(s_{1,2r-3},s_{2,2r-3},\ldots,s_{m,2r-3})x^{2r-3}+\ldots+$$
$$+(s_{1,1},s_{2,1},\ldots,s_{m,1})x+(s_{1,0},s_{2,0},\ldots,s_{m,0}). \quad (2.4)$$

Если на вход схемы несистематического сверточного кодера подать информационный вектор вида $\{0, 0, \ldots, 1\}$, то информационный многочлен запишется как $I(x)=1$, а кодовое слово запишется в виде

$$P(x)=(p_{1,r-1},p_{2,r-1},\ldots,p_{m,r-1})x^{r-1}+(p_{1,r-2},p_{2,r-2},\ldots,p_{m,r-2})x^{r-2}+\ldots+$$
$$+(p_{1,1},p_{2,1},\ldots,p_{m,1})x+(p_{1,0},p_{2,0},\ldots,p_{m,0}). \quad (2.5)$$

Последнее выражение однозначно определяет несистематическое правило сверточного кодирования.

Рассмотрим конечное поле $GF(q^m)$, построенное по кольцу многочленов, с коэффициентами над $GF(q)$. В выражении (2.5) каждому набору $\{p_{1,i}, p_{2,i}, \ldots, p_{m,i}\}$ сопоставим элемент поля $\beta_i \in GF(q^m)$, такой, что

$$\beta_i = p_{1,i} + p_{2,i}x^2 + \ldots + p_{m,i}x^m .$$

Выражение (2.5) запишем в виде

$$P(x)=\beta_{r-1}x^{r-1}+\beta_{r-2}x^{r-2}+\ldots+\beta_1x+\beta_0. \quad (2.6)$$

Если выражение (2.6) суть порождающий многочлен недвоичного (N, K, D) циклического кода над $GF(q^m)$, то справедлива следующая теорема.

Теорема 2.1. Несистематический сверточный код над $GF(q)$ (рис. 2.3) с $R = 1/m$ однозначно задается многочленом $P(x)$ над $GF(q^m)$ вида (2.6). Если многочлен (2.6) задает недвоичный (N, K, D) циклический код над $GF(q^m)$, то он однозначно

определяет *(n, k)* несистематический сверточный код над *GF(q)* с кодовым ограничением $\nu = r \cdot k^0 = r$ и параметрами

$$\begin{cases} k^0 = 1, \\ n^0 = m, \\ k = r + 1, \\ n = k \cdot m, \\ R = 1/m, \\ d_\infty \geq D. \end{cases}$$

Доказательство. Действительно, недвоичный *(N, K, D)* циклический код над $GF(q^m)$, порожденный многочленом *P(x),* степени *r* однозначно определяет набор регистров сдвига, соединенных связями (рис. 2.3) и задает рекуррентное правило кодирования, т.е. однозначного соответствия входной (информационной) последовательности в кодовую (выходную) последовательность

$$C(x) = I(x) \cdot P(x).$$

Параметры несистематического кода соответствуют рассмотренному выше примеру (рис. 2.3), т.е.: $k^0 = 1$, $n^0 = m$. Циклический код над $GF(q^m)$ задает длину кодирующего регистра и, соответственно, число хранящихся в кодере информационных кадров. Длина кодового ограничения ν, конструктивные параметры *n* и *k*, скорость *R* сверточного кодирования определяются выражениями

$$\nu = r \cdot k^0 = r,\ k = r + 1,\ n = (r + 1) \cdot n^0 = k \cdot m,\ R = 1 / m.$$

Если на вход кодирующего устройства подать информационный блок данных длиной *K* *q* – ичных символов, то считанная с выхода кодовая последовательность длиной *N* q^m – ичных символов суть кодовое слово циклического *(N, K, D)* кода над $GF(q^m)$. Следовательно, два любых кодовых блока, соответствующих двум произвольным входным последовательностям длиной *K* *q* – ичных символов, будут отличаться в *D* q^m – ичных символов. Последовательное считывание символов при одинаковых степенях многочленов

$F_1(x)$ ... $F_m(x)$ – суть отображение элементов поля $GF(q^m)$ в элементы образующего поля $GF(q)$, которое не уменьшает кодовое расстояние между произвольными q – ичными кодовыми словами длины $N \cdot m$.

По условию теоремы длина информационного кадра $k^0 = 1$, следовательно, для кодовых слов, соответствующих K различным информационным кадрам, $d_K \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_{r+1} \leq d_{r+2} \leq \ldots \leq d_\infty$. Если выполняется условие $K \leq r$, то, очевидно, $d_\infty \geq d \geq d_K$. Если $K > r$, то выполняется лишь равенство $d_\infty \geq d_K$.

Рассмотренное обобщение несистематического сверточного (n, k) кода и теорема 2.1 позволяют алгебраически задавать параметры сверточного кода для произвольной длины кодового ограничения. С использованием такого подхода в работах [39-41] подробно рассмотрены алгоритмы формирования порождающих многочленов сверточного кода и алгебраические алгоритмы построения на их основе двоичных сверточных кодов с заранее заданными конструктивными свойствами. Отметим, что в результате выполнения этих алгоритмов удается упростить процедуру построения сверточных кодов с предварительной оценкой их параметров. Уточнение кодового расстояния (условие $d_\infty \geq D$) позволяет, как правило, улучшить кодовые характеристики. В работе [24] приведены многочисленные примеры использования такого подхода, причем кодовые характеристики полученных сверточных кодов являются одними из лучших известных на сегодняшний день кодов.

К сожалению, рассмотренный алгебраический метод позволяет строить сверточные коды только для скорости $R = 1 / m$, где m – степень расширения базового поля, над которым задается порождающий многочлен циклического кода. Это обстоятельство сужает область практического использования рассмотренного метода. Кроме того, наибольший энергетический выигрыш от кодирования большинство линейных кодов позволяет при скорости $R \approx 1 / 2 – 2 / 3$.

Предлагается алгебраический метод сверточного кодирования, позволяющий снять указанные ограничения и алгебраически задать несистематические сверточные коды с $R = k^0 / m$.

Алгебраический метод построения сверточных кодов для $R = k^0/m$

В основе рассмотренного алгебраического метода построения сверточных кодов лежит ограничение недвоичного циклического кода над $GF(q^m)$ на подполе $GF(q)$. Подобное представление позволяет алгебраически определить несистематический (n, k) сверточный код с $R = 1 / m$.

Для снятия ограничения по скорости кодирования предлагается алгебраический метод построения сверточных кодов, в основе которого лежит ограничение недвоичного циклического кода над $GF(q^m)$ на произвольное подмножество $H \subseteq GF(q^m)$, $\left| H \right| \geq \left| GF(q) \right|$.

Если $\left| H \right| = \left| GF(q) \right|$, получим, как частный случай, изложенный выше метод.

Рассмотрим несистематический сверточный (n, k) – код над $GF(q)$ со скоростью $R = k^0 / m$ (см. рис. 2.4).

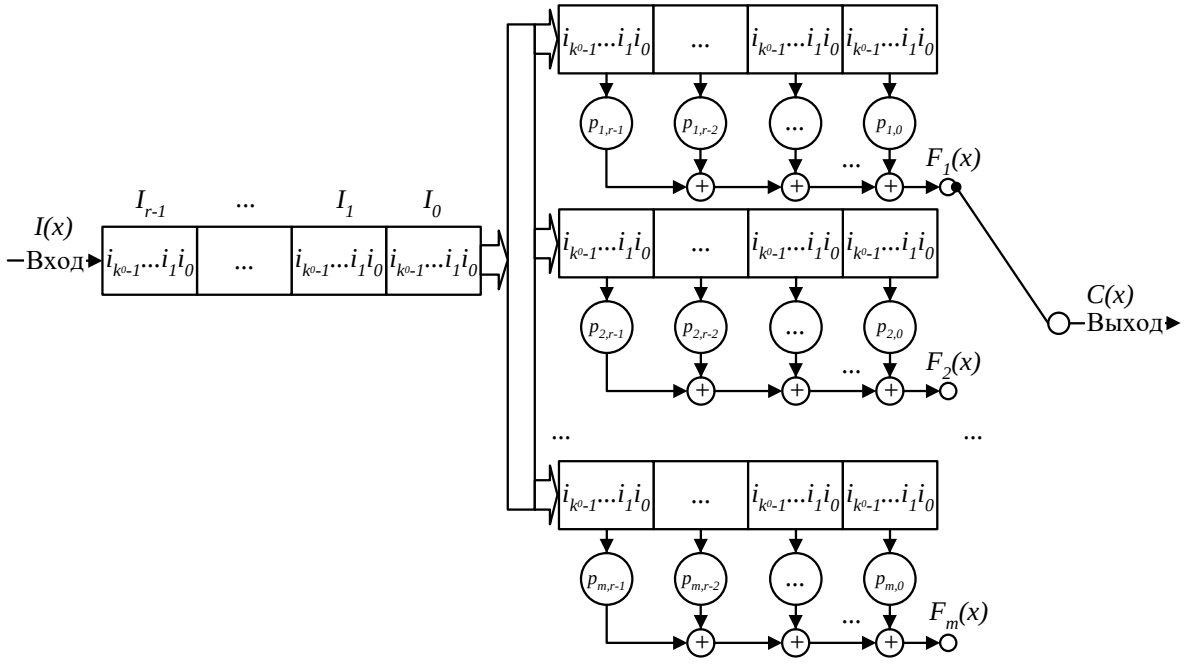


Рис. 2.4. Схема несистематического сверточного кодера при $R = k^0/m$

Разобьем входную информационную последовательность на информационные кадры по $k^0 \geq 1$ символов, каждый символ которых принадлежит $GF(q)$. В общем случае информационная последовательность может быть бесконечной длины, т.е. состоять из бесконечного числа информационных кадров по k^0 символов.

Сопоставим каждому информационному кадру из k^0 символов один символ из множества $H \subseteq GF(q^m)$, $/H/ \geq /GF(q)/$.

Тогда информационный многочлен представим в виде

$$I(x) = I_{r-1}x^{r-1} + I_{r-1}x^{r-2} + ... + I_1 x + I_0, \qquad (2.7)$$

где $I_j \in H, j = 0,..., r-1, \log_q /H/ = k^0, m \geq k^0$.

Пусть, как и прежде, многочлены $P_1(x)$, $P_2(x)$, ..., $P_m(x)$ – порождающие многочлены представленного на рис. 2.4 несистематического сверточного кода.

Процесс кодирования информации – информационная последовательность $I(x)$ вида (2.7) поступает в кодер (рис. 2.4), где происходит ее умножение на многочлены $P_1(x)$ ... $P_m(x)$ вида (2.2). Получим последовательности $F_1(x)$ ... $F_m(x)$:

$$\begin{aligned}
&F_1(x) = I(x)P_1(x) = S_{1,2r-2}x^{2r-2} + S_{1,2r-3}x^{2r-3} + ... + S_{1,1}x + S_{1,0};\\
&F_2(x) = I(x)P_2(x) = S_{2,2r-2}x^{2r-2} + S_{2,2r-3}x^{2r-3} + ... + S_{2,1}x + S_{2,0}; \qquad (2.8)\\
&\ldots\\
&F_m(x) = I(x)P_m(x) = S_{m,2r-2}x^{2r-2} + S_{m,2r-3}x^{2r-3} + ... + S_{m,1}x + S_{m,0},
\end{aligned}$$

где $S_{i,j}$ – коэффициент в многочлене $F_i(x)$ при x^j в результате перемножения многочлена $I(x)$ вида (2.7) и многочленов $P_i(x)$ вида (2.2).

Кодовое слово $C(x)$ формируется путем последовательного считывания символов при одинаковых степенях многочленов $F_1(x)$ ... $F_m(x)$, т.е.:

$$\begin{aligned}
C(x) = &(S_{1,2r-2}, S_{2,2r-2}, ..., S_{m,2r-2})x^{2r-2} + (S_{1,2r-3}, S_{2,2r-3}, ..., S_{m,2r-3})x^{2r-3} + ...\\
&+ (S_{1,1}, S_{2,1}, ..., S_{m,1})x + (S_{1,0}, S_{2,0}, ..., S_{m,0}).
\end{aligned} \qquad (2.9)$$

Если на вход сверточного кода подать информационный вектор вида *{0, 0, ...,1}*, то информационный многочлен запишется как *I(x)=1*, а кодовое слово (2.9) запишется в виде порождающего многочлена циклического кода, т.е. *C(x) = P(x)*. Таким образом, порождающий многочлен циклического кода однозначно определяет несистематическое правило сверточного кодирования. Справедлива следующая теорема.

Теорема 2.2. Если зафиксировать конечное множество *H* элементов поля $GF(q^m)$, причем $\log_q / H / = k^0$, $m \geq k^0$, то произвольный многочлен степени *r* с коэффициентами над $GF(q^m)$ полностью определяет несистематический сверточный *(n, k)* код над *GF(q)* с информационным кадром длины k^0, кодовым ограничением $\nu = r \cdot k^0$ и параметрами:

$$\begin{cases} n^0 = m, \\ k = (r+1) \cdot k^0, \\ n = k \cdot n^0 / k^0, \\ R = k^0 / m, m \geq k^0. \end{cases}$$

Доказательство. Кодирование, по определению, это процесс однозначного сопоставления (соответствия) информационной и кодовой последовательностей. Пусть задан произвольный многочлен *P(x)* над $GF(q^m)$ степени *r* вида (2.6) и входная последовательность над *GF(q)*.

Представим информационную последовательность в виде многочлена (2.7) с коэффициентами над *H*. Т.е. коэффициенты многочлена *I(x)* в выражении (2.7) являются многочленами над *GF(q)* степени *m – 1*:

$$I_j = z_{m-1}x^{m-1} + \ldots + z_{k^0}x^{k^0} + z_{k^0-1}x^{k^0-1} + z_{k^0-2}x^{k^0-2} + \ldots + z_1 x + z_0, \qquad (2.10)$$

где $z_i \in GF(q)$, причем $m - k^0$ коэффициентов z_i равны нулю. Положим, для определенности, $z_i = 0$ для $i = k^0, \ldots, m-1$. Первые k^0 элементов z_i в выражении (2.10) образуют информационный кадр k^0 символов над *GF(q)*. Определенное таким образом отображение символов *GF(q)* в символы $GF(q^m)$ является однозначным соответствием.

Недвоичный *(N, K, D)* циклический код над $GF(q^m)$, порожденный многочленом *P(x)* степени *r*, однозначно определяет набор регистров сдвига соединенных связями (рис. 2.4) и задает рекуррентное правило кодирования, т.е. однозначного соответствия входной (информационной) последовательности в кодовую (выходную) последовательность: $C(x) = I(x) \cdot P(x)$. Параметры несистематического кода соответствуют рассмотренному выше примеру (рис. 2.4), т.е. каждому информационному кадру длиной k^0 символов над *GF(q)* (или, что эквивалентно, каждому символу из множества *H*) ставится в соответствие кадр кодовых символов длиной n^0.

Степень *r* порождающего многочлена *P(x)* циклического *(N, K, D)* кода над $GF(q^m)$ задает длину кодирующего регистра и, соответственно, число хранящихся в кодере информационных кадров. Следовательно, длина кодового ограничения ν, конструктивные параметры *n* и *k* и скорость *R* сверточного кодирования определяются, соответственно, следующими выражениями:

$$\nu = r \cdot k^0,\ k = (r + 1) \cdot k^0,\ n = k \cdot n^0 / k^0,\ R = k^0 / m,\ m \geq k^0.$$

Лемма 1. Если существует такое целое *w*, что $m = w \cdot k^0$, то порождающий многочлен степени *r (N, K, D)* циклического кода над $GF(q^m)$ полностью определяет несистематический сверточный (n_*, k_*, d_*) код над $GF(q^{k^0})$ с кодовым ограничением $\nu_* = r \cdot k_*^{\,0} = r$ и параметрами

$$\begin{cases} k_*^0 = 1, \\ n_*^0 = m, \\ k_* = r_* + 1, \\ n_* = (r+1) \cdot n_*^0 = k_* \cdot m, \\ R = 1/w. \end{cases}$$

Доказательство. Согласно теореме 2.2 произвольный многочлен степени *r* с коэффициентами над $GF(q^m)$ полностью определяет несистематический сверточный *(n, k, d)* код над *GF(q)* с кодовым ограничением ν, причем $n^0 = m$, $\nu = r \cdot k^0$,

$k = (r + 1) \cdot k^0$, $n = k \cdot n^0 / k^0$, $R = k^0 / m$, $m \geq k^0$, $k^0 = log_q \left| H \right|$, $H \subseteq GF(q^m)$. Если каждый информационный кадр длиной k^0 q – ичных символов представить одним q^{k^0} – ичным символом, то получим несистематический сверточный (n_*, k_*, d_*) код над $GF(q^{k^0})$, где $GF(q^{k^0})$ изоморфно множеству H. На вход такого кодера подступает K информационных кадров по одному q^{k^0} – ичному символу, следовательно, $k_*^0 = 1$. С выхода кодера снимается кодовая последовательность длиной N q^m – ичных символов. Если при этом выполняется равенство $m = w \cdot k^0$ для произвольного целого w, то $n_*^0 = w$. Тогда, очевидно, выполняются равенства: $\nu = r$; $k = r + 1$; $n = k \cdot w$ $R = 1 / w$, а по теореме 2.2: $C(x) = I(x) \cdot P(x)$, что соответствует обобщению теоремы 2.1 на случай несистематических сверточных кодов над $GF(q^{k^0})$.

Лемма 2. Если $\left| H \right| = \left| GF(q) \right|$ получим, как частный случай теоремы 2.2, алгебраически заданный сверточный код для $R = 1 / m$, что соответствует результату теоремы 2.1.

Доказательство. Действительно, если $\left| H \right| = \left| GF(q) \right|$, то по теореме 2.2 получим $k^0 = 1$. Следовательно, процесс сверточного кодирования соответствует ограничению поля $GF(q^m)$ на подполе $GF(q)$ и $k^0 = 1$, $n^0 = m$, $\nu = r \cdot k^0 = r$, $k = r + 1$, $n = k \cdot n^0 / k^0$, $R = 1 / m$, $C(x) = I(x) \cdot P(x)$, что соответствует результату теоремы 2.1.

Лемма 3. Если $q = 2^{m^*}$, то получим, как частный случай теоремы 2.2, алгебраически заданный двоичный сверточный код с $R = k^0 / u$, причем $u = m \cdot m^*$.

Доказательство. По теореме 2.2 имеем несистематический сверточный (n, k, d) код над $GF(q)$ с кодовым ограничением $\nu = r \cdot k^0$ и параметрами: $n = k \cdot n^0 / k^0$; $k = (r + 1) \cdot k^0$; $R = k^0 / m$. Если на вход такого кодера подать информационный кадр из $m^* \cdot k^0$ двоичных символов (что эквивалентно подаче кадра из k^0 q – ичных символов), а снятый с выхода кадр кодового слова – q^m–ичный символ преобразовать в $m \cdot m^*$ бит, получим

однозначное отображение – двоичное правило кодирования. Подставив эти параметры в результат теорем 2.2–2.3, получим: длина двоичного информационного кадра $k_2^0 = m^* \cdot k^0$; $n_2^0 = u$; $\nu_2 = r \cdot m^* \cdot k^0$; $k_2 = (r + 1) \cdot m^* \cdot k^0$; $n_2 = k \cdot n^0 / (m^* \cdot k^0)$; $R = m^* \cdot k^0 / u$; $u \geq m^* \cdot k^0$; $k^0 = \log_q \left| H \right|$; $H \subseteq GF(q^m)$.

Лемма 4. Если $\left| H \right| = \left| GF(q^m) \right|$, получим отображение информационной последовательности самое в себя, а процесс кодирования будет безизбыточным.

Доказательство. Действительно, если $\left| H \right| = \left| GF(q^m) \right|$ то $k^0 = m$ и процесс сверточного кодирования соответствует ограничению поля $GF(q^m)$ на поле $GF(q^m)$, т.е. процесс кодирования соответствует отображению элементов поля в элементы этого же поля. Подставив значение $k^0 = m$ в результат теоремы 2.2, получим: $n^0 = m$; $\nu = r \cdot m$; $k = (r + 1) \cdot m$; $n = k \cdot n^0 / k^0 = k$; $R = k / k = 1$. Следовательно, кодирование безизбыточное, а при условии бесконечности многочлена $I(x)$ информационная последовательность отображается самое в себя.

Приведем *пример*. Зафиксируем конечное поле $GF(2^2)$, построенное по кольцу многочленов по модулю $g(z) = z^2 + z + 1$. Поле $GF(2^2)$ состоит из четырех элементов: $\alpha^{-\infty} = 0$; $\alpha^0 = 1$; $\alpha^1 = z$; $\alpha^2 = z+1$.

Зафиксируем конечное поле $GF(4^3)$, построенное по кольцу многочленов по модулю $g(x) = x^3 + x^2 + x + 3$. Коэффициенты многочлена $g(x)$ – суть элементы поля $GF(2^2)$. В табл. 2.1 представлены элементы поля $GF(4^3)$ по классам сопряженных элементов, порядки элементов поля и степени минимальных многочленов.

Таблица 2.1

Структура конечного поля $GF(4^3)$

α^i	α^{4i}	α^{16i}	$deg(\alpha^i)$	$deg(f_i)$
α^0 (1 0 0)				1
α^1 (0 1 0)	α^4 (3 2 0)	α^{16} (2 3 0)	63	3
α^2 (0 0 1)	α^8 (2 0 3)	α^{32} (3 0 2)	63	3
α^3 (3 1 1)	α^{12} (2 2 3)	α^{48} (2 3 2)	21	3
α^5 (0 3 2)	α^{20} (1 1 1)	α^{17} (0 2 3)	63	3
α^6 (1 2 1)	α^{24} (2 3 3)	α^{33} (1 1 2)	21	3
α^7 (3 0 3)	α^{28} (2 0 2)	α^{49} (1 0 1)	9	3
α^9 (2 1 3)	α^{36} (0 2 2)	α^{18} (2 3 1)	7	3
α^{10} (2 1 2)	α^{40} (2 2 1)	α^{34} (1 3 3)	63	3
α^{11} (1 0 3)	α^{44} (0 0 2)	α^{50} (3 0 1)	63	3
α^{13} (2 1 1)	α^{52} (3 2 3)	α^{19} (3 3 2)	63	3
α^{14} (3 3 0)	α^{56} (1 1 0)	α^{35} (2 2 0)	9	3
α^{15} (0 3 3)	α^{60} (3 1 2)	α^{51} (3 2 1)	21	3
α^{21} (3 0 0)			3	1
α^{22} (0 3 0)	α^{25} (2 1 0)	α^{37} (1 2 0)	63	3
α^{23} (0 0 3)	α^{29} (1 0 2)	α^{53} (2 0 1)	63	3
α^{26} (0 2 1)	α^{41} (3 3 3)	α^{38} (0 1 2)	63	3
α^{27} (3 1 3)	α^{45} (1 2 2)	α^{54} (3 3 1)	7	3
α^{30} (1 3 2)	α^{57} (0 1 1)	α^{39} (1 2 3)	21	3
α^{31} (1 3 1)	α^{61} (1 1 3)	α^{55} (3 2 2)	63	3
α^{42} (2 0 0)			3	1
α^{43} (0 2 0)	α^{46} (1 3 0)	α^{58} (3 1 0)	63	3
α^{47} (0 1 3)	α^{62} (2 2 2)	α^{59} (0 3 1)	63	3

Зафиксируем порождающий многочлен $P(x)$ примитивного циклического (N, K, D) кода с коэффициентами над $GF(4^3)$. Пусть $N = 4095$, $D = 7$. Для выбора многочлена $P(x)$ рассмотрим конечное поле $GF(64^2)$, построенное по кольцу многочленов по модулю $G(x) = x^2 + x + 3$. Коэффициенты многочлена $G(x)$ – суть элементы поля $GF(4^3)$. В табл. 2.2 представлены первые четыре класса сопряженных элементов поля $GF(64^2)$, порядки элементов и степени минимальных многочленов.

Зададим циклический *(N, K, D)* код порождающим многочленом вида:

$P(x) = НОК(f_1, f_2, f_3, f_4, f_5, f_6) = (x + \alpha^1) \cdot (x + \alpha^{64}) \cdot (x + \alpha^2) \cdot (x + \alpha^{128}) \cdot (x + \alpha^3) \cdot (x + \alpha^{192}) \cdot (x + \alpha^4) \cdot (x + \alpha^{256}) \cdot (x + \alpha^5) \cdot (x + \alpha^{320}) \cdot (x + \alpha^6) \cdot (x + \alpha^{384}) = x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x^1 + 1$. Очевидно, многочлен $P(x)$ задает код БЧХ над $GF(4^3)$ и по теореме БЧХ–кодов имеем: $N = 4095$, $K = 4082$, $D = 7$. Степень многочлена $P(x)$ равна $r = 12$.

Воспользовавшись результатом теоремы 2.1 и леммы 2, получим несистематический сверточный (n, k, d) код над $GF(2^2)$ с параметрами: $m = 3$; $k^0 = 1$; $n^0 = m = 3$; $\nu = r \cdot k^0 = 12$; $k = r + 1 = 13$; $n = (r + 1) \cdot n^0 = k \cdot m = 39$; $R = 1 / 3$; $C(x) = I(x) \cdot P(x)$. Работа такого кодера состоит в следующем. Поток входящих информационных символов разбивается на кадры по одному двухбитному символу. В течение каждого момента времени в регистр сдвига вводится новый информационный символ, вплоть до $r = 12$ символов, которые заполняют весь кодирующий регистр. Кодер, по введенному символу и $r = 12$ хранящимся в нем символам вычисляет три двухбитных символа кодового слова (один символ кодового слова БЧХ кода над $GF(4^3)$). Эти три двухбитных кодовых символа выводятся из кодера, как только следующий информационный символ поступает в него. Следовательно, каждому информационному символу над $GF(2^2)$ соответствуют три кодовых символа над $GF(2^2)$.

Воспользуемся рассмотренным сверточным кодом над $GF(2^2)$ для построения на его основе двоичного кода. Воспользовавшись леммой 3, получим обобщение результата теоремы 2.1: $k_2^0 = 2$; $n_2^0 = 6$; $\nu_2 = 24$; $k_2 = 26$; $n_2 = 78$; $R = 1 / 3$; $d_\infty \geq 7$; $C(x)=I(x)\cdot P(x)$. Работа такого кодера состоит в следующем. Разобьем поток входных информационных символов на подблоки по 2 бита и представим его как элемент поля $GF(2^2)$. В течение каждого момента времени в регистр сдвига вводится новый информационный символ (суть два входных бита), вплоть до $r = 12$ символов, которые заполняют весь кодирующий регистр. Кодер вычисляет три двухбитных символа кодового слова, преобразует их в шестибитовый кадр кодового слова.

Следовательно, каждому информационному кадру по 2 бита соответствует кадр кодовых символов из шести бит.

Рассмотрим множество H элементов поля $GF(4^3)$ вида

$$I_j = z_2 x^2 + z_1 x + z_0,$$

где $z_i \in GF(q)$, причем $z_2 = 0$. Элементы z_1 и z_2 образуют информационный кадр из $k^0 = 2$ символов над $GF(q)$. Определенное таким образом отображение символов $GF(q)$ в символы $GF(q^m)$ является однозначным соответствием. В соответствии с табл. 2.1 множество H содержит следующие элементы: $\{\alpha^{-\infty}, \alpha^1, \alpha^2, \alpha^5, \alpha^{15}, \alpha^{17}, \alpha^{22}, \alpha^{23}, \alpha^{26}, \alpha^{36}, \alpha^{38}, \alpha^{43}, \alpha^{44}, \alpha^{47}, \alpha^{57}, \alpha^{59}\}$, $\log_q \left| H \right| = 2$. Воспользовавшись результатом теоремы 2.2, получим несистематический сверточный (n, k, d) код над $GF(2^2)$ с параметрами: $m = 3$; $k^0 = \log_q \left| H \right| = 2$; $m \geq k^0$; $n^0 = 3$; $\nu = r \cdot k^0 = 24$; $k = (r+1) \cdot k^0 = 26$; $n = k \cdot n^0 / k^0 = 39$; $R = k^0 / m = 2 / 3$; $C(x) = I(x) \cdot P(x)$. Работа такого кодера состоит в следующем. Поток входящих информационных символов разбивается на кадры по два двухбитных символа и однозначно отождествляется одному из элементов группы H. По условию $H \subseteq GF(q^m)$, следовательно, имеет смысл выражение $C(x)=I(x) \cdot P(x)$. В течение каждого момента времени в регистр сдвига вводится новый информационный символ, принадлежащий группе $H \subseteq GF(q^m)$ и являющийся образом двух двухбитных входных символов. Поступающие в кодирующий регистр символы заполняют его вплоть до $r = 12$ символов. Кодер, по введенному символу (суть два двухбитных символа) и $r = 12$ хранящимся в нем символам вычисляет три двухбитных символа кодового слова (один символ кодового слова БЧХ кода над $GF(4^3)$). Эти три двухбитных кодовых символа выводятся из кодера, как только следующий информационный символ поступает в него. Следовательно, каждому информационному символу из группы H – двум двухбитным символам над $GF(2^2)$ соответствуют три кодовых символа над $GF(2^2)$.

Последняя конструкция позволяет также определить двоичный несистематический сверточный код. Воспользуемся результатом леммы 3. Разобьем поток входных информационных

символов на подблоки по 4 бита. Мощность алфавита подблоков равна $2^4 = 16$, мощность группы H равна 21. Следовательно, всегда можно выбрать однозначное сопоставление потока входных 16–ичных символов набору символов группы H. По условию $H \subseteq GF(q^m)$, следовательно, имеет смысл выражение $C(x) = I(x) \cdot P(x)$. Воспользовавшись леммой 4, получим: $k_2^0 = 4$; $n_2^0 = 6$; $\nu_2 = 48$; $k_2 = 52$; $n_2 = 78$; $R = 2 / 3$.

Таким образом, порождающий многочлен $P(x)$ (N, K, D) кода БЧХ над $GF(4^3)$ с параметрами $N = 4095$, $K = 4082$, $D = 7$ в зависимости от способа обработки входных символов однозначно определяет следующие несистематические сверточные коды.

1. Недвоичный сверточный код над $GF(2^2)$ с параметрами: $k^0 = 1$; $n^0 = 3$; $\nu = 12$; $k = 13$; $n = 39$; $R = 1 / 3$;
2. Двоичный сверточный код с параметрами: $k_2^0 = 2$; $n_2^0 = 6$; $\nu_2 = 24$; $k_2 = 26$; $n_2 = 78$; $R = 1 / 3$;
3. Недвоичный сверточный код над $GF(2^2)$ с параметрами: $k^0 = 2$; $n^0 = 3$; $\nu = 24$; $k = 26$; $n = 39$; $R = 2 / 3$;
4. Двоичный сверточный код с параметрами: $k_2^0 = 4$; $n_2^0 = 6$; $\nu_2 = 48$; $k_2 = 52$; $n_2 = 78$; $R = 2 / 3$.

Таким образом, результат теоремы 2.2 и лемм 1–4 позволяет обобщить построение несистематических сверточных кодов произвольной скорости. Для определения минимального расстояния сверточного кода сформулируем и докажем следующую теорему.

Теорема 2.3. Порождающий многочлен степени r (N, K, D) циклического кода над $GF(q^m)$ полностью определяет несистематический сверточный (n, k, d) код над $GF(q)$ с кодовым ограничением $\nu = r \cdot k^0$ и параметрами

$$\begin{cases} n = k \cdot n^0 / k^0, \\ k = (r+1) \cdot k^0, \\ R = k^0 / m, \\ d_\infty \geq D. \end{cases}$$

Доказательство. Согласно теореме 2.2 произвольный многочлен степени r с коэффициентами над $GF(q^m)$ полностью определяет несистематический сверточный (n, k, d) код над $GF(q)$ (см. рис. 2.4) с кодовым ограничением ν, причем $n^0 = m$; $\nu = r \cdot k^0$; $k = (r + 1) \cdot k^0$; $n = k \cdot n^0 / k^0$; $R = k^0 / m$; $m \geq k^0$; $k^0 = log_q \left| H \right|$; $H \subseteq GF(q^m)$.

Если на вход устройства (рис. 2.4) подать K информационных кадров по k^0 q–ичных символов (что эквивалентно подаче K кадров по одному q^{k^0}–ичному символу), то снятая с выхода кодовая последовательность длиной N q^m – ичных символов суть кодовое слово циклического (N, K, D) кода над $GF(q^m)$. Следовательно, два любых кодовых блока, соответствующих двум произвольным входным последовательностям длиной K q^{k^0}–ичных символов, будут отличаться, по крайней мере, в D q^m–ичных символов. Последовательное считывание символов при одинаковых степенях многочленов $F_1(x)$ ... $F_m(x)$ суть отображение элементов поля $GF(q^m)$, в элементы образующего поля $GF(q)$ которое не уменьшает кодовое расстояние между произвольными q–ичными кодовыми словами длины $N \cdot m$. По условию теоремы длина информационного кадра равна k^0, следовательно, $d_K \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_{r+1} \leq d_{r+2} \leq \ldots \leq d_\infty$. Если выполняется условие $K \leq r$, то, очевидно, $d_\infty \geq d \geq d_K$. Если $K > r$, то выполняется лишь неравенство $d_\infty \geq d_K$, что и завершает доказательство.

Пример. Воспользуемся рассмотренным выше примером построения несистематических сверточных кодов. Согласно теореме 2.3 выполняется условие $K > r$ и справедливо неравенство $d_\infty \geq 7$. Таким образом, имеем четыре несистематических сверточных кода:

- $(39, 13)$ сверточный код над $GF(2^2)$ с $\nu = 12$; $R = 1 / 3$, $d_\infty \geq 7$;
- двоичный $(78, 26)$ сверточный код с $\nu = 24$; $R = 1 / 3$, $d_\infty \geq 7$;
- $(39, 26)$ сверточный код над $GF(2^2)$ с $\nu = 24$; $R = 2 / 3$, $d_\infty \geq 7$;
- двоичный $(78, 52)$ сверточный код с $\nu = 48$; $R = 2 / 3$, $d_\infty \geq 7$.

Отметим, что для поиска переборным методом сверточных кодов с $\nu = 48$ необходимо перебрать $2^{\nu} = 2^{48} = 281474976710656$ кодирующих устройств. Для определения кодового расстояния необходимо протестировать $2^{52} = 4503599627370496$ кодовых слов в каждом устройстве, что является практически неразрешимой задачей. Рассмотренные примеры наглядно демонстрируют конструктивность предложенного алгебраического метода построения несистематических сверточных кодов.

В то же время, очевидно, что оценка $d_{\infty} \geq D$ в теоремах 2.1, 2.3 весьма не точна. Основным недостатком рассмотренного подхода построения сверточных кодов является низкая конструктивная величина свободного минимального расстояния. Ниже предлагается подход по предсказанию (прогнозированию) свободного кодового расстояния несистематических сверточных кодов, заданных с помощью порождающего многочлена циклического кода.

Предложение. Предсказанное (прогнозируемое) свободное минимальное расстояние $d_{П}$ несистематического сверточного (n, k, d) – кода над $GF(q)$, алгебраически заданного порождающим многочленом (N, K, D) циклического кода над $GF(q^m)$, определяется выражением

$$d_{П} = \frac{q^m - q^{m-1}}{q^m - 1} \cdot m \cdot D \cdot \qquad (2.11)$$

Вывод выражения (2.11) основан на подсчете ненулевых q–ичных символов в выходной кодовой последовательности несистематического сверточного (n, k, d) кода, алгебраически заданного с помощью порождающего многочлена (N, K, D) циклического кода над $GF(q^m)$.

По теоремам 2.1 – 2.3 несистематический сверточный код эквивалентен ограничению недвоичного циклического кода над $GF(q^m)$ на подполе $GF(q)$, т.е. отображению символов кодовых слов циклического кода над $GF(q^m)$ в символы сверточного кода над $GF(q)$. Мощность множества прообразов равна q^m, а без нулевого символа поля $GF(q^m)$ мощность множества ненулевых прообразов равна $q^m - 1$. Каждому символу над $GF(q^m)$

соответствует m q – ичных символов, т.е. мощность множества образов равна $m \cdot q^m$. Всего ненулевых q–ичных символов в множестве образов равно $m \cdot (q^m - q^{m-1})$. Таким образом, при алгебраически заданном сверточном кодировании множество из $q^m - 1$ ненулевых символов над $GF(q^m)$ отображается в множество из $m \cdot (q^m - q^{m-1})$ ненулевых q–ичных символов. Следовательно, среднее число ненулевых q–ичных символов на выходе несистематического сверточного кода будет определяться как

$$\frac{\left(q^m - q^{m-1}\right) \cdot m}{q^m - 1} \cdot D,$$

где D – минимальное кодовое расстояние (N, K, D) циклического кода над $GF(q^m)$.

Следствие. Для несистематического сверточного (n_*, k_*, d_*) код над $GF(q^{k^0})$ с параметрами: $k_*^0 = 1$; $n_*^0 = m \cdot k_*^0 = m$; $v_* = r \cdot k_*^0 = r$; $k_* = r_* + 1$; $n_* = (r + 1) \cdot n_*^0 = k_* \cdot m_*$; $R = 1 / w$; $d_\infty \geq D$; $C(x) = I(x) \cdot P(x)$ для такого целого w, что $m = w \cdot k^0$ выражение (2.11) запишется в виде

$$d_П = \frac{q^w - q^{w-1}}{q^w - 1} \cdot w \cdot D. \qquad (2.12)$$

Действительно, по лемме 1 порождающий многочлен степени r (N, K, D) циклического кода над $GF(q^m)$ полностью определяет несистематический сверточный (n_*, k_*, d_*) код над $GF(q^{k^0})$ с параметрами: $k_*^0 = 1$; $n_*^0 = m \cdot k_*^0 = m$; $v_* = r \cdot k_*^0 = r$; $k_* = r_* + 1$; $n_* = (r + 1) \cdot n_*^0 = k_* \cdot m_*$; $R = 1 / w$; $d_\infty \geq D$; $C(x) = I(x) \cdot P(x)$ для такого целого w, что $m = w \cdot k^0$. По сути, такой код является ограничением (N, K, D) циклического кода над $GF(q^m)$ на поле $GF(q^w)$, т.е. отображением символов кодовых слов циклического кода над $GF(q^m)$ в символы сверточного кода над $GF(q^w)$. Проведя аналогичные рассуждения, получим искомое выражение (2.12).

По аналогии с рассуждениями для вывода выражений (2.11) и (2.11) для леммы 4 прогнозируемое свободное минимальное расстояние $d_П$ будет определяться выражением (2.11) после подстановки значения m* вместо m.

Пример. Воспользуемся рассмотренным выше примером построения несистематических сверточных кодов, алгебраически заданных с помощью порождающего многочлена *P(x) (N, K, D)* кода БЧХ над $GF(4^3)$ с параметрами $N = 4095$, $K = 4082$, $D = 7$. Рассчитаем прогнозируемое свободное кодовое расстояние $d_П$ несистематического сверточного *(n, k, d)* кода над $GF(2^2)$ с параметрами: $k^0 = 1$; $n^0 = 3$; $\nu = 12$; $k = 13$; $n = 39$; $R = 1 / 3$; $d_\infty \geq 7$ (случай 1). Подставив в выражение (2.9) параметры кода, получим

$$d_{П_1} = \frac{q^m - q^{m-1}}{q^m - 1} \cdot m \cdot D = \frac{4^3 - 4^2}{4^3 - 1} \cdot 3 \cdot 7 = 16.$$

Аналогично для двоичного *(78, 26)* сверточного кода с $\nu = 24$, $R = 1 / 3$, $d_\infty \geq 7$ (случай 2):

$$d_{П_2} = \frac{q^m - q^{m-1}}{q^m - 1} \cdot m \cdot D = \frac{2^6 - 2^5}{2^6 - 1} \cdot 6 \cdot 7 \approx 21.$$

Для *(39, 26)* сверточного кода над $GF(2^2)$ с $\nu = 24$, $R = 2 / 3$, $d_\infty \geq 7$ (случай 3):

$$d_{П_3} = \frac{q^m - q^{m-1}}{q^m - 1} \cdot m \cdot D = \frac{4^3 - 4^2}{4^3 - 1} \cdot 3 \cdot 7 = 16.$$

Для двоичного *(78, 52)* сверточного кода с $\nu = 48$, $R = 2 / 3$, $d_\infty \geq 7$ (случай 4):

$$d_{П_4} = \frac{q^m - q^{m-1}}{q^m - 1} \cdot m \cdot D = \frac{2^6 - 2^5}{2^6 - 1} \cdot 6 \cdot 7 \approx 21.$$

Рассмотрим еще один *пример* построения несистематического сверточного кода через порождающий многочлен кода Рида–Соломона (РС). Зафиксируем, как и в предыдущем примере, конечные поля $GF(2^2)$ и $GF(4^3)$. Зададим *(N, K, D)* код РС над $GF(4^3)$ через проверочный многочлен *P(x)* вида

$$P(x) = (x - \alpha^i) \cdot (x - \alpha^{2i}) \cdot \ldots \cdot (x - \alpha^{2ti}),$$

где t – число ошибок, которые должен исправлять *(N, K, D)* код РС, $N = q^m - 1$; $r = deg P(x)$; $K = N - r$; $D = 2t + 1$; $\alpha^i \in GF(q^m)$.

Пусть $D = 7$, $i = 1$. Тогда $P(x) = (x - \alpha^1) \cdot (x - \alpha^2) \cdot (x - \alpha^3) \cdot (x - \alpha^4) \cdot (x - \alpha^5) \cdot (x - \alpha^6) = x^6 + x^5 + x^4 + x^3 + x^2 + x + a$, $N = 63$, $r = 6$, $K = 57$.

По аналогии с рассмотренным выше примером определим несистематические сверточные коды с следующими параметрами.

1. Недвоичный сверточный код над $GF(2^2)$ с параметрами: $k^0 = 1$; $n^0 = 3$; $\nu = 6$; $k = 7$; $n = 21$; $R = 1 / 3$; $d_\infty \geq 7$; $d_{П_1} = 16$;

2. Двоичный сверточный код с параметрами: $k_2^0 = 2$; $n_2^0 = 6$; $\nu_2 = 12$; $k_2 = 14$; $n_2 = 42$; $R = 1 / 3$; $d_\infty \geq 7$; $d_{П_2} = 21$;

3. Недвоичный сверточный код над $GF(2^2)$ с параметрами: $k^0 = 2$; $n^0 = 3$; $\nu = 12$; $k = 14$; $n = 21$; $R = 2 / 3$; $d_\infty \geq 7$; $d_{П_3} = 16$;

4. Двоичный сверточный код (n, k, d) код над $GF(2)$ с параметрами: $k_2^0 = 4$; $n_2^0 = 6$; $\nu_2 = 24$; $k_2 = 28$; $n_2 = 42$; $R = 2 / 3$; $d_\infty \geq 7$; $d_{П_4} = 21$.

В результате проведенного уточнения свободного минимального расстояния рассмотренных кодов получены следующие значения: $d_{\infty_1} = 23$, $d_{\infty_2} = 35$, $d_{\infty_3} = 21$, $d_{\infty_4} = 31$, что позволяет отнести их к лучшим известным сверточным кодам.

Рассмотренные выше примеры построения несистематических сверточных кодов через порождающий многочлен (N, K, D) циклического кода над $GF(q^m)$ удовлетворяют условию $d_\infty \geq d_П$, где d_∞ – истинное свободное минимальное расстояние, полученное уточнением кодового расстояния. Отметим также, что при тестировании (уточнении) кодового расстояния несистематических сверточных кодов автором не было получено ни одного случая с $d_\infty < d_П$. Это говорит о конструктивности предложенного способа предсказания свободного минимального расстояния несистематических сверточных кодов.

В следующем подразделе разрабатываются практические алгоритмы построения несистематических сверточных кодов с заданными конструктивными характеристиками.

2.3. Разработка алгоритмов построения несистематических сверточных кодов с заданными конструктивными характеристиками

Практическое использование результатов доказанных теорем 2.1 – 2.3 позволяет алгебраически задавать несистематический сверточный код порождающим многочленом циклического кода. Конструктивные характеристики сверточного *(n, k, d)* кода над *GF(q)* связаны с параметрами образующего циклического *(N, K, D)* кода над $GF(q^m)$ с порождающим многочленом степени *r*:

$k^0 = \log_q \left/ H \right/;$
$n^0 = m;$
$\nu = r \cdot k^0;$
$k = (r + 1) \cdot k^0;$
$n = k \cdot n^0 / k^0;$
$d_\infty \geq D;$
$R = k^0 / m,\ m \geq k^0;$

где $H \subseteq GF(q^m)$.

Алгоритм построения сверточного *(n, k, d)* кода над *GF(q)* определим в виде последовательности следующих шагов.

ШАГ 1. Выбор конструктивных параметров сверточного *(n, k, d)* кода над *GF(q)*.

ШАГ 2. Расчет параметров образующего поля $GF(q^m)$. Выбор циклического кода, расчет его конструктивных *(N, K, D)* параметров над $GF(q^m)$.

ШАГ 3. Выбор порождающего многочлена циклического *(N, K, D)* кода $GF(q^m)$. Расчет прогнозируемого свободного расстояния сверточного кода.

ШАГ 4. Определение порождающих многочленов несистематического сверточного *(n, k, d)* кода, построение схемы кодера.

ШАГ 5. Уточнение минимального кодового расстояния и свободного кодового расстояния несистематического сверточного *(n, k, d)* кода (при необходимости).

Рассмотрим выполнение предложенного алгоритма более подробно.

После ввода конструктивных параметров сверточного кода над $GF(q)$ – параметров ν, n^0, k^0 и q на втором шаге выполняется расчет параметров образующего поля $GF(q^m)$, осуществляется выбор циклического кода и расчет его конструктивных (N, K, D) параметров над $GF(q^m)$. Для этого выражения связывающие параметры сверточного и циклических кодов перепишем в виде

$$R = k^0 / n^0;$$
$$m = n^0;$$
$$r = \nu / k^0;$$
$$D \le d_\infty.$$

После расчета параметров образующего поля $GF(q^m)$ необходимо выбрать циклический код, порождающий многочлен которого будет задавать сверточный код.

Рассмотрим случай, когда в качестве циклического кода выбран примитивный код БЧХ. Для расчета его конструктивных (N, K, D) параметров зафиксируем двучлен $(x^M - 1)$ так, что конструктивная длина примитивного кода БЧХ равна

$$N = (q^m)^M - 1.$$

Далее, определив степень r порождающего многочлена примитивного кода БЧХ, рассмотрим поле разложения двучлена $(x^M - 1)$ на минимальные многочлены элементов поля $GF((q^m)^M)$ над $GF(q^m)$. Порождающий многочлен примитивного кода БЧХ задается в виде

$$P(x) = НОК(f_1, f_2, \ldots, f_{2t}),$$

где t – число ошибок, которые должен исправлять циклический (N, K, D) код, $N = (q^m)^M - 1$, $r = degP(x)$, $K = N - r$, $D = 2t + 1$, f_i – минимальные многочлены над $GF(q^m)$ элементов $\alpha^i \in GF((q^m)^M)$. После расчета $d_П$ третий шаг алгоритма для примитивных кодов БЧХ завершен.

Рассмотрим случай, когда в качестве циклического кода выбран непримитивный код БЧХ. По определению, длина непримитивного кода БЧХ равна одному из сомножителей в разложении числа $(q^m)^M - 1$ (если, конечно, число $(q^m)^M - 1$ не является простым), т.е.

$$N = ((q^m)^M - 1)/g$$

для произвольного целого g, делящего нацело число $(q^m)^M - 1$. Очевидно, что должно выполняться также условие $r < N$.

Порождающий многочлен непримитивного кода БЧХ задается в виде

$$P(x) = НОК(\varphi_1, \varphi_2, \ldots, \varphi_{2t}),$$

где t – число ошибок, которые должен исправлять циклический (N, K, D) код, $N = ((q^m)^M - 1)/g$; $r = degP(x)$; $K = N - r$; $D = 2t + 1$, φ_i – минимальные многочлены над $GF(q^m)$ элементов $\beta^i \in GF((q^m)^M)$ такие, что их порядок равен N, т.е. $\beta^i = \alpha^{jg}$, $j = 1, 2, \ldots, M/2$. После расчета $d_П$ третий шаг алгоритма для непримитивных кодов БЧХ завершен.

Рассмотрим случай, когда в качестве циклического кода выбран код РС. По определению, порождающий многочлен кода РС задается в виде

$$P(x) = (x - \alpha^i) \cdot (x - \alpha^{2i}) \cdot \ldots \cdot (x - \alpha^{2ti}),$$

где t – число ошибок, которые должен исправлять (N, K, D) код РС, $N = q^m - 1$; $r = degP(x)$; $K = N - r$; $D = 2t + 1$; $\alpha^i \in GF(q^m)$. После вычисления (N, K, D) параметров кода РС, выбора порождающего многочлена и расчета $d_П$ третий шаг алгоритма для рассмотренного случая завершен.

На четвертом шаге предложенного алгоритма построения несистематических сверточных кодов производится определение порождающих многочленов сверточного кода над $GF(q)$, строится схема кодера. Если порождающий многочлен $P(x)$ циклического (N, K, D) кода над $GF(q^m)$

$$P(x)=\alpha_{r-1}x^{r-1}+\alpha_{r-2}x^{r-2}+...+\alpha_1 x+\alpha_0,\ \alpha_i \in GF(q^m)$$

записать в виде

$$P(x)=(p_{1,r-1},p_{2,r-1},...,p_{m,r-1})x^{r-1}+(p_{1,r-2},p_{2,r-2},...,p_{m,r-2})x^{r-2}+...+$$
$$+(p_{1,1},p_{2,1},...,p_{m,1})x+(p_{1,0},p_{2,0},...,p_{m,0}),\ p_{i,j}\in GF(q),$$

то многочлены

$$P_1(x)=p_{1,r-1}x^{r-1}+p_{1,r-2}x^{r-2}+...+p_{1,1}x+p_{1,0};$$

$$P_2(x)=p_{2,r-1}x^{r-1}+p_{2,r-2}x^{r-2}+...+p_{2,1}x+p_{2,0};$$

...

$$P_m(x)=p_{m,r-1}x^{r-1}+p_{m,r-2}x^{r-2}+...+p_{m,1}x+p_{m,0}$$

будут являться порождающими многочленами искомого несистематического сверточного кода. Схема алгоритма формирования порождающих многочленов несистематического сверточного кода представлена на рис. 2.5.

Подставим в общую схему несистематического сверточного кодера (см. рис. 2.4) параметры полученных многочленов $P_1(x)$ ... $P_m(x)$. Коэффициенты многочленов $P_1(x)$ ... $P_m(x)$ однозначно определяют кодирующие регистры с обратными связями, т.е. однозначно задают схему кодера искомого сверточного (n, k, d) кода.

На пятом шаге разработанного алгоритма построения сверточных кодов путем тестирования производится уточнение кодового расстояния (при необходимости).

На рис. 2.6 представлена общая схема алгоритма алгебраического построения несистематического сверточного кода с использованием разработанного метода.

Проведем исследования свойств несистематических сверточных кодов, заданных порождающими многочленами циклических кодов.

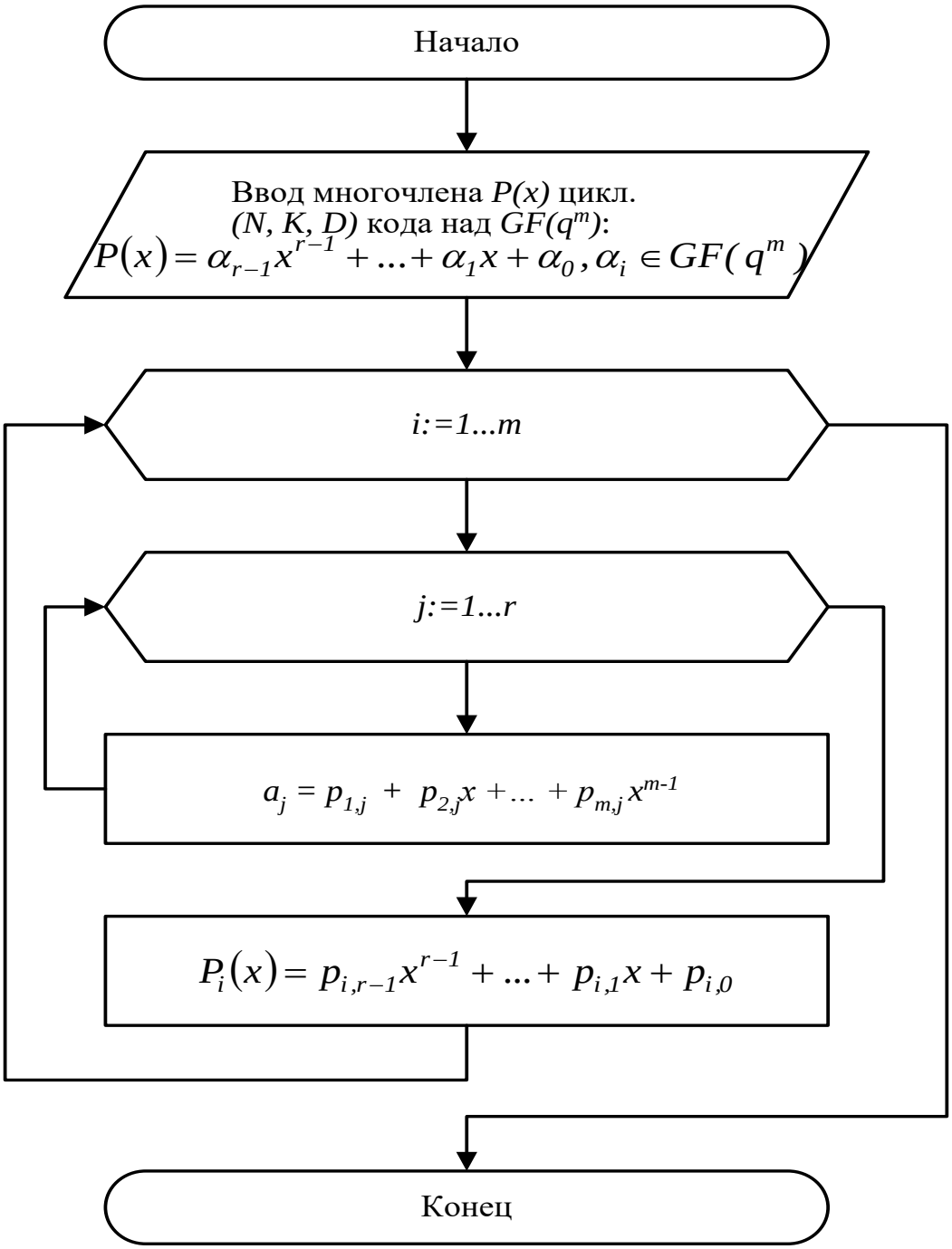


Рис. 2.5. Схема алгоритма формирования порождающих многочленов несистематического сверточного кода

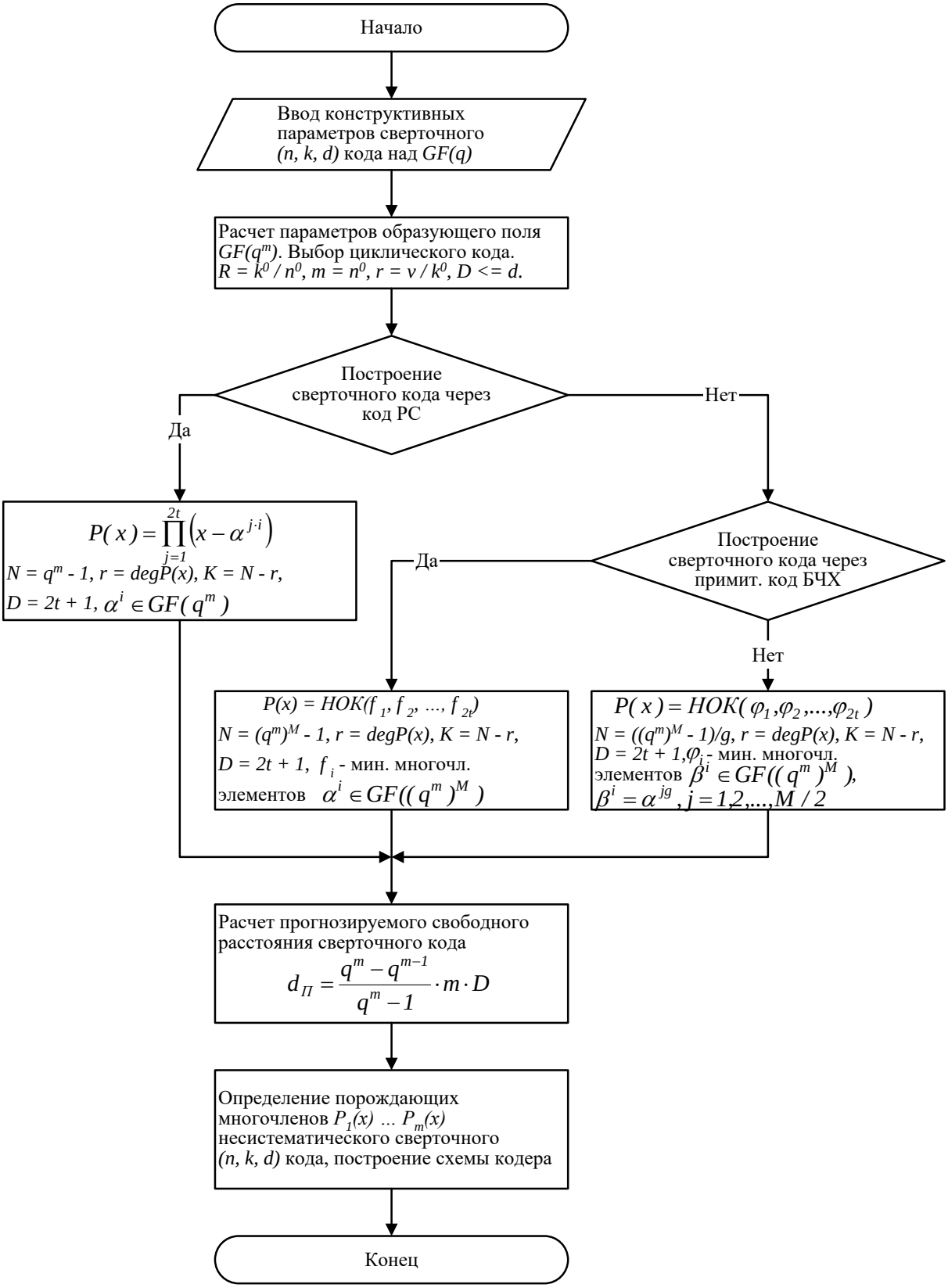


Рис. 2.6. Схема алгоритма алгебраического построения несистематического сверточного кода

2.4. Исследование свойств несистематических сверточных кодов, заданных порождающим многочленом циклического кода

Разработанный метод построения несистематических сверточных кодов позволяет алгебраически определить сверточный *(n, k, d)* код через порождающий многочлен циклического кода. Проведем исследования свойств сверточных кодов в зависимости от выбранного циклического кода.

Зафиксируем циклический *(N, K, D)* код над $GF(q^m)$ через его порождающий многочлен *P(x)* степени *r*. Параметры сверточного *(n, k, d)* кода над *GF(q)* запишутся в виде: $n^0 = m$; $k^0 = \log_q |H|$; $H \subseteq GF(q^m)$; $\nu = r \cdot k^0$; $k = (r + 1) \cdot k^0$; $n = k \cdot n^0 / k^0$; $d_\infty \geq D$; $R = k^0 / m$, $m \geq k^0$. Лучшим будет такой сверточный *(n, k, d)* код, который при меньших конструктивных характеристиках *n* и *k* обеспечит большие значения *d* и/или d_∞. Проанализируем потенциальные возможности примитивных и непримитивных кодов БЧХ, кодов РС для построения хороших сверточных кодов.

Определяющим в выражениях по расчету конструктивных параметров *n* и *k* сверточного кода является показатель *r* – длина регистра сдвига в схеме сверточного кодирования (рис. 2.1). Если несистематический сверточный *(n, k, d)* код задан через порождающий многочлен *P(x)* циклического *(N, K, D)* кода над $GF(q^m)$, то показатель *r* соответствует степени порождающего многочлена $r = degP(x)$, а задача построения хорошего сверточного кода сводится к выбору такого циклического кода, который при минимальном *r* обеспечивал максимальное значение *D*. В алгебраической теории блоковых кодов известно следующее ограничение:

$$D \leq N - K + 1 = r + 1, \qquad (2.11)$$

которое носит название границы Синлгтона линейного *(N, K, D)* кода. Применительно к алгебраическому построению сверточного кода выражение (2.11) дает верхнюю оценку *d* и/или d_∞ для фиксированного показателя *r*.

Предложенный алгоритм (рис. 2.6) алгебраического построения несистематического сверточного *(n, k, d)* кода оперирует с кодами БЧХ и РС. Для примитивного кода БЧХ показатель $r = degP(x)$ определяется как сумма степеней минимальных многочленов элементов $\alpha^i \in GF((q^m)^M)$, образующих непрерывную цепочку длиной не менее t, $D = 2t + 1$. Для непримитивных кодов БЧХ показатель $r = degP(x)$ определяется как сумма степеней минимальных многочленов элементов $\beta^i \in GF((q^m)^M)$ таких, что их порядок равен N, т.е. $\beta^i = \alpha^{jg}$; $j = 1, 2, \ldots, M/2$; $N = ((q^m)^M - 1)/g$. В обоих случаях для кодов БЧХ показатель $r = degP(x)$ не удовлетворяет границе (2.11), следовательно, не обеспечивает высокое значение d и/или d_∞ алгебраически заданного сверточного *(n, k, d)* кода.

Важным классом циклических кодов являются коды РС. Коды РС – такие коды БЧХ, у которых мультипликативный порядок алфавита символов кодового слова делится на длину кода. Параметры *(N, K, D)* кода РС над $GF(q^m)$ связаны следующим соотношением:

$$N = q^m - 1,\ N - K = D - 1.$$

Следовательно, коды РС являются оптимальными в смысле границы Синлгтона и являются кодами с максимально достижимым кодовым расстоянием. Практически это означает, что при фиксированных N и K не существует кода, у которого минимальное расстояние D больше, чем у кода РС. Таким образом, несистематический сверточный *(n, k, d)* код, заданный порождающим многочленом кода РС, будет обладать лучшими конструктивными характеристиками по сравнению с остальными циклическими кодами.

Проведем оценку конструктивных параметров сверточных *(n, k, d)* кодов, алгебраически заданных порождающим многочленом кода РС.

Зафиксируем конечное поле $GF(2^2)$ и рассмотрим коды РС с параметрами: $N = 2^2 - 1 = 3$, $3 - K = D - 1$. В табл. 2.2 представлены параметры кодов РС над $GF(2^2)$, конструктивные параметры сверточных *(n, k, d)* кодов, алгебраически заданных порождающим многочленом кода РС, предсказанное и истинное

значение свободного кодового расстояния. Случаи для кодов *(n, 1, n)* соответствуют тривиальному коду с повтором символов.

Таблица 2.2.

Конструктивные характеристики двоичных сверточных кодов, заданных через порождающий многочлен кода РС над $GF(2^2)$

(N, K, D)	*(n, k, d)*	*v*	*R*	$d_П$	d_∞
(3, 1, 3)	(6, 3, 3)	2	1 / 2	4	5
(3, 2, 2)	(4, 2, 3)	1	1 / 2	2,7	3

Зафиксируем конечное поле $GF(2^3)$ и рассмотрим коды РС с параметрами $N = 2^3 - 1 = 7$; $7 - K = D - 1$. В табл. 2.3 представлены параметры кодов РС над $GF(2^3)$, конструктивные параметры сверточных *(n, k, d)* кодов, алгебраически заданных порождающим многочленом кода РС, предсказанное значение свободного кодового расстояния и истинное кодовое расстояние полученных сверточных кодов.

Таблица 2.3

Конструктивные характеристики двоичных сверточных кодов, заданных через порождающий многочлен кода РС над $GF(2^3)$

(N, K, D)	*(n, k, d)*	*v*	*R*	$d_П$	d_∞
(7, 1, 7)	(21, 7, 7)	6	1 / 3	12	15
	(21, 14, 7)	12	2 / 3	12	13
(7, 2, 6)	(18, 6, 6)	5	1 / 3	10,3	
	(18, 12, 6)	10	2 / 3	10,3	
(7, 3, 5)	(15, 5, 5)	4	1 / 3	8,6	
	(15, 10, 5)	8	2 / 3	8,6	
(7, 4, 4)	(12, 4, 4)	3	1 / 3	6,9	
	(12, 8, 4)	6	2 / 3	6,9	
(7, 5, 3)	(9, 3, 3)	2	1 / 3	5,1	
	(9, 6, 3)	4	2 / 3	5,1	
(7, 6, 2)	(6, 2, 2)	1	1 / 3	3,4	
	(6, 4, 2)	2	2 / 3	3,4	

Зафиксируем конечное поле $GF(2^4)$ и рассмотрим коды РС с параметрами $N = 2^4 - 1 = 15$; $15 - K = D - 1$. В табл. 2.4 представлены параметры кодов РС над $GF(2^4)$, конструктивные параметры сверточных (n, k, d) кодов, $d \geq D$, алгебраически заданных порождающим многочленом кода РС, предсказанное значение свободного кодового расстояния и истинное кодовое расстояние полученных сверточных кодов.

Таблица 2.4.

Конструктивные характеристики двоичных сверточных кодов, заданных через порождающий многочлен кода РС над $GF(2^4)$

(N, K, D)	(n, k, d)	ν	R	$d_П$	d_∞
(15, 1, 15)	(60, 15, 15)	14	1 / 4	32	
	(60, 30, 15)	28	1 / 2	32	
	(60, 45, 15)	42	3 / 4	32	
(15, 2, 14)	(56, 14, 14)	13	1 / 4	29,9	
	(56, 28, 14)	26	1 / 2	29,9	
	(56, 42, 14)	39	3 / 4	29,9	
(15, 3, 13)	(52, 13, 13)	12	1 / 4	27,8	
	(52, 26, 13)	24	1 / 2	27,8	
	(52, 39, 13)	36	3 / 4	27,8	
(15, 4, 12)	(48, 12, 12)	11	1 / 4	25,6	
	(48, 24, 12)	22	1 / 2	25,6	
	(48, 36, 12)	33	3 / 4	25,6	
(15, 5, 11)	(44, 11, 11)	10	1 / 4	23,5	
	(44, 22, 11)	20	1 / 2	23,5	
	(44, 33, 11)	30	3 / 4	23,5	
(15, 6, 10)	(40, 10, 10)	9	1 / 4	21,3	
	(40, 20, 10)	18	1 / 2	21,3	
	(40, 30, 10)	27	3 / 4	21,3	

Продолжение табл. 2.4

(N, K, D)	*(n, k, d)*	*v*	*R*	$d_П$	d_∞
(15, 7, 9)	(36, 9, 9)	8	1 / 4	19,2	
	(36, 18, 9)	16	1 / 2	19,2	
	(36, 27, 9)	24	3 / 4	19,2	
(15, 8, 8)	(32, 8, 8)	7	1 / 4	17,1	
	(32, 16, 8)	14	1 / 2	17,1	
	(32, 24, 8)	21	3 / 4	17,1	
(15, 9, 7)	(28, 7, 7)	6	1 / 4	14,9	
	(28, 14, 7)	12	1 / 2	14,9	
	(28, 21, 7)	18	3 / 4	14,9	
(15, 10, 6)	(24, 6, 6)	5	1 / 4	12,8	
	(24, 12, 6)	10	1 / 2	12,8	
	(24, 18, 6)	15	3 / 4	12,8	
(15, 11, 5)	(20, 5, 5)	4	1 / 4	10,7	
	(20, 10, 5)	8	1 / 2	10,7	
	(20, 15, 5)	12	3 / 4	10,7	
(15, 12, 4)	(16, 4, 4)	3	1 / 4	8,5	
	(16, 8, 4)	6	1 / 2	8,5	
	(16, 12, 4)	9	3 / 4	8,5	
(15, 13, 3)	(12, 3, 3)	2	1 / 4	6,4	
	(12, 6, 3)	4	1 / 2	6,4	
	(12, 9, 3)	6	3 / 4	6,4	
(15, 14, 2)	(8, 2, 2)	1	1 / 4	4,3	
	(8, 4, 2)	2	1 / 2	4,3	
	(8, 6, 2)	3	3 / 4	4,3	

Выводы

1. Проведенные исследования показали, что порождающий многочлен недвоичного циклического кода над $GF(q^m)$ однозначно задает порождающие многочлены несистематического сверточного кода над $GF(q)$. Параметры сверточного (n, k, d) кода определяются свойствами недвоичного циклического (N, K, D) кода.

2. В результате доказанных теорем 2.1–2.3 получены аналитические соотношения, которые связывают параметры сверточного кода с кодовыми характеристиками недвоичного циклического кода и позволяют алгебраически определять коды с наперед заданными конструктивными свойствами.

3. Предложен метод алгебраического построения несистематических сверточных кодов над $GF(q)$, который отличается от известных представлением порождающих многочленов сверточного кода через порождающий многочлен недвоичного циклического кода над $GF(q^m)$, ограниченного на произвольное множество $H \subseteq GF(q^m)$, что позволяет строить коды с $R = k^0 / m$ и снять основное ограничение по скорости кодирования.

4. Разработанные алгоритмы формирования порождающих многочленов сверточных кодов и алгоритмы построения несистематических сверточных кодов позволяют за конечное число шагов однозначно определить правило сверточного кодирования и построить схему кодера с заданными конструктивными характеристиками.

5. Проведенные исследования свойств алгебраически заданных несистематических сверточных кодов, построенных с использованием разработанного метода, показали, что полученные коды являются одними из лучших известных сверточных кодов и близки по своим кодовым характеристикам к оптимальным.

РАЗДЕЛ 3
РАЗРАБОТКА И ИССЛЕДОВАНИЕ АЛГЕБРАИЧЕСКИХ МЕТОДОВ ПОСТРОЕНИЯ РЕКУРСИВНЫХ СВЕРТОЧНЫХ КОДОВ

Предлагается алгебраический метод построения рекурсивных сверточных кодов в систематическом и несистематическом виде. Разрабатываются алгоритмы построения рекурсивных сверточных кодов, заданных через порождающий и/или проверочный многочлен циклического кода. Исследуются свойства алгебраически заданных сверточных кодов, оцениваются их конструктивные параметры.

3.1. Разработка алгебраических рекурсивных сверточных кодов в несистематическом виде

При построении алгебраических рекурсивных сверточных кодов в несистематическом виде воспользуемся известными свойствами цикличных кодов.

Каждый линейный (n, k, d) код над $GF(q)$ является подпространством $GF^k(q)$ пространства $GF^n(q)$. Циклический код является частным случаем подпространства, т.к. обладает дополнительным свойством цикличности. Каждый вектор из $GF^n(q)$ представим многочленом от формальной переменной x степени не выше $n-1$. Компоненты вектора отождествим коэффициентам этого многочлена. Множество многочленов обладает структурой векторного пространства, идентичной структуре пространства $GF^n(q)$, а также структурой кольца многочленов $GF(q)[x]/(x^n - 1)$.

В кольце многочленов их умножение определяется

$$p_1(x) \cdot p_2(x) = R_{x^n-1}\left[p_1(x) \cdot p_2(x)\right],$$

а циклический сдвиг записывается в виде выражения

$$x \cdot p(x) = R_{x^n-1}\left[x \cdot p(x)\right].$$

Если кодовые слова *(n, k, d)* кода над *GF(q)* задаются в виде многочленов, то код является подмножеством кольца $GF(q)[x]/(x^n - 1)$. Код является циклическим, если вместе с кодовым словом *c(x)* он содержит также многочлен $x \cdot c(x)$.

Любой циклический код можно задать через порождающий многочлен *g(x)*, что доказывает следующая теорема.

Теорема 3.1. Единственный приведенный ненулевой многочлен *g(x)* наименьшей степени $r = n - k$ однозначно задает *(n, k, d)* циклический код над *GF(q)* и обозначается порождающим многочленом

$$g(x) = \prod_i (x - \beta^i),$$

где $\beta^i \in GF(q^m)$.

Теорема 3.1. Определяет механизм построения циклических кодов. Как показано выше, его использование позволяет эффективно реализовать процедуру кодирования на основе применения нерекурсивных цифровых фильтров. В то же время циклический код можно однозначно задать другим многочленом – мультипликативно обратным многочлену *g(x)*. При этом справедлива лемма.

Лемма 3.1. Единственный многочлен *h(x)* (проверочный многочлен), мультипликативно обратный приведенному ненулевому многочлену *g(x)*, однозначно задает *(n, k, d)* циклический код над *GF(q)* и обозначается проверочным многочленом, при этом, если

$$g(x) = \prod_i (x - \beta^i), \tag{3.1}$$

то

$$h(x) = \prod_j (x - \beta^j), \tag{3.2}$$

где $\beta^i, \beta^j \in GF(q^m), j \neq i$.

Доказательство. Многочлен $g(x)$ делит многочлен $x^n - 1$, который, в свою очередь, делит многочлен $x^m - 1$, так что $g(x)$ делит также $x^{q^m-1}-1$. Допустим α – примитивный элемент поля $GF(q^m)$, пусть $q^m - 1 = n \cdot b$, и пусть $\beta = \alpha^b$. Тогда все корни многочлена $x^n - 1$, как и корни многочлена $g(x)$, исчерпываются степенями элемента β. Простые делители многочлена $x^n - 1$ имеют своими корнями только такие элементы. Следовательно, в кольце многочленов $GF(q)[x]/(x^n - 1)$ существует некоторый многочлен $h(x)$, являющийся сомножителем $g(x)$ в разложении двучлена $x^n - 1$, т.е. существует многочлен $h(x)$ – делитель $x^n - 1$, и корни многочлена $h(x)$ так же исчерпываются степенями элемента β. Это означает, что произвольный циклический код можно однозначно задать либо порождающим многочленом $g(x)$, либо мультипликативно обратным ему в кольце $GF(q)[x]/(x^n - 1)$ многочленом $h(x)$, причем, если

$$g(x) = \prod_i (x - \beta^i),$$

то

$$h(x) = \prod_j (x - \beta^j),$$

где $\beta^i, \beta^j \in GF(q^m), j \neq i$.

Следствие леммы 3.1. $\deg h(x) = n - \deg g(x) = n - r = k$.

Целесообразно воспользоваться понятием проверочного многочлена для построения правила циклического кодирования в несистематическом виде. Кодовое слово несистематического циклического кода можно представить в виде

$$C(x) = I(x) \cdot g(x).$$

Выразим порождающий многочлен $g(x)$ через проверочный многочлен $h(x)$ и двучлен $x^n - 1$:

$$g(x) = (x^n - 1)/h(x) = 1/h(x)$$

с операцией деления в кольце многочленов $GF(q)[x]/(x^n - 1)$.

После подстановки получим

$$C(x) = I(x)/h(x). \qquad (3.3)$$

Для реализации процедуры деления на многочлен воспользуемся цифровым фильтром с бесконечным импульсным откликом (рекурсивным фильтром). На рис. 3.1 приведена структурная схема цифрового рекурсивного фильтра.

Если на вход цифрового рекурсивного фильтра подать последовательность символов $\{i_k, \ldots, i_1, , i_0\}$, то считанная с выхода последовательность $\{\ldots, c_1, , c_0\}$ удовлетворяет свойству рекурсии:

$$c_j = -\sum_{i=1}^{L} h_i i_{j-i} + i_j \cdot$$

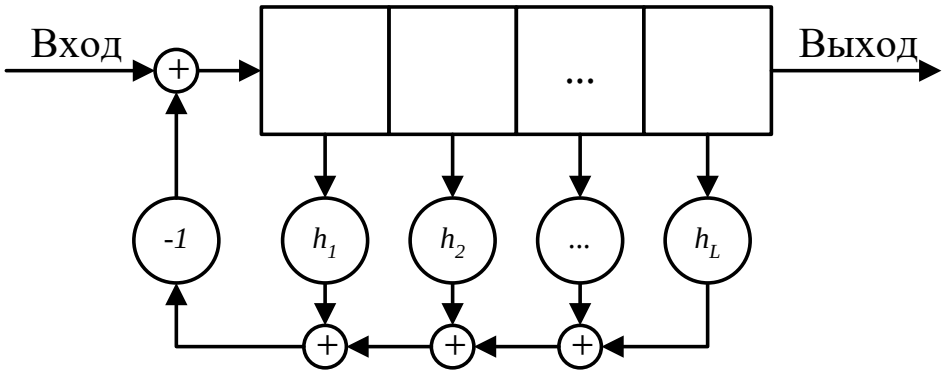


Рис. 3.1. Структурная схема цифрового рекурсивного фильтра

Зададим проверочный многочлен в виде

$$h(x) = h_0 + h_1x + h_2x^2 + \ldots + h_kx^k. \qquad (3.4)$$

Тогда структурную схему несистематического кодера, реализующего правило кодирования (3.3), представим в виде схемы на рис. 3.2.

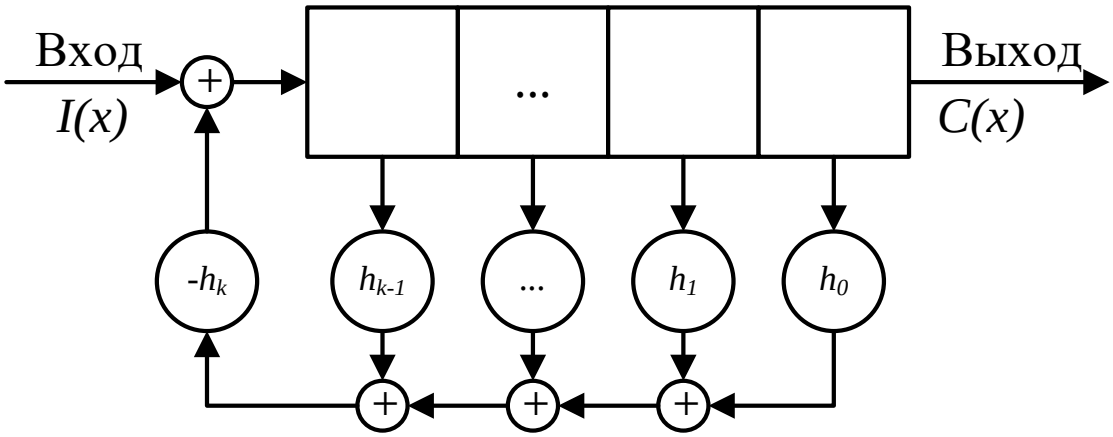


Рис. 3.2. Структурная схема несистематического кодера циклического кода, построенного с использованием проверочного многочлена

Для построения алгебраических рекурсивных сверточных кодов в несистематическом виде воспользуемся выражением (3.3). Рассмотрим процедуру сверточного кодирования с $R = 1/m$. Для построения рекурсивного кодера используем рекурсивный фильтр (рис. 3.2). Если на вход устройства подать информационный многочлен (3.2), в общем случае бесконечной длины, то выходную последовательность с символами из $GF(q^m)$ отобразим в последовательность символов из $GF(q)$. Справедлива следующая теорема.

Теорема 3.2. Несистематический сверточный код над $GF(q)$ с $R = 1/m$ однозначно задается многочленом $h(x)$ над $GF(q^m)$ вида $h(x) = h_0 + h_1x + h_2x^2 + \ldots + h_kx^k$. Если многочлен $h(x)$ – проверочный многочлен недвоичного (N, K, D) циклического кода над $GF(q^m)$, то он однозначно определяет (n, k) несистематический рекурсивный сверточный код над $GF(q)$ с правилом кодирования $C(x)=I(x)/h(x)$ с длиной кодового ограничения

$$\nu = K$$

и конструктивными параметрами

$$\begin{cases} k^0 = 1, \\ n^0 = m, \\ k = K + 1, \\ n = (K+1) \cdot n^0 = k \cdot m, \\ R = 1/m, \\ d_\infty \geq D. \end{cases} \qquad (3.5)$$

Доказательство. Циклический *(N, K, D)* код над $GF(q^m)$ с проверочным многочленом *h(x)* степени *K* однозначно определяет набор регистров сдвига, соединенных связями (рис. 3.2), и задает рекуррентное правило кодирования

$$C(x) = I(x)/h(x).$$

Если на вход устройства подать последовательность символов из *GF(q)*, то считанная с выхода кодовая последовательность длиной *N* q^m – ичных символов является кодовым словом циклического *(N, K, D)*–кода над $GF(q^m)$, а рекурсивный сверточный код является обобщением исходного цикличного кода на непрерывный случай. Параметры сверточного кода связаны соотношениями

$\nu = K \cdot k^0 = r$; $k^0 = 1$; $n^0 = m$; $k = K + 1$; $n = (K + 1) \cdot n^0 = k \cdot m$; $R = 1 / m$

и два любых кодовых слова будут отличаться, по крайней мере, в *D* q^m – ичных символов. Отображение элементов поля $GF(q^m)$ в элементы поля *GF(q)* не уменьшает кодовое расстояние между произвольными *q* – ичными кодовыми словами, следовательно, $d_K \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_{K+1} \leq d_{K+2} \leq \ldots \leq d_\infty$, откуда $d_\infty \geq d_K$, что и завершает доказательство.

Для рассмотрения процедуры сверточного кодирования с $R = k^0 / m$ сформулируем и докажем следующую теорему.

Теорема 3.3. Если зафиксировать конечное множество *H* элементов поля $GF(q^m)$, причем $\log_q |H| = k^0$, $m \geq k^0$, то проверочный многочлен циклического *(N, K, D)* кода над $GF(q^m)$

полностью определяет несистематический рекурсивный сверточный *(n, k, d)* код над *GF(q)* с информационным кадром длины k^0, длиной кодового ограничения

$$\nu = K \cdot k^0$$

и параметрами

$$\begin{cases} n^0 = m, \\ k = (K+1) \cdot k^0, \\ n = (K+1) \cdot n^0, \\ R = k^0 / m, \\ d_\infty \geq D. \end{cases} \qquad (3.6)$$

Доказательство. Представим информационную последовательность в виде многочлена с коэффициентами над *H*, т.е. коэффициенты многочлена *I(x)* представим в виде многочленов над *GF(q)* степени $m - 1$:

$$I_j = z_{m-1}x^{m-1} + \ldots + z_{k^0}x^{k^0} + z_{k^0-1}x^{k^0-1} + z_{k^0-2}x^{k^0-2} + \ldots + z_1 x + z_0 ,$$

где $z_i \in GF(q)$, причем $m - k^0$ коэффициентов z_i заданы произвольно.

Положим, для определенности, $z_i = 0$ для $i = k^0, \ldots, m-1$. Первые k^0 элементов z_i образуют информационный кадр k^0 символов над *GF(q)*. Определенное таким образом отображение символов *GF(q)* в символы $GF(q^m)$ является однозначным соответствием. Недвоичный *(N, K, D)* циклический код над $GF(q^m)$ с проверочным многочленом *h(x)* однозначно задает рекуррентное правило кодирования

$$C(x) = I(x)/h(x).$$

При кодировании каждому информационному кадру длиной k^0 символов над *GF(q)* (или, что эквивалентно, каждому символу из множества *H*) ставится в соответствие кадр кодовых символов длиной n^0.

Степень K проверочного многочлена $h(x)$ циклического (N, K, D) кода над $GF(q^m)$ задает длину кодирующего регистра и, соответственно, число хранящихся в кодере информационных кадров. Следовательно, длина кодового ограничения ν, конструктивные параметры n и k и скорость R сверточного кодирования определяются, соответственно, следующими выражениями:

$$\nu = K \cdot k^0;\ k = (K + 1) \cdot k^0;\ n = k \cdot n^0 / k^0;\ R = k^0 / m,\ m \geq k^0.$$

Если на вход устройства (рис. 3.2) подать K информационных кадров по k^0 q–ичных символов (что эквивалентно подаче K кадров по одному q^{k^0}–ичному символу), то снятая с выхода кодовая последовательность длиной N q^m–ичных символов является кодовым словом циклического (N, K, D) кода над $GF(q^m)$. Следовательно, два любых кодовых блока, соответствующих двум произвольным входным последовательностям длиной K q^{k^0}–ичных символов, будут отличаться, по крайней мере, в D q^m–ичных символов.

Отображение элементов поля $GF(q^m)$ в элементы поля $GF(q)$ не уменьшает кодовое расстояние между произвольными q–ичными кодовыми словами, следовательно, $d_K \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_{r+1} \leq d_{r+2} \leq \ldots \leq d_\infty$, откуда $d_\infty \geq d_K$, что и завершает доказательство.

Теоремы 3.1. – 3.3. определяют механизм построения алгебраических рекурсивных сверточных кодов в несистематическом виде. Их параметры алгебраически связаны с параметрами недвоичных циклических кодов, что позволяет конструктивно строить рекурсивные сверточные коды с требуемыми свойствами. Общая схема сверточного кодера приведена на рис. 3.2 с дополнительно включенными входными и выходными буферами для отображения символов из $GF(q^m)$ в $GF(q)$ и обратно. Такой кодер реализует обработку символов из $GF(q^m)$.

Для построения схемы рекурсивного сверточного кодера с обработкой символов из $GF(q)$ рассмотрим несистематическое

кодирование через умножение информационного многочлена на порождающие многочлены $P_1(x)$, $P_2(x)$, …, $P_m(x)$.

Предположим, что некоторый многочлен $P_i(x)$ является делителем двучлена $x^n - 1$. Тогда многочлен $P_i(x)$ порождает циклический (n, k) код над $GF(q)$. Воспользовавшись результатом леммы 3.1, получим проверочный многочлен $h_i(x)$, который так же однозначно задает циклический (n, k) код над $GF(q)$. Используя цифровой рекурсивный фильтр (3.3), получим схему рекурсивного сверточного кодера в несистематическом виде с обработкой символов из $GF(q)$(см. рис. 3.3).

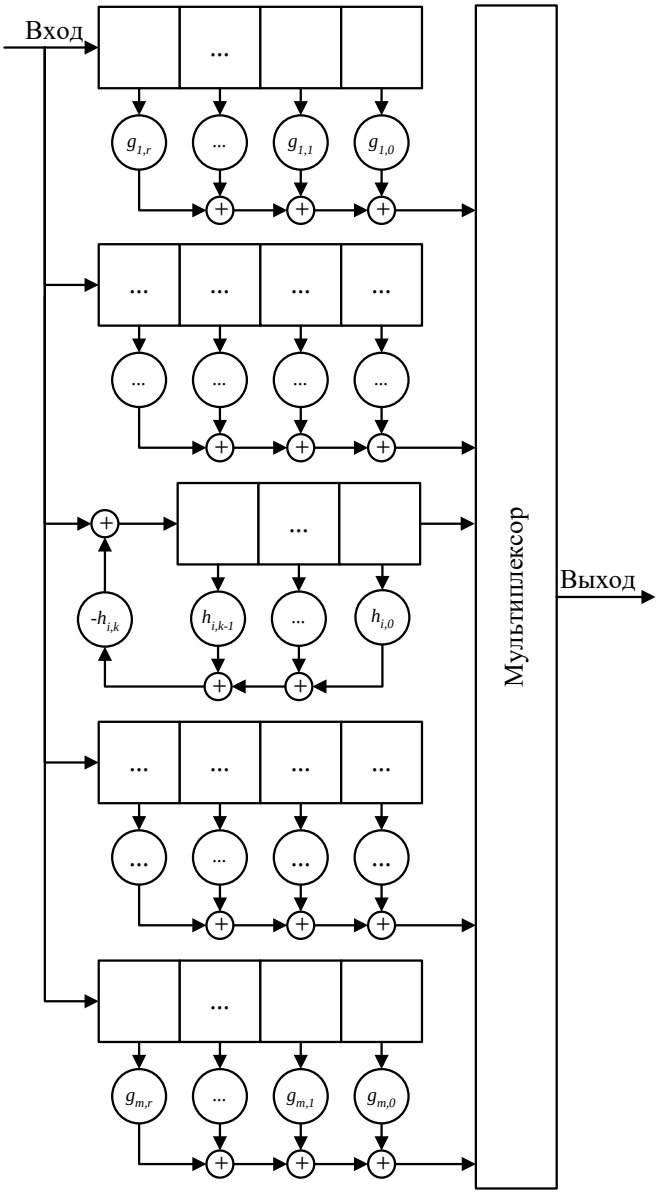


Рис. 3.3. Схема несистематического кодера алгебраического рекурсивного сверточного кода с обработкой элементов из $GF(q)$

Пример 3.1. Зафиксируем конечное поле $GF(2^3)$, построенное по кольцу многочленов с операциями по модулю многочлена $x^3 + x + 1$. Элементы поля приведены в табл. 3.1.

Зафиксируем РС код $(7, 3, 5)$ над $GF(2^3)$ с порождающим многочленом $g(x) = (x + \alpha^0)\cdot(x + \alpha^1)\cdot(x + \alpha^2)\cdot(x + \alpha^3) = x^4 + \alpha^2\cdot x^3 + \alpha^5\cdot x^2 + \alpha^5\cdot x + \alpha^6$.

Мультипликативно обратный многочлену $g(x) = x^4 + \alpha^2\cdot x^3 + \alpha^5\cdot x^2 + \alpha^5\cdot x + \alpha^6$ в кольце $GF(q)[x]/(x^n - 1)$ является многочлен $h(x) = x^3 + \alpha^2\cdot x^2 + x + \alpha^2$. Воспользовавшись результатами теорем 3.1 – 3.3, можем получить рекурсивные сверточные коды в несистематическом виде со следующими параметрами:

1. Двоичный сверточный (n, k, d) код с параметрами: $k^0 = 1$; $n^0 = 3$; $\nu = 3$; $k = 4$; $n = 12$; $R = 1 / 3$; $d_\infty \geq 5$; $d_П = 8{,}57$.
2. Двоичный сверточный код (n, k, d) код с параметрами: $k^0 = 2$; $n^0 = 3$; $\nu = 6$; $k = 8$; $n = 12$; $R = 2 / 3$; $d_\infty \geq 5$; $d_П = 8{,}57$.

Таблица 3.1

Элементы конечного поля $GF(2^3)$

0.	0	0	0	0	$\alpha^{-\infty}$
1.	0	0	1	1	α^0
2.	0	1	0	x	α^1
3.	1	0	0	x^2	α^2
4.	0	1	1	$x + 1$	α^3
5.	1	1	0	$x^2 + x$	α^4
6.	1	1	1	$x^2 + x + 1$	α^5
7.	1	0	1	$x^2 + 1$	α^6

Построим кодер с обработкой элементов из $GF(2^3)$, т.е. пакетами по 3 бита (как на рис. 3.2). На рис. 3.4 представлена схема рекурсивного кодера с обработкой символов из $GF(2^3)$.

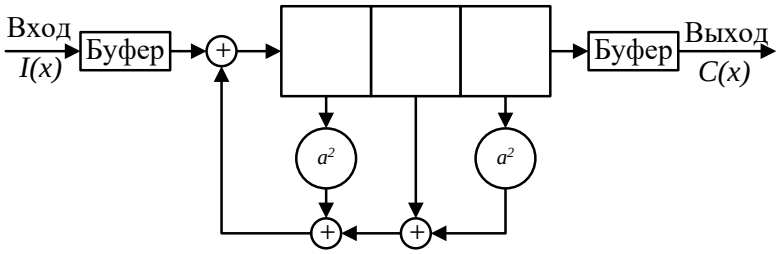


Рис. 3.4. Схема рекурсивного кодера алгебраического сверточного кода (12, 4)/(12, 8) с обработкой символов из $GF(2^3)$

Для построения кодера с обработкой двоичных символов рассмотрим порождающие многочлены алгебраического сверточного кода, заданного через порождающий многочлен РС кода:

$$g_1(x) = x^3 + x^2 + x + 1;$$

$$g_2(x) = x^2 + x;$$

$$g_3(x) = x^4 + x^2 + x + 1.$$

Многочлен $g_3(x)$ является делителем двучлена $(x^n - 1)$, его мультипликативно обратным элементом в кольце $GF(q)[x]/(x^n - 1)$ является многочлен $h_3(x) = x^3 + x + 1$. Схема соответствующего рекурсивного кодера приведена на рис. 3.5.

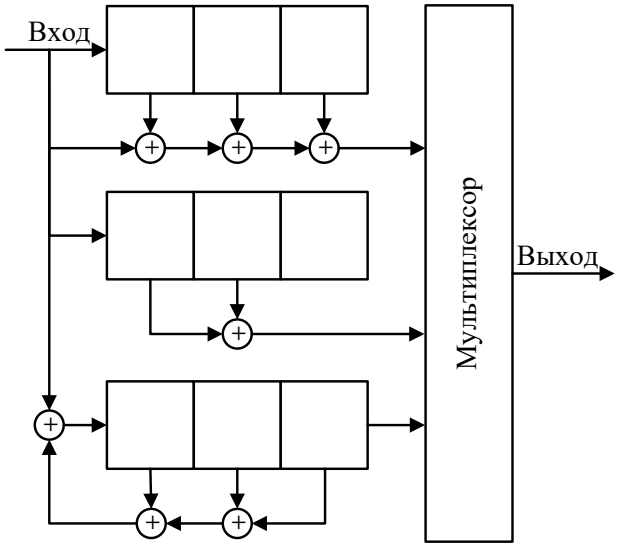


Рис. 3.5. Схема рекурсивного кодера алгебраического сверточного кода (12, 4)/(12, 8) с обработкой двоичных символов

Рассмотренный пример демонстрирует конструктивность предложенного подхода алгебраического построения рекурсивных сверточных кодов. Следует отметить, что современные методы построения турбокодов оперируют рекурсивными сверточными кодами в систематическом виде. Это позволяет формировать кодовое слово с использованием одной и той же информационной части (аналогичной для двух сверточных кодеров) и двух различных проверочных частей (для каждого сверточного кодера). Этот прием позволяет существенно повысить информационную скорость передачи при сохранении фиксированного показателя помехоустойчивости, что, соответственно, ведет к росту энергетической эффективности турбокодирования. Таким образом, проблема дальнейшей разработки, исследования алгебраических методов построения систематических сверточных кодов и их реализации на цифровых рекурсивных фильтрах с бесконечным импульсным откликом представляется весьма актуальной.

3.2. Разработка алгебраических рекурсивных сверточных кодов в систематическом виде

Для решения проблемы алгебраического построения рекурсивных систематических сверточных кодов в систематическом виде воспользуемся систематическими циклическими кодами. Рассмотрим блок данных длины n символов, поместим информационные символы в старшие разряды и подберем проверочные символы так, чтобы получить разрешенное кодовое слово, т.е. кодовое слово, принадлежащее циклическому (n, k, d) коду. Кодовое слово циклического кода в систематическом виде можно записать:

$$C(x) = x^{n-k} I(x) + T(x), \qquad (3.7)$$

где

$$T(x) = - R_{g(x)}[x^{n-k} I(x)],$$

так что

$$R_{g(x)}[C(x)] = 0.$$

Для реализации процедуры систематического кодирования циклического кода воспользуемся цифровым фильтром с бесконечным импульсным откликом (рекурсивным фильтром), который реализует цепь деления на многочлен. Пусть коэффициенты порождающего многочлена $g(x) = g_0 + g_1x + g_2x^2 + \ldots + g_rx^r$ равны весовым множителям в отводах регистра сдвига рекурсивного фильтра. Тогда схема кодера, реализующего систематическое кодирование циклических кодов по выражению (3.7) может быть представлена в виде, отображенном на рис. 3.6.

Действительно, если на вход устройства деления поступает произвольная последовательность, представленная в виде многочлена $I(x) = I_0 + I_1x + I_2x^2 + \ldots + I_{n-1}x^{n-1}$, то рекуррентные равенства процедуры деления многочленов запишем в виде

$$Q^{(i)} = Q^{(i-1)}(x) + R_{n-i}^{(i-1)}x^{k-i},$$

где $Q^{(i)}(x)$ и $R^{(i)}(x)$ – соответственно частное и остаток на i–ом шаге рекурсии с начальными значениями

$$Q^{(0)}(x) = 0 \text{ и } R^{(0)}(x) = I(x), \; R^{(i)} = R^{(i-1)}(x) - R_{n-i}^{(i-1)}x^{k-i}g(x),$$

и после k шагов итерации получаются частное $Q^{(k)}(x)$ и остаток $R^{(k)}(x)$.

Перепишем последнее выражение в виде

$$R^{(i)}(x) = I(x) - Q^{(i)}(x)g(x),$$

так что

$$R^{(i-1)} = I_{n-i} - \sum_{j=1}^{n-1} g_{n-i-j}Q_j^{(r-i)}$$

и

$$Q^{(i)}(x) = g^{(i-1)}(x) + R_{n-i}^{(i-1)}x^{k-i}.$$

Тогда деление многочлена *I(x)* на многочлен *g(x)* описывают процессы, происходящие в изображенном на рис. 3.6 регистре сдвига.

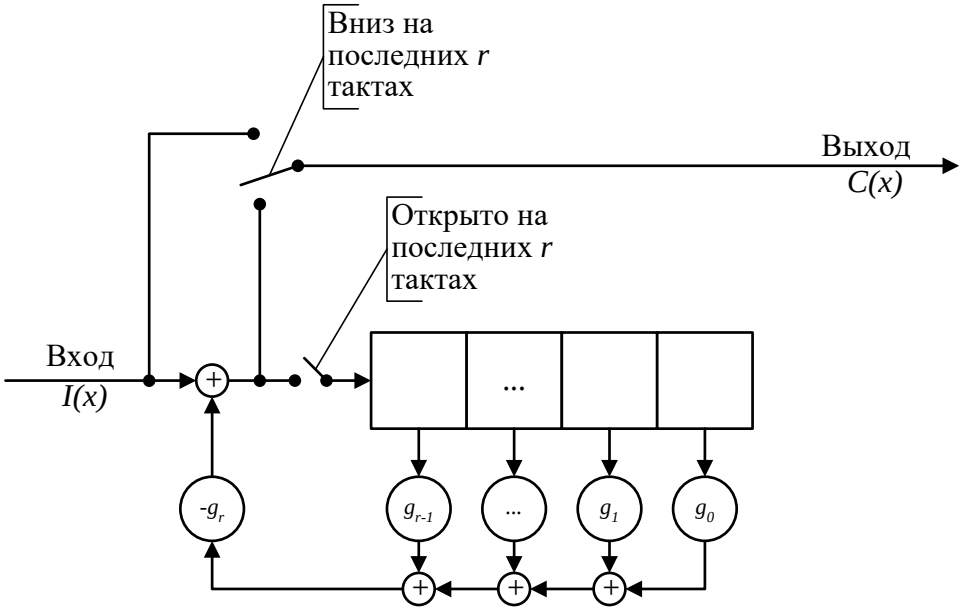


Рис. 3.6. Схема систематического кодера циклических кодов

Обобщим систематическое кодирование циклических кодов на случай бесконечной длины. Используем полученную схему для построения кодера алгебраического рекурсивного сверточного кода в систематическом виде. Схема такого кодера представлена на рис. 3.7. Устройство работает следующим образом.

Предположим, что используется циклический *(N, K, D)* код, заданный порождающим многочленом *g(x)*, *deg g(x)= r = N – K*. На вход устройства подадим непрерывный поток символов из *GF(q)*. В первом буфере входные символы накапливаются в блок из *K* символов поля *GF(q)* и подаются на вход мультиплексора и на вход рекурсивного фильтра. Через *K* тактов цифровой фильтр начинает формировать проверочные символы, которые поступают на вход второго буфера. После *N – K* тактов будут сформированы все проверочные символы и поданы на вход мультиплексора. Обработка данных осуществляется посимвольно элементами из *GF(q)*. Скорость кода *R = K /N*, его параметры определяет следующая теорема.

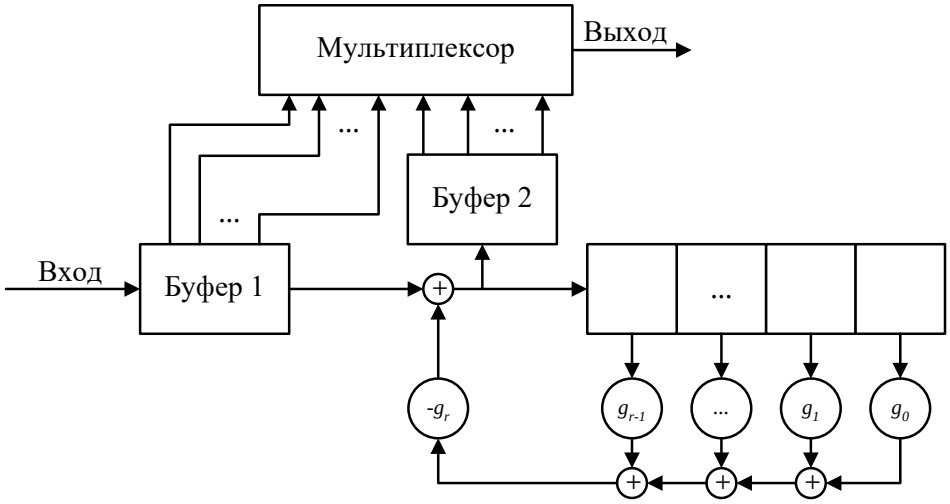


Рис. 3.7. Схема систематического кодера алгебраического сверточного кода

Теорема 3.4. Порождающий многочлен *g(x)* циклического *(N, K, D)* кода над *GF(q)* полностью определяет рекурсивный систематический сверточный *(n, k, d)* код над *GF(q)* с кодовым ограничением

$$\nu = (N - K) \cdot K$$

и параметрами

$$\begin{cases} k^0 = K, \\ n^0 = N, \\ k = (N - K + 1) \cdot K, \\ n = (N - K + 1) \cdot N, \\ R = K^0 / N, \\ d_\infty \geq D. \end{cases} \qquad (3.8)$$

Доказательство. Как известно, кодер циклического *(N, K, D)* кода в систематическом виде однозначно определяет рекурсивный фильтр с коэффициентами многочлена *g(x)* в виде весовых множителей в отводах регистра. Степень $r = N - K$ многочлена *g(x)* задает длину регистра и, соответственно, число хранящихся в регистре символов. Следовательно, $\nu = (N - K) \cdot k^0$. Если на вход устройства (см. рис. 3.7) подавать непрерывный поток символов из *GF(q)*, то на

выходе мультиплексора получится кодовое слово непрерывного кода. Если длина входного (информационного) кадра равна $k^0 = K$, а длина выходного кадра (кадра кодовых символов) равна $n^0 = K + N - K = N$, то полученная на выходе мультиплексора последовательность является кодовым словом систематического циклического кода. Скорость кода определяется выражением $R = K/N$. Для кодовых слов, соответствующих различным информационным кадрам, выполняется условие $d_1 \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_1 \leq d_2 \leq \ldots \leq d_\infty$, откуда имеем $d_\infty \geq d_1$.

Применение недвоичных циклических кодов позволяет дополнительно варьировать степенью расширения поля $GF(q^m)$ при изменении параметров сверточного кода над $GF(q)$. Устройство, приведенное на рис. 3.7, в этом случае будет работать следующим образом. На вход устройства подадим непрерывный поток символов из $GF(q)$. В первом буфере входные символы накапливаются и преобразуются в K символов из $H \subseteq GF(q^m)$. На вход мультиплексора с первого буфера поступают $K \cdot \lceil \log_q H \rceil$ символов из $GF(q)$. На вход рекурсивного фильтра с первого буфера поступают K символов, каждый из которых отождествлен символу из $GF(q^m)$. Через K тактов цифровой фильтр начинает формировать проверочные символы из $GF(q^m)$, которые поступают на вход второго буфера. После $N - K$ тактов будут сформированы все проверочные символы и поданы на вход мультиплексора в виде $(N - K) \cdot m$ символов из $GF(q)$. Таким образом, обработка входных данных в кодере, представленном на рис. 3.7, осуществляется пакетами по m символов из $GF(q)$ или, что эквивалентно, по одному элементу из $GF(q^m)$. Скорость полученного непрерывного кода над $GF(q)$ составит $K \cdot \lceil \log_q H \rceil / ((N - K) \cdot m + K \cdot \lceil \log_q H \rceil)$ и может изменяться в пределах

$$K / ((N - K) \cdot m + K) \leq R \leq K / N, \qquad (3.9)$$

в зависимости от мощности множества H. Параметры кода определяются следующей теоремой.

Теорема 3.5. Если зафиксировать $GF(q^m)$ и конечное множество H элементов поля $GF(q^m)$, причем $\log_q / H / = k^0$, $m \geq k^0$, то порождающий многочлен $g(x)$ циклического (N, K, D) кода над $GF(q^m)$ полностью определяет рекурсивный систематический сверточный (n, k, d) код над $GF(q)$ с кодовым ограничением

$$\nu = (N-K) \cdot K \cdot / \log_q H /$$

и параметрами

$$\begin{cases} k^0 = K \cdot \log_q |H|, \\ n^0 = (N-K) \cdot m + K \cdot \log_q |H|, \\ k = (N-K+1) \cdot K \cdot \log_q |H|, \\ n = (N-K+1) \cdot \left((N-K) \cdot m + K \cdot \log_q |H|\right), \\ R = \dfrac{K \cdot \log_q |H|}{(N-K) \cdot m + K \cdot \log_q |H|}, \\ d_\infty \geq D. \end{cases} \qquad (3.10)$$

Доказательство. Если на вход кодера (рис. 3.7) подать непрерывный поток символов из $GF(q)$, разбить их на блоки по $k^0 = K \cdot \log_q / H /$ символов и сопоставить набору символов из $GF(q^m)$, а кодирование в рекурсивном фильтре осуществлять элементами из $GF(q^m)$, то на выходе мультиплексора получим кодовое слово непрерывного кода.

Если при этом весовые множители будут задаваться коэффициентами многочлена $g(x)$ циклического (N, K, D) кода над $GF(q^m)$, то произвольное слово длины N символов из $GF(q^m)$ на выходе цифрового фильтра будет кодовым словом циклического (N, K, D) кода.

Длина входного (информационного) кадра равна $k^0 = K \cdot \log_q / H /$ и зависит от мощности множества H. Длина выходного кадра (кадра кодовых символов) равна

$$n^0 = ((N-K) \cdot m + K \cdot / \log_q H /)$$

и также зависит от мощности множества H. Скорость кода определяется, соответственно, выражением

$$R = K \cdot \log_q / H / / ((N-K) \cdot m + K \cdot / \log_q H /),$$

а

$$k=(N–K+1)\cdot K\cdot \lfloor \log_q H \rfloor;\ n=(N–K+1)\cdot((N–K)\cdot m+K\cdot \lfloor \log_q H \rfloor).$$

Для кодовых слов, соответствующих различным информационным кадрам, выполняется условие $d_1 \geq D$. По определению дистанционного профиля непрерывных кодов выполняется равенство $d = d_1 \leq d_2 \leq \ldots \leq d_\infty$, откуда имеем $d_\infty \geq d_1$.

Следствие 1. Если $H = GF(q^m)$, то $\log_q \lfloor H \rfloor = m$ и, очевидно,

$$k^0=K\cdot m;\ n^0=((N–K)\cdot m+K\cdot m) = N\cdot m,\ k=(N–K+1)\cdot K\cdot m;$$

$$n=(N–K+1)\cdot((N–K)\cdot m+K\cdot m) = (N–K+1)\cdot N\cdot m;$$

$$R=K\cdot m/((N–)\cdot m+K\cdot m) = K\cdot m/N\cdot m = K/N,\ d_\infty \geq D,$$

что соответствует обобщению результата теоремы 3.6 на недвоичные коды. Скорость кода соответствует верхней границе в выражении (3.9).

Следствие 2. Если $H = GF(q)$, то $\log_q \lfloor H \rfloor = 1$ и, очевидно, $k^0=K$; $n^0=((N–K)\cdot m+K)$; $k=(N–K+1)\cdot K$; $n=(N–K+1)\cdot((N–K)\cdot m+K)$; $R=K/((N–K)\cdot m+ K)$, $d_\infty \geq D$, что соответствует нижней границе скорости в выражении (3.9).

Пример. Рассмотрим конечное поле $GF(2^3)$, построенное по кольцу многочленов $\{0 = \alpha^{-\infty},\ 1 = \alpha^0,\ x = \alpha^1,\ x^2 = \alpha^2,\ x + 1 = \alpha^3,\ x^2 + x = \alpha^4,\ x^2 + x + 1 = \alpha^5,\ x^2 + 1 = \alpha^6\}$ по модулю $G(x) = x^3 + x + 1$.

Зададим код Рида Соломона (7, 5, 3) с порождающим многочленом $g(x) = (x + \alpha^0)\cdot(x + \alpha^1) = x^2 + \alpha^3 x + \alpha^1$.

Воспользуемся результатом теоремы 3.4. Получим систематический сверточный (n, k, d) код над $GF(2^3)$ с кодовым ограничением $\nu = 10$ и параметрами $k^0 = 5$; $n^0 = 7$; $k=10$; $n = 14$; $R = 5/7$; $d_\infty \geq 3$. На рис. 3.8 представлен вариант схемы такого кодера в систематическом виде.

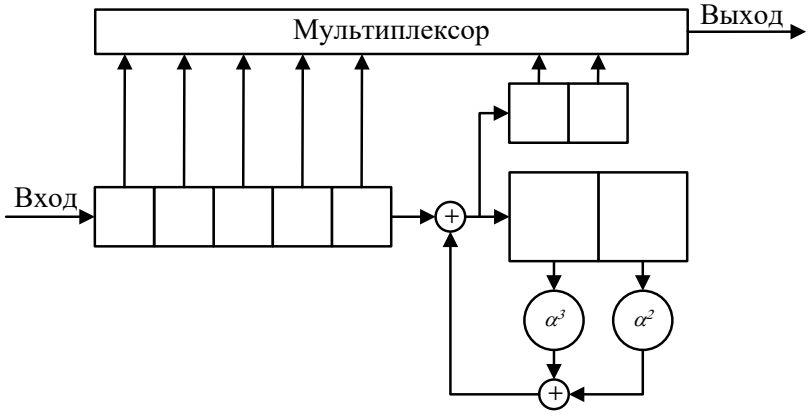


Рис. 3.8. Схема кодера алгебраического рекурсивного сверточного (14,10) кода над $GF(2^3)$ в систематическом виде (вариант)

Построим теперь двоичный алгебраический рекурсивный сверточный код с использованием результатов теоремы 3.5. Зафиксируем множество $H \subset GF(2^3)$ так, что $\log_2 |H| = k^0 = 2$, например, $H = \{0, 1, x^2 + x, x^2 + x + 1\}$.

Каждый элемент множества H сопоставим информационному кадру из $k^0 = 2$ бит, например, следующим образом (см. табл. 3.2).

Таблица 3.2

Сопоставление информационных кадров элементам множества H

Инф. кадр		Элемент множества H
00	$\leftrightarrow$	0(0, 0, 0)
01	$\leftrightarrow$	α^0(0, 0, 1)
10	$\leftrightarrow$	α^4(1, 1, 0)
11	$\leftrightarrow$	α^5(1, 1, 1)

Отображение информационных кадров в символы множества H состоит только в дописывании слева (копирование) первого бита информационной последовательности. Воспользовавшись результатом теоремы 3.7, получим двоичный сверточный (n, k, d) код с кодовым ограничением $\nu=20$ и параметрами $k^0=10$; $n^0=16$; $k=30$; $n=48$; $R=5/8$; $d_\infty \geq 3$. Возможный вариант схемы такого кодера в систематическом виде представлен на рис. 3.9.

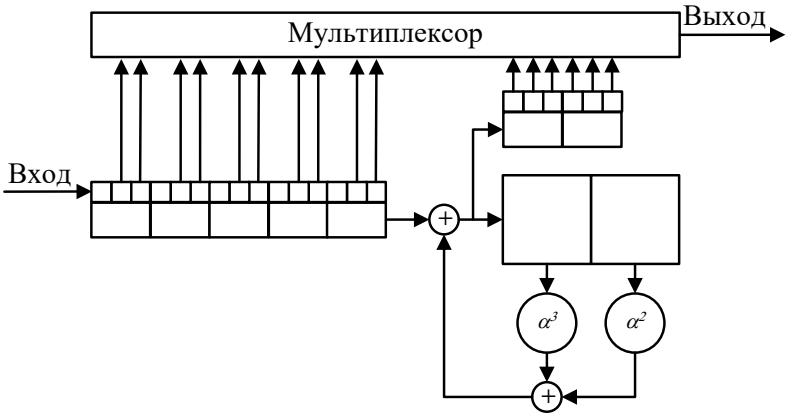


Рис. 3.9. Схема кодера алгебраического двоичного сверточного *(48, 30)* кода в систематическом виде (вариант)

На практике в существующих схемах рекурсивных сверточных кодов в систематическом виде входные и выходные буферы заменяют мультиплексированием и добавляют дополнительное устройство, выполняющее процедуру кодирования проверочной части нерекурсивным несистематическим кодом. Тогда общая схема кодера может быть представлена в виде (см. рис. 3.10).

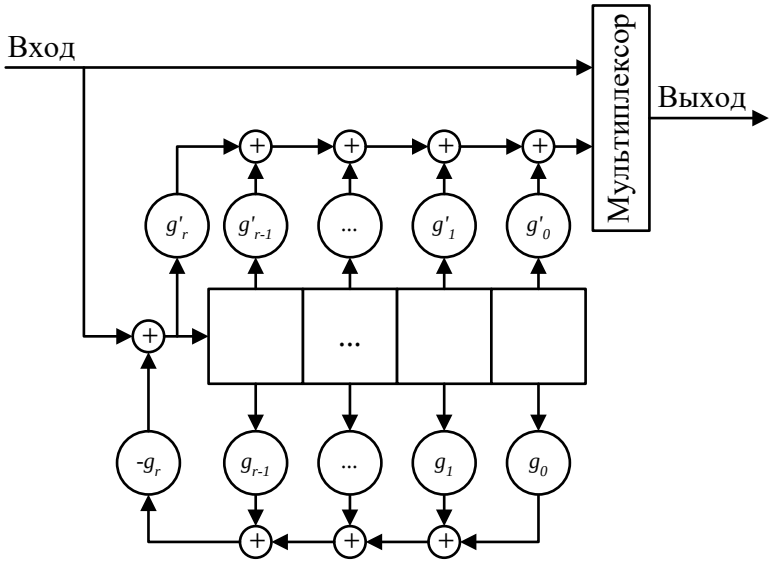


Рис. 3.10. Общая схема рекурсивного систематического кодера

Возможен другой способ алгебраического построения рекурсивных сверточных кодов в систематическом виде. Этот способ несколько проще, но не всегда реализуем. Он состоит в возможном представлении одного из многочленов несистематического нерекурсивного алгебраического сверточного кода в виде единичного многочлена и трансформации кодера к требуемой форме. Действительно, если зафиксировать конечное поле $GF(q^m)$, некоторое множество $H \subseteq GF(q^m)$, циклический (N, K, D) код над $GF(q^m)$ и образованные таким способом порождающие многочлены несистематического нерекурсивного сверточного кода $g_1(x), g_2(x), \ldots, g_m(x)$, то для построения систематического сверточного кода оказывается необходимым и достаточным выполнение условия следующей леммы.

Лемма 3.2. Для построения сверточного кода с $R = 1/m$ в систематическом виде необходимо и достаточно выполнение равенства единице любого многочлена из выражения $g(x) = \prod_i (x - \beta^i)$ (3.1).

Доказательство. Пусть информационная последовательность описывается многочленом вида

$$I(x) = i_{r-1}x^{r-1} + i_{r-1}x^{r-1} + \ldots + i_1 x + i_0$$

и является информационной последовательностью, подлежащей кодированию. Кодовое слово $C(x)$ сверточного кода формируется путем последовательного считывания символов при одинаковых степенях многочленов

$$F_1(x) = I(x)P_1(x),\ F_2(x) = I(x)P_2(x), \ldots, F_m(x) = I(x)P_m(x),$$

где $I(x)$ – информационный многочлен.

Для простоты предположим, что единице равен первый многочлен в (3.1). Тогда $F_1(x) = I(x)$, а кодовое слово запишется в виде

$$C(x) = (0, s_{2,2r-2}, ..., s_{m,2r-2})x^{2r-2} + (0, s_{2,2r-3}, ..., s_{m,2r-3})x^{2r-3} + ... +$$
$$+ (i_{r-1}, s_{2,r-1}, ..., s_{m,r-1})x^{r-1} + ... + (i_1, s_{2,1}, ..., s_{m,1})x + (i_0, s_{2,0}, ..., s_{m,0}),$$

где $s_{i,j}$ – коэффициенты в многочлене $F_i(x)$ при x^j, образующиеся в результате перемножения многочленов $I(x)$ и $P_i(x)$.

Очевидно, что в первых r кодовых кадрах из $n^0 = m$ символов в явном виде содержится по одному информационному символу i_0, i_1, ..., i_{r-1}, что соответствует систематическому виду кодирования с $R = 1/m$.

Практический интерес представляет систематическое кодирование с $R = k^0 1/m$. Поэтому обобщим лемму 3.2 для сверточных кодов с $R = k^0/m$.

Лемма 3.3. Для построения сверточного кода с $R = k^0/m$ в систематическом виде необходимо и достаточно выполнение равенства единице любых k^0 многочленов из выражения $g(x) = \prod_i (x - \beta^i)$ (3.1).

Доказательство аналогично лемме 3.2. Действительно, приравняв единице любые k^0 порождающих многочленов и записав соответствующее кодовое слово, получим, что в каждом из первых r кодовых кадров из $n^0 = m$ символов в явном виде содержатся информационные символы i_0, i_1, ..., i_{k0}, что соответствует систематическому виду кодирования с $R = k^0/m$.

Схема построенного таким образом алгебраического рекурсивного сверточного кода в систематическом виде с обработкой элементов из $GF(q)$ в общем виде представлена на рис. 3.11.

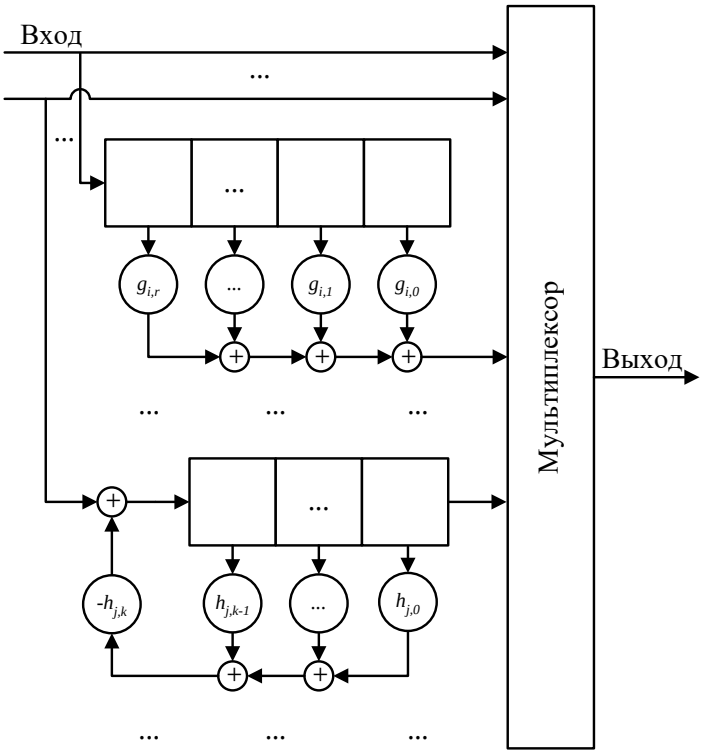


Рис. 3.11. Схема систематического кодера алгебраического рекурсивного сверточного кода с обработкой элементов из *GF(q)*

Приведем *пример* использования леммы 3.2 для построения систематического кодера алгебраического рекурсивного сверточного кода. Зафиксируем конечное поле $GF(2^2)$, построенное по кольцу многочленов

$$\{0 = \alpha^{-\infty}, 1 = \alpha^0, x = \alpha^1, x + 1 = \alpha^2\}$$

с операциями, по модулю

$$G(x) = x^2 + x + 1.$$

Зафиксируем (3, 2, 2) код РС с порождающим многочленом $g(x) = x + \alpha^2$. Тогда соответствующие порождающие многочлены несистематического нерекурсивного сверточного кода окажутся равными

$$g_1(x) = x + 1;$$

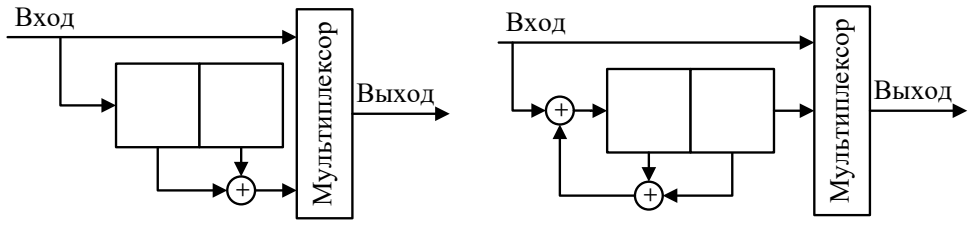
$$g_2(x) = 1.$$

Воспользовавшись результатом леммы 3.2 построим нерекурсивный сверточный кодер в систематическом виде, схема устройства представлена на рис. 3.12, а.

Многочлен $g_1(x) = x + 1$ является делителем многочлена $x^3 + 1$. Мультипликативно обратным ему в кольце многочленов с операциями по модулю $x^3 + 1$ является многочлен $h_1(x) = x^2 + x + 1$. Тогда рекурсивный сверточный код в систематическом виде, построенный по порождающему многочлену (3, 2, 2) кода РС, имеет параметры

$$\nu = 2;\ k^0 = 1;\ n^0 = 2;\ k=3;\ n = 6;\ R = 1 / 2;\ d_\infty \geq 2.$$

Соответствующая схема кодера представлена на рис. 3.12, б.



а б

Рис. 3.12. Схема сверточного кодера

Приведенный пример показывает возможности предложенного подхода алгебраического построения рекурсивных сверточных кодов в систематическом виде. Он значительно проще, чем метод, предложенный теоремами 2.4–2.7. Однако он предъявляет более жесткое требование – выполнение условий лемм 3.2. – 3.3, т.е. необходимо равенство единице любых k^0 многочленов из выражения $g(x) = \prod_i (x - \beta^i)$ (3.1).

3.3. Разработка алгоритмов построения рекурсивных сверточных кодов

Разработанный метод построения рекурсивных сверточных кодов позволяет выразить конструктивные параметры алгебраически заданных сверточных кодов через соответствующие

параметры циклических кодов (см. теоремы 3.3–3.7). Для практического использования полученных результатов разработаны алгоритмы алгебраического построения рекурсивных сверточных кодов в систематическом и несистематическом виде.

Для алгебраического построения рекурсивного сверточного кода с конструктивными *(n, k, d)* параметрами необходимо и достаточно задать порождающий и/или проверочный многочлен циклического *(N, K, D)* кода. При этом конструктивные параметры сверточного *(n, k, d)* кода будут аналитически связаны с параметрами циклического *(N, K, D)* кода и могут быть заданы выражениями (3.5–3.10). А*лгоритм построения рекурсивных сверточных кодов* в общем виде представим в виде последовательности шагов.

ШАГ 1. Ввод параметров рекурсивного сверточного *(n, k, d)* кода и мощности алфавита кодовых символов *q*.

ШАГ 2. Выбор варианта построения сверточного кода над *GF(q)*:

– рекурсивного сверточного *(n, k, d)* кода с $R = 1/m$ в несистематическом виде (см. теорему 3.2);

– рекурсивного сверточного *(n, k, d)* кода с $R = k^0/m$ в несистематическом виде (см. теорему 3.3);

– рекурсивного сверточного *(n, k, d)* кода с $R = K/N$ в систематическом виде (см. теорему 3.4);

– рекурсивного сверточного *(n, k, d)* кода с $R = (K \cdot \log_q H)/((N-K) \cdot m + K \cdot \log_q H)$ в систематическом виде (см. теорему 3.5).

ШАГ 3. Расчет параметров циклического *(N, K, D)* кода над $GF(q^m)$.

ШАГ 4. Выбор и формирование порождающего и/или проверочного многочлена циклического *(N, K, D)* кода над $GF(q^m)$.

ШАГ 5. Выбор способа обработки кодовых символов. Формирование порождающих многочленов рекурсивного сверточного *(n, k, d)* кода над *GF(q)*, построение схемы кодера рекурсивного сверточного *(n, k, d)* кода над *GF(q)*.

Разработанный алгоритм позволяет конструктивным способом за конечное число шагов построить рекурсивный сверточный код с требуемыми параметрами. Схема алгоритма представлена на рис. 3.13. После ввода параметров сверточного кода (шаг 1) и выбора варианта его построения (шаг 2) на третьем шаге алгоритма производится расчет параметров циклического *(N, K, D)* кода над $GF(q^m)$. Рассмотрим алгоритм подробно.

В случае, когда на втором шаге алгоритма выбран первый вариант построения сверточного кода, воспользуемся результатами теоремы 3.4. Зафиксируем конечное поле *GF(q)* и параметры рекурсивного сверточного *(n, k, d)* кода с $R = 1/m$ в несистематическом виде над *GF(q)*. По теореме 3.2 такой код однозначно задается проверочным многочленом *h(x)* циклического *(N, K, D)* кода над $GF(q^m)$.

Воспользовавшись выражением (3.5), выразим параметры циклического *(N, K, D)* кода над $GF(q^m)$ через фиксированные параметры рекурсивного сверточного *(n, k, d)* кода в несистематическом виде над *GF(q)*. Получим

$$\begin{cases} K = k - 1, \\ D = d_{\infty}, \\ m = n^0. \end{cases} \tag{3.11}$$

Если при этом требуется построить сверточный код с длиной кодового ограничения *v*, то необходимо использовать циклический *(N, K, D)* код над $GF(q^m)$ с $K = v$. На этом третий шаг алгоритма для первого варианта построения рекурсивного сверточного кода завершен.

Когда на втором шаге алгоритма выбран второй вариант построения сверточного кода, воспользуемся теоремой 3.3. Зафиксируем конечное поле *GF(q)* и параметры рекурсивного сверточного *(n, k, d)* кода с $R = k^0/m$ в несистематическом виде над *GF(q)*. По теореме 3.3 такой код однозначно задается проверочным многочленом *h(x)* циклического *(N, K, D)* кода над $GF(q^m)$. Используя выражение (3.6), выразим параметры циклического *(N, K, D)* кода над $GF(q^m)$ через фиксированные параметры рекурсивного сверточного *(n, k, d)* кода в несистематическом виде над *GF(q)*.

Получим

$$\begin{cases} K = k / k^0 - 1, \\ D = d_\infty, \\ m = n \cdot k^0 / k. \end{cases} \qquad (3.12)$$

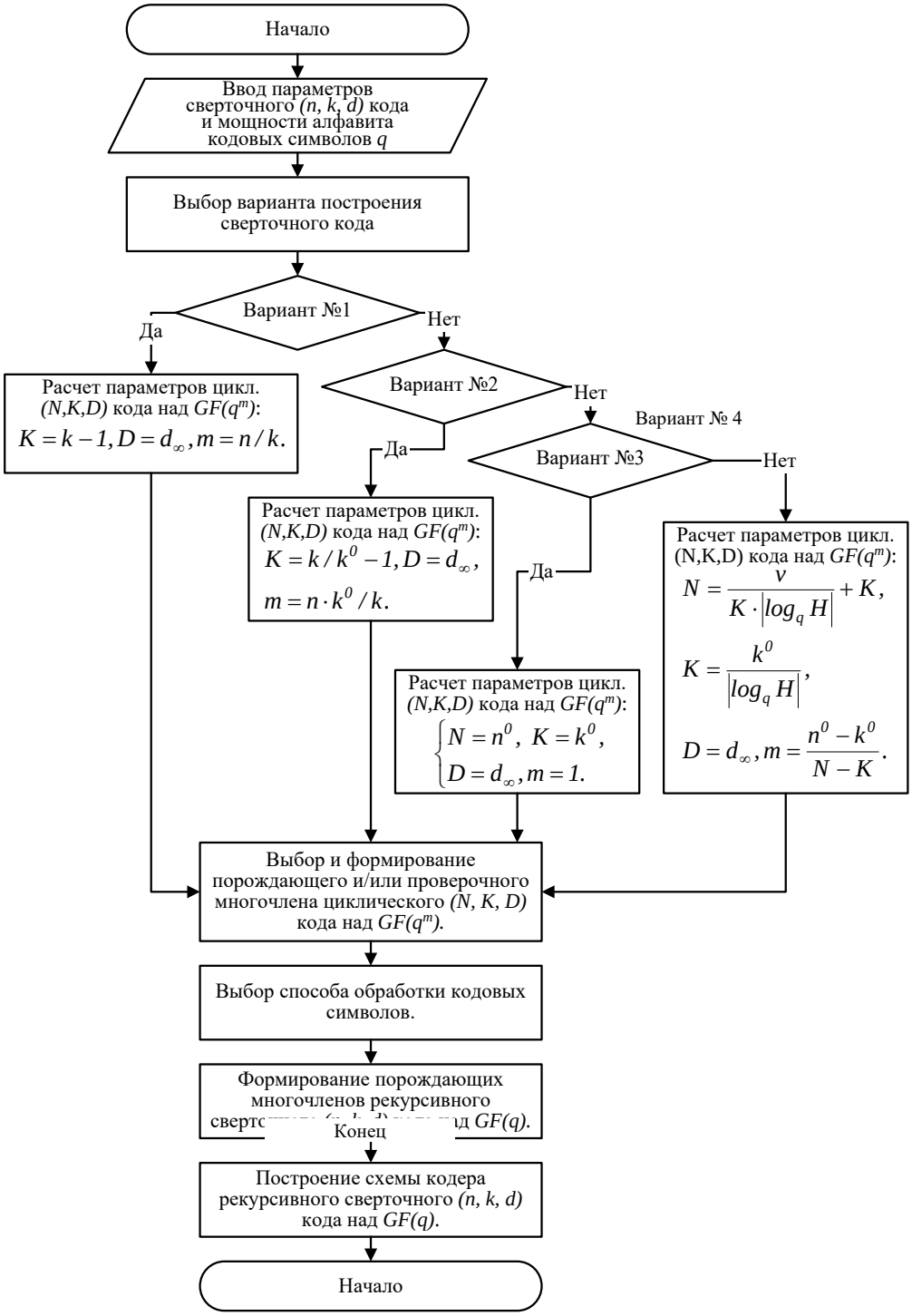


Рис. 3.13. Схема алгоритма построения рекурсивных сверточных кодов

Если при этом требуется построить сверточный код с длиной кодового ограничения v, то необходимо использовать циклический (N, K, D) код над $GF(q^m)$ с $K = v/k^0$. На этом третий шаг алгоритма для второго варианта построения рекурсивного сверточного кода завершен.

Предположим, что на втором шаге алгоритма выбран третий вариант построения сверточного кода. Воспользуемся результатами теоремы 3.4. Зафиксируем конечное поле $GF(q)$ и параметры рекурсивного сверточного (n, k, d) кода в систематическом виде над $GF(q)$. По теореме 3.4 такой код с $R = K/N$ однозначно задается порождающим многочленом $g(x)$ циклического (N, K, D) кода над $GF(q^m)$. Воспользуемся выражением (3.8), выразим параметры циклического (N, K, D) кода над $GF(q^m)$ через фиксированные параметры рекурсивного сверточного (n, k, d) кода в систематическом виде над $GF(q)$. Получим

$$\begin{cases} N = n^0, \\ K = k^0, \\ D = d_\infty, \\ m = 1. \end{cases} \tag{3.13}$$

Если при этом требуется построить сверточный код с длиной кодового ограничения v, то необходимо использовать циклический (N, K, D) код над $GF(q^m)$ с $K = (n^0 - k^0)\cdot k^0$. На этом третий шаг алгоритма для третьего варианта построения рекурсивного сверточного кода завершен.

Предположим, что на втором шаге алгоритма выбран четвертый вариант построения сверточного кода. Воспользуемся результатами теоремы 3.5. Зафиксируем конечное поле $GF(q)$ и параметры рекурсивного сверточного (n, k, d) кода в систематическом виде над $GF(q)$. По теореме 3.5 такой код с

$$R = \frac{K \cdot \left| \log_q H \right|}{(N - K)\cdot m + K \cdot \left| \log_q H \right|}$$

однозначно задается порождающим многочленом $g(x)$ циклического (N, K, D) кода над $GF(q^m)$. Воспользуемся выражением (3.10), выразим параметры циклического (N, K, D)

кода над $GF(q^m)$ через фиксированные параметры рекурсивного сверточного (n, k, d) кода в систематическом виде над $GF(q)$. Получим

$$\begin{cases} N = \dfrac{v}{K \cdot \left| log_q H \right|} + K, \\ K = \dfrac{k^0}{\left| log_q H \right|}, \\ D = d_\infty, \\ m = \dfrac{n^0 - k^0}{N - K}. \end{cases} \qquad (3.14)$$

На этом третий шаг алгоритма для четвертого варианта построения рекурсивного сверточного кода завершен.

Рассмотрим особенности выполнения четвертого шага разработанного алгоритма. На этом шаге алгоритма производится выбор схемы кодирования циклического кода (через порождающий или проверочный многочлен), что определяет так же схему кодирования рекурсивного сверточного кода.

Предположим, что в качестве циклического кода выбран примитивный код БЧХ, его длина равна $N = (q^m)^M - 1$. Рассмотрим поле разложения двучлена $(x^M - 1)$ на минимальные многочлены элементов поля $GF((q^m)^M)$ над $GF(q^m)$. Порождающий многочлен примитивного кода БЧХ задается в виде

$$g(x) = НОК(f_1, f_2, \ldots, f_{2t}),$$

где $D = 2t + 1$, f_i – минимальный многочлен над $GF(q^m)$ элементов $\alpha^i \in GF((q^m)^M)$.

Проверочный многочлен $h(x)$ определим как сомножитель $g(x)$ в разложении двучлена $x^N - 1$:

$$h(x) = (x^N - 1)/g(x).$$

Последнее выражение эквивалентно следующему:

$$h(x) = НОК(\varphi_1, \varphi_2, \ldots),$$

где φ_j – минимальный многочлен над $GF(q^m)$ элементов $\alpha^j \in GF((q^m)^M)$, причем $\alpha^i \neq \alpha^j$.

Рассмотрим случай, когда в качестве циклического кода выбран непримитивный код БЧХ. По определению, длина непримитивного кода БЧХ равна одному из сомножителей в разложении числа $(q^m)^M - 1$ (если, конечно, число $(q^m)^M - 1$ не является простым), т.е. $N = ((q^m)^M - 1)/g$ для произвольного целого g, делящего нацело число $(q^m)^M - 1$. Очевидно, что должно выполняться также условие $r < N$.

Порождающий многочлен непримитивного кода БЧХ задается в виде

$$g(x) = НОК(f_1, f_2, \ldots, f_{2t}),$$

где $D = 2t + 1$, f_i – минимальные многочлены над $GF(q^m)$ элементов $\beta^i \in GF((q^m)^M)$ такие, что их порядок равен N, т.е. $\beta^i = \alpha^{jg}$, $j = 1, 2, \ldots, M/2$.

Проверочный многочлен определяется аналогично случаю, рассмотренному выше.

Рассмотрим случай, когда в качестве циклического кода выбран код РС. По определению, порождающий многочлен кода РС задается в виде

$$g(x) = (x - \alpha^i) \cdot (x - \alpha^{2i}) \cdot \ldots \cdot (x - \alpha^{2ti}),$$

где $D = 2t + 1$; $\alpha^i \in GF(q^m)$.

Аналогично рассмотренному выше случаю формируется проверочный многочлен $h(x)$.

На шестом шаге алгоритма выбирается способ обработки кодовых символов: по одному элементу из $GF(q)$ или пакетами по m элементов из $GF(q)$ – по одному символу из $GF(q^m)$. В соответствии с выбранным способом обработки формируются порождающие многочлены рекурсивного сверточного кода.

Таким образом, разработанный алгоритм позволяет алгебраически строить рекурсивные сверточные коды в систематическом и несистематичесом виде.

Проведем исследования конструктивных свойств алгебраически заданных рекурсивных сверточных кодов.

3.4. Исследование свойств алгебраически заданных рекурсивных сверточных кодов

Зафиксируем конечное поле $GF(2^2)$ и рассмотрим коды РС с параметрами $N = 2^2 - 1 = 3$; $3 - K = D - 1$.

Воспользуемся результатами теорем 2.2. – 2.3. Исследуем конструктивные кодовые параметры алгебраически заданных рекурсивных сверточных кодов в несистематическом виде. В табл. 3.3 представлены параметры кодов РС над $GF(2^2)$, конструктивные параметры рекурсивных сверточных (n, k, d) кодов в несистематическом виде, алгебраически заданных порождающим многочленом кода РС, предсказанное и истинное значение свободного кодового расстояния. Случаи для кодов $(n, 1, n)$ соответствуют тривиальному коду с повтором символов.

Таблица 3.3

Конструктивные характеристики двоичных рекурсивных сверточных кодов в несистематическом виде, заданных через порождающий многочлен кода РС над $GF(2^2)$

(N, K, D)	(n, k, d)	v	R	$d_П$	d_∞
(3, 1, 3)	(4, 2, 3)	1	1 / 2		
(3, 2, 2)	(6, 3, 3)	2	1 / 2		

Воспользуемся результатами теорем 2.4.–2.5. Исследуем конструктивные кодовые параметры алгебраически заданных рекурсивных сверточных кодов в систематическом виде.

В табл. 3.4 представлены параметры кодов РС над $GF(2^2)$, конструктивные параметры рекурсивных сверточных (n, k, d) кодов в несистематическом виде, алгебраически заданных порождающим многочленом кода РС, предсказанное и истинное значение свободного кодового расстояния.

Таблица 3.4

Конструктивные характеристики двоичных рекурсивных сверточных кодов в систематическом виде, заданных через порождающий многочлен кода РС над $GF(2^2)$

(N, K, D)	(n, k, d)	v	R	$d_П$	d_∞
(3, 1, 3)	(15, 3, 3)	2	1 / 5		
	(18, 6, 3)	4	1 / 3		
(3, 2, 2)	(8, 4, 2)	2	1 / 2		
	(12, 8, 2)	4	2 / 3		

Зафиксируем конечное поле $GF(2^3)$ и рассмотрим коды РС с параметрами

$$N = 2^3 - 1 = 7;\ 7 - K = D - 1.$$

В табл. 3.5 представлены параметры кодов РС над $GF(2^3)$, конструктивные параметры рекурсивных сверточных (n, k, d) кодов в несистематическом виде, алгебраически заданных порождающим многочленом кода РС, предсказанное и истинное значение свободного кодового расстояния.

Таблица 3.5

Конструктивные характеристики двоичных рекурсивных сверточных кодов в несистематическом виде, заданных через порождающий многочлен кода РС над $GF(2^3)$

(N, K, D)	(n, k, d)	v	R	$d_П$	d_∞
	(6, 2, 7)	1	1 / 3		
(7, 1, 7)	(6, 4, 7)	2	2 / 3		
	(9, 3, 6)	2	1 / 3		
(7, 2, 6)	(9, 6, 6)	4	2 / 3		
	(12, 4, 5)	3	1 / 3		
(7, 3, 5)	(12, 8, 5)	6	2 / 3		
	(15, 5, 4)	4	1 / 3		
(7, 4, 4)	(15, 10, 4)	8	2 / 3		
	(18, 6, 3)	5	1 / 3		
(7, 5, 3)	(18, 12, 3)	10	2 / 3		
	(21, 7, 2)	6	1 / 3		
(7, 6, 2)	(21, 14, 2)	12	2 / 3		

В табл. 3.6 представлены параметры кодов РС над $GF(2^3)$, конструктивные параметры рекурсивных сверточных (n, k, d) кодов в систематическом виде, алгебраически заданных порождающим многочленом кода РС, предсказанное и истинное значение свободного кодового расстояния.

Таблица 3.6

Конструктивные характеристики двоичных рекурсивных сверточных кодов в несистематическом виде, заданных через порождающий многочлен кода РС над $GF(2^3)$

(N, K, D)	(n, k, d)	v	R	$d_П$	d_∞
(7, 1, 7)	(133, 7, 7)	6	1 / 19		
	(140, 14, 7)	12	1 / 10		
	(147, 21, 7)	18	1 / 7		
(7, 2, 6)	(102, 12, 6)	10	2 / 17		
	(114, 24, 6)	20	4 / 19		
	(126, 36, 6)	30	2 / 7		
(7, 3, 5)	(75, 15, 5)	12	1 / 5		
	(90, 30, 5)	24	1 / 3		
	(105, 45, 5)	36	3 / 7		
(7, 4, 4)	(52, 16, 4)	12	4 / 13		
	(68, 32, 4)	24	8 / 17		
	(84, 48, 4)	36	4 / 7		
(7, 5, 3)	(33, 15, 3)	10	5 / 11		
	(48, 30, 3)	20	6 / 8		
	(63, 45, 3)	30	5 / 7		
(7, 6, 2)	(18, 12, 2)	6	2 / 3		
	(30, 24, 2)	12	4 / 5		
	(42, 36, 2)	18	6 / 7		

Конструктивные кодовые параметры, приведенные в табл. 3.3, 3.5, свидетельствуют о том, что для построения

хороших рекурсивных сверточных кодов в несистематическом виде следует использовать низкоскоростные циклические коды, например коды РС. При этом удается получить практически весь спектр скоростей кодирования сверточных кодов и даже при небольшом кодовом ограничении хорошие кодовые параметры.

Как следует из значений, приведенных в табл. 3.4, 3.6, для построения хороших рекурсивных сверточных кодов в систематическом виде следует использовать высокоскоростные циклические коды, например коды РС. При этом также удается получить практически весь спектр скоростей кодирования сверточных кодов и даже при небольшом кодовом ограничении хорошие кодовые параметры.

Выводы

1. Получили дальнейшее развитие алгебраические методы построения сверточных кодов, отличающиеся от известных представлением сверточного кода через порождающий многочлен рекурсивного циклического кода и ограничением на произвольном подполе. Это позволяет строить рекурсивные сверточные коды в систематическом и несистематическом виде с требуемыми конструктивными характеристиками.

2. Теоретически обоснованы процедуры алгебраического построения рекурсивных сверточных кодов через обобщение циклических кодов на случай бесконечной длины. Доказанные теоремы 2.4–2.5 позволяют аналитически связать параметры несистематических циклических кодов, заданных через проверочный многочлен, с конструктивными параметрами рекурсивных сверточных кодов в несистематическом виде. Сформулированные и доказанные теоремы 2.6–2.7 позволяют аналитически связать параметры систематических циклических кодов, заданных через порождающий многочлен с конструктивными параметрами рекурсивных сверточных кодов в систематическом виде.

3. Предложенный алгоритм построения сверточных кодов практически реализует алгебраический подход к описанию рекурсивных сверточных кодов, учитывает различные варианты их построения, что позволяет алгебраически задавать кодовые

конструкции с требуемыми свойствами, строить схемы кодеров и аналитически рассчитывать конструктивные кодовые параметры.

4. Проведенные исследования свойств алгебраически заданных рекурсивных сверточных кодов показали, что для построения хороших кодов в несистематическом виде следует использовать низкоскоростные циклические коды, а для построения хороших кодов в систематическом виде необходимо применять высокоскоростные циклические коды. Это позволяет получить практически весь спектр скоростей сверточного кодирования и даже при небольшом кодовом ограничении хорошие конструктивные параметры.

РАЗДЕЛ 4
РАЗРАБОТКА И ИССЛЕДОВАНИЕ МЕТОДОВ ДЕКОДИРОВАНИЯ АЛГЕБРАИЧЕСКИХ СВЕРТОЧНЫХ КОДОВ

Разрабатывается алгебраический метод декодирования сверточных кодов, основанный на использовании бесконечной серии синдромов кодовых слов циклического кода. Предлагается способ формирования бесконечной серии синдромов алгебраического сверточного кода, исследуются особенности работы кодера при его реализации. Разрабатывается подход комбинированного декодирования алгебраических сверточных кодов, состоящий в совмещении алгебраических процедур и последовательного поиска по кодовой решетке.

4.1. Исследование существующих методов декодирования сверточных кодов

Развитие теории декодирования сверточных кодов происходило в трех направлениях разработки методов и алгоритмов:

- методы и алгоритмы порогового декодирования;
- методы и алгоритмы декодирования по максимуму правдоподобия;

− методы и алгоритмы последовательного декодирования.

Общая классификация известных методов декодирования сверточных кодов приведена на рис. 4.1.

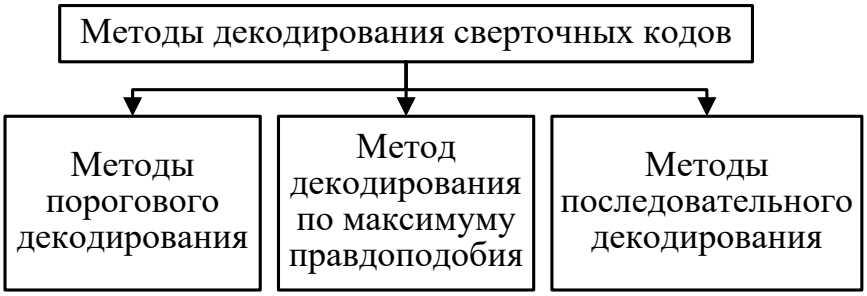


Рис. 4.1. Классификация методов декодирования сверточных кодов

Методы порогового декодирования аналогичны методам мажоритарного декодирования блоковых кодов. Достоинством последних является простота построения алгоритмов и практической реализации декодеров. Число операций, необходимых для декодирования одного информационного символа, при этом не превосходит некоторой постоянной величины.

Методы декодирования по максимуму правдоподобия теоретически, с точки зрения реализации исправляющей способности кода, более эффективны по сравнению с предыдущими. Однако сложность алгоритмов и практических устройств, необходимых для реализации, растет экспоненциально с ростом длины кода.

В основе *методов последовательного декодирования* лежит вероятностный подход, при котором число операций, необходимых для декодирования одного символа, является случайной величиной.

Существенный интерес представляет исследование особенностей различных методов декодирования сверточных кодов и возможности применения этих методов к декодированию алгебраически заданных непрерывных кодов.

Принципы порогового декодирования

Пороговое декодирование сверточных кодов основано на тех же принципах, что и мажоритарное декодирование блоковых кодов. Этот подход является одним из наиболее простых и удобных в реализации методов декодирования. Практическая реализация таких декодеров привлекательна высоким быстродействием и низкой сложностью.

Суть рассматриваемого метода декодирования состоит в мажоритарном оценивании веса ошибки, произошедшей в l–ом символе кодового слова. Подобная оценка проводится над проверочными уравнениями, ортогональными относительно l–ого кодового символа. Ортогональное относительно l–ой координаты подмножество проверочных уравнений состоит из всех уравнений, в каждое из которых входит l–я компонента, а остальные компоненты входят не более чем в одно уравнение. Если структура кода такова, что множество проверочных уравнений ортогонально относительно нескольких координат, то мажоритарное решение применимо для локализации ошибки в подмножестве этих компонент. Для нахождения ошибки также используется мажоритарная логика. Многошаговый мажоритарный декодер, работающий по такому принципу, позволяет декодировать большее число ошибок, но, тем не менее, не всегда позволяет декодировать все ошибки по минимальному расстоянию. Максимальное число ошибок, которые правит одношаговый мажоритарный декодер, определяется следующей теоремой.

Теорема 4.1. Число ошибок, которые исправляет одношаговый мажоритарный декодер для произвольного кода над $GF(q)$, не превосходит $(n-1)/(2d_{\perp}-2)$, где $d_{\perp}$ – минимальное кодовое расстояние дуального кода.

Доказательство теоремы основано на том факте, что проверочные уравнения мажоритарного декодирования соответствуют кодовым словам дуального кода. Число ортогональных относительно любой из координат проверок не превышает $(n-1)/(2t)$. Известно, что если для каждой

координаты мажоритарного декодера имеется l проверок, ортогональных относительно нее, то декодер исправляет $l/2$ ошибок. Следовательно, число ошибок, которые исправляет одношаговый мажоритарный декодер, не превосходит $(n-1)/(2d_{\perp}-2)$.

Несколько большее число ошибок позволяет править многошаговый декодер.

Теорема 4.2. Число ошибок, которые правит многошаговый мажоритарный декодер для произвольного кода над $GF(q)$, не превосходит $n/d_{\perp}-0{,}5$.

Доказательство аналогично предыдущему.

При пороговом декодировании сверточных кодов предлагается использовать аналогичный принцип мажоритарного принятия решения. При этом ошибки в бесконечном кодовом слове последовательно исправляются сначала в первом принятом блоке длины n^0 кодовых символов, затем во втором и т.д. Такой подход может привести к негативным последствиям, к так называемому эффекту распространения ошибок. Влияние найденных ошибок может быть исключено и соответствующий синдром откорректирован (с помощью обратной связи).

Если при этом ошибка декодирована неправильно, синдром будет откорректирован не верно, что приведет к последующему неверному декодированию. Если декодирование не предусматривает коррекцию символов синдрома, то его называют дефинитным. При этом корректирующие способности кода ухудшаются. Однако, благодаря отсутствию обратной связи, возникающий эффект распространения ошибок ограничен некоторой конечной глубиной.

Корректирующие способности порогового декодирования определяются исправляющей способностью на одном блоке кодовых символов. Следовательно, исследование пороговых методов декодирования можно ограничить изучением влияяния структуры синдромов и проверочной матрицы кода на декодирование первого блока кодовых символов.

Самыми простыми среди сверточных кодов, допускающих пороговое декодирование, являются самоортогональные коды. Это коды, допускающие полную ортогонализацию, т.е. коды, которые на блоке из n^0 символов могут быть мажоритарно декодированы. Метод построения таких кодов основан на использовании совершенных разностных множеств.

Скорость кодирования их равна $R = (n^0 – 1)/n0$ и $R = 1/n^0$. К сожалению, иных конструктивных способов построения самоортогональных кодов не известно.

Еще одним примером сверточных кодов, допускающих пороговое декодирование, являются ортогонализируемые коды – коды, которые допускают построение для каждого кодового символа составных проверок из линейной комбинации символов синдромов. По сравнению с самоортогональными эти коды могут иметь меньшее кодовое ограничение и в этом смысле имеют более высокие конструктивные параметры. Однако ортогонализируемые коды строятся методом перебора и могут иметь бесконечную глубину распространения ошибок.

Декодирование по максимуму правдоподобия

Верхние границы вероятности ошибки при декодировании сверточных кодов по максимуму правдоподобия лучше, чем у блоковых кодов той же длины. При этом используется алгоритм декодирования Витерби, который, реализует декодирование по максимуму правдоподобия.

Декодер Витерби итеративно обрабатывает кадр за кадром и, двигаясь по решетке, пытается повторить путь кодера. Решеткой называется граф, узлы которого находятся в прямоугольной координатной сетке, полубесконечной справа, число узлов в каждом столбце конечно. Конфигурация ребер, соединяющих узлы каждого столбца с узлами столбца справа, одинакова для всех столбцов. Типичная решетка для двоичного кодового алфавита приведена на рис. 4.2.

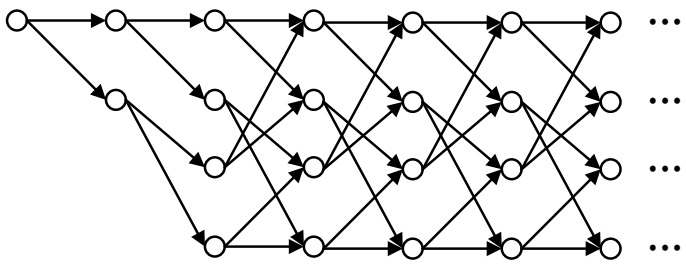


Рис. 4.2. Решетка сверточного кода

Узлы в каждом столбце представляют q^v состояний, в которых может находиться регистр сдвига. Каждый следующий столбец представляет собой набор состояний в следующий момент времени. Поступление на вход нового кадра приводит к изменению состояния регистра, соответствующего ребру, которое ведет к следующему узлу.

В любой момент времени декодер не знает, в каком узле находится кодер, и поэтому не пытается декодировать этот узел. Вместо этого декодер по принятой последовательности определяет наиболее правдоподобный путь к каждому узлу и определяет расстояние между каждым таким путем и принятой последовательностью. Это расстояние называют мерой расходимости пути. Если все пути в множестве наиболее правдоподобных начинаются одинаково, то декодер, как правило, знает начало пути, пройденного кодером.

В следующем кадре декодер определяет наиболее правдоподобный путь к каждому из новых узлов этого кадра. Наиболее правдоподобный путь находится прибавлением приращения меры расходимости на продолжениях старых путей к мере расходимости путей, ведущих в старый узел. В каждый новый узел ведет q^v путей, и путь с наименьшей мерой расходимости является наиболее правдоподобным путем. Этот процесс повторяется для каждого из новых узлов. В конце итерации декодер знает наиболее правдоподобный путь к каждому из узлов в новом кадре. Если все выжившие пути проходят через один и тот же узел в первом временном кадре, то вне зависимости от того в каком узле кодер находится в другом временном кадре, становится известным наиболее прадоподобный первый временной кадр. Иначе говоря, принимается решение о первом временном кадре.

Для построения декодера Витерби необходимо выбрать ширину окна декодирования, которая обычно в несколько раз превосходит длину кодового блока. На рис. 4.3 представлено окно декодирования Витерби с изображенной частью кодовой решетки и выжившими путями.

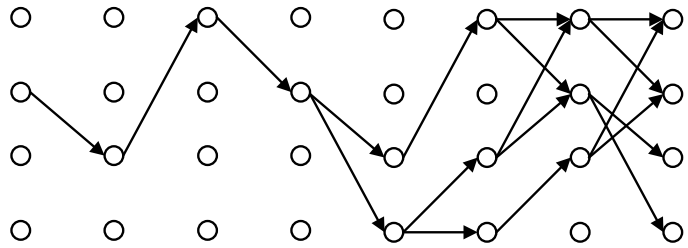


Рис. 4.3. Окно декодирования Витерби

По мере продвижения декодера к последующим кадрам из его памяти выводятся ранние кадры. Если в самом старом кадре существует лишь один узел, через который проходит путь, то декодирование является полным. В случае, когда узлов несколько, декодер неполный. Избежать этого можно, увеличив ширину окна декодирования. К сожалению, сложность декодера Витерби быстро растет. Действительно, для сверточного кода с длиной кодового ограничения v необходимо хранить в q^v путей, что для больших v становится совершенно непригодно.

Последовательное декодирование

Для того чтобы ослабить влияние больших v на сложность декодирования сверточных кодов, исследованы последовательные процедуры поиска по дереву.

Общая стратегия последовательного декодирования состоит в игнорировании маловероятных путей по решетке. В отличие от оптимальной процедуры Витерби последовательный декодер просматривает только один узел и по определенному правилу принимает решение о продвижении вперед на одно ребро графа. Если принято неправильное решение, то декодер возвращается и начинает последовательный поиск ребер заново. Наиболее популярный алгоритм последовательного декодирования - декодер Фано.

Исходными данными для алгоритма Фано является вероятность P_0 появления ошибочного символа в канале (или, по крайней мере, верхняя граница для P_0). Если декодер следует по правильному пути, вероятное число ошибок в первых l кадрах $\approx P_0 n_0 l$.

Выберем параметр P^* так, что $P_0 < P^* < 1/2$ и введем следующий показатель:

$$t(l) = P^* n_0 l - d(l),$$

где $d(l)$ – расстояние Хемминга между принятым словом и текущим путем по решетке.

Для правильного пути $d(l) \approx P_0 n_0 l$, следовательно $t(l)$ возрастает. Пока $t(l)$ возрастает, декодер следует по правильному пути. Если $t(l)$ начинает уменьшаться, то декодер принимает решение об ошибочности выбранного пути, возвращается назад и начинает последовательно двигаться по решетке далее. Иногда последовательный декодер выполняет так много вычислений, что величина входного буфера становится недостаточной. Это явление называется *переполнением буфера* и является существенным ограничением для применения алгоритма Фано. Вероятность переполнения буфера с ростом его размера уменьшается очень медленно. Наиболее надежным способом управления переполнением буфера является периодическая подача в кодер заранее известной последовательности (например, нулей) с длиной, равной длине кодового ограничения. Если буфер переполнился, то декодер считает декодирование неудавшимся, ждет соответствующего момента и снова начинает декодирование. Все данные, поступившие между моментом переполнения буфера и следующим включением, теряются. Такой подход несколько уменьшает скорость кода и ставит при конструировании декодеров задачу синхронизации по времени.

Таким образом, существующие методы декодирования сверточных кодов позволяют эффективно решать задачу исправления ошибок в бесконечном кодовом слове непрерывного кода. Однако им присущи следующие недостатки:

- для применения порогового декодирования сверточный код должен обладать дополнительной алгебраической структурой (самоортогональность или ортогонализуемость);
- сложность реализации декодера по максимуму правдоподобия растет экспоненциально в зависимости от длины кодового ограничения *v*, что совершенно непригодно для больших *v*;
- переполнение буфера является существенным ограничением для использования последовательных декодеров, в том числе алгоритма Фано.

4.2. Разработка и исследование алгебраического метода декодирования сверточных кодов

В то же время, как показано ниже, алгебраические сверточные коды, заданные через порождающий многочлен циклического кода, обладают дополнительными алгебраическими свойствами, что существенно упрощает их декодирование.

Сверточный код по определению состоит из бесконечного числа бесконечно длинных кодовых слов. Он линеен, следовательно, может быть задан бесконечной порождающей матрицей.

Предположим, что нерекурсивный сверточный код над *GF(q)* в несистематическом виде задан порождающими многочленами вида

$$P_1(x) = p_{1,r-1}x^{r-1} + p_{1,r-2}x^{r-2} + ... + p_{1,1}x + p_{1,0};$$

$$P_2(x) = p_{2,r-1}x^{r-1} + p_{2,r-2}x^{r-2} + ... + p_{2,1}x + p_{2,0};$$

…

$$P_m(x) = p_{m,r-1}x^{r-1} + p_{m,r-2}x^{r-2} + ... + p_{m,1}x + p_{m,0},$$

где коэффициенты при *x* являются элементами *GF(q)*, $n^0 = m$.

Тогда соответствующая полубесконечная порождающая матрица запишется в виде

$$G = \begin{vmatrix} G_0 & G_1 & G_2 & \dots & G_r & 0 & 0 & \dots & 0 & \dots \\ 0 & G_0 & G_1 & \dots & G_{r-1} & G_r & 0 & \dots & 0 & \dots \\ 0 & 0 & G_0 & \dots & G_{r-2} & G_{r-1} & G_r & \dots & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & G_0 & G_1 & G_2 & \dots & G_r & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{vmatrix}, \qquad (4.1)$$

где G_i – матрица-строка, состоящая из коэффициентов порождающих многочленов сверточного кода при x^i, т.е.

$$G_i = (p_{1,i}, p_{2,i}, \dots, p_{m,i}). \qquad (4.2)$$

Символом *0* в (4.1) обозначена матрица – строка, состоящая из n^0 нулевых символов из *GF(q)*.

В случае систематического сверточного кода

$$G_0 = (1, p_{2,0}, \dots, p_{m,0}) = (1, P_0)$$

и

$$G_i = (0, p_{2,i}, \dots, p_{m,i}) = (0, P_i).$$

Тогда матрица (4.1) преобразуется в вид

$$G = \begin{vmatrix} 1 & P_0 & 0 & P_1 & 0 & P_2 & 0 & \dots & 0 & P_r & 0 & 0 & 0 & \dots \\ 0 & 0 & 1 & P_0 & 0 & P_1 & 0 & \dots & 0 & P_{r-1} & 0 & P_r & 0 & \dots \\ 0 & 0 & 0 & 0 & 1 & P_0 & 0 & \dots & 0 & P_{r-2} & 0 & P_{r-1} & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 1 & P_0 & 0 & P_1 & 0 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{vmatrix}.$$

Соответствующую полубесконечную проверочную матрицу запишем в виде

$$H = \begin{vmatrix} P_0^T & -1 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ P_1^T & 0 & P_0^T & -1 & 0 & 0 & ... & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ P_2^T & 0 & P_1^T & 0 & P_0^T & -1 & ... & 0 & 0 & 0 & 0 & 0 & 0 & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ P_r^T & 0 & P_{r-1}^T & 0 & P_{r-2}^T & 0 & ... & 0 & P_0^T & -1 & 0 & 0 & 0 & ... \\ 0 & 0 & P_r^T & 0 & P_{r-1}^T & 0 & ... & 0 & P_1^T & 0 & P_0^T & -1 & 0 & ... \\ 0 & 0 & 0 & 0 & P_r^T & 0 & ... & 0 & P_2^T & 0 & P_1^T & 0 & P_0^T & ... \\ 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & P_3^T & 0 & P_2^T & 0 & P_1^T & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \end{vmatrix}.$$

Воспользуемся введенным выше (разд. 2 – 3) алгебраическим описанием нерекурсивных сверочных кодов. Сопоставим каждую подматрицу G_i элементу поля $\beta_i \in GF(q^m)$, так, например, что

$$\beta_i = p_{1,i} + p_{2,i}x + \ldots + p_{m,i}x^m.$$

Тогда (4.2) перепишем в виде

$$G_i = (p_{1,i}, p_{2,i}, \ldots, p_{m,i}) = \beta_i,$$

а полубесконечную матрицу (4.1) представим в виде соответствующей матрицы с элементами из $GF(q^m)$:

$$G = \begin{vmatrix} \beta_0 & \beta_1 & \beta_2 & ... & \beta_r & 0 & 0 & ... & 0 & ... \\ 0 & \beta_0 & \beta_1 & ... & \beta_{r-1} & \beta_r & 0 & ... & 0 & ... \\ 0 & 0 & \beta_0 & ... & \beta_{r-2} & \beta_{r-1} & \beta_r & ... & 0 & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & \beta_0 & \beta_1 & \beta_2 & ... & \beta_r & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \end{vmatrix}. \qquad (4.3)$$

В полиномиально–матричном представлении последнее выражение перепишем в виде матрицы многочленов $G(x)$:

$$G(x) = \begin{vmatrix} P(x) \\ x \cdot P(x) \\ x^2 \cdot P(x) \\ ... \\ x^r \cdot P(x) \\ ... \end{vmatrix}, \qquad (4.4)$$

где по теореме 2.1 многочлен

$$P(x) = \beta_r x^r + \beta_{r-1} x^{r-1} + ... + \beta_1 x + \beta_0 \qquad (4.5)$$

является порождающим многочленом недвоичного *(N, K, D)* циклического кода над $GF(q^m)$, который однозначно задает *(n, k)* несистематический сверточный код над *GF(q)* с параметрами

$k^0 = 1$; $n^0 = m$; $\nu = r \cdot k^0 = r$; $k = r + 1$; $n = (r + 1) \cdot n^0 = k \cdot m$; $R = 1 / m$, $d_\infty \geq D$,

$$C(x) = I(x) \cdot P(x). \qquad (4.6)$$

Тогда подматрица

$$\|G\|_{K,N} = \begin{vmatrix} \beta_0 & \beta_1 & \beta_2 & ... & \beta_r & 0 & 0 & ... & 0 \\ 0 & \beta_0 & \beta_1 & ... & \beta_{r-1} & \beta_r & 0 & ... & 0 \\ 0 & 0 & \beta_0 & ... & \beta_{r-2} & \beta_{r-1} & \beta_r & ... & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & \beta_0 & \beta_1 & \beta_2 & ... & \beta_r \end{vmatrix} \qquad (4.7)$$

является порождающей матрицей *(N, K, D)* циклического кода над $GF(q^m)$, что в полиномиально-матричном представлении запишется в виде

$$\|G(x)\|_K = \begin{vmatrix} P(x) \\ x \cdot P(x) \\ x^2 \cdot P(x) \\ ... \\ x^{K-1} \cdot P(x) \end{vmatrix}. \qquad (4.8)$$

Введенное обобщение *(N, K, D)* циклического кода на непрерывный случай позволяет использовать свойства колец многочленов при описании соответствующих сверточных кодов.

Пусть $I(x) = i_0 + i_1 x + i_2 x^2 + ...$ – информационный многочлен, возможно бесконечной длины, с коэффициентами из *GF(q)*. Предположим, что *I(x)* поступает на вход несистематического кодера нерекурсивного сверточного *(n, k, d)* кода, алгебраически заданного через порождающий многочлен *P(x)* циклического *(N, K, D)* кода над $GF(q^m)$. Тогда кодовое слово сверточного кода

является обобщением на непрерывный случай кодового слова циклического кода, ограниченного на подполе $GF(q)$. Кодовая последовательность на выходе сверточного кодера будет задаваться выражением

$$C(x) = I(x) \cdot P(x) = C_0 + C_1 x + C_2 x^2 + \ldots, \qquad (4.9)$$

где C_i – элементы поля $GF(q^m)$, отображаемые в наборы по m символов из подполя $GF(q)$.

В матричной форме последнее выражение принимает вид

$$C = I \cdot G, \qquad (4.10)$$

где $C = (C_0, C_1, C_2, \ldots)$, $I = (i_0, i_1, i_2, \ldots)$ – кодовый и информационный векторы, составленные из коэффициентов соответствующих многочленов.

Рассмотрим правило формирования коэффициентов кодового многочлена (4.12), аналитически свяжем значение каждого кодового символа с информационными символами, поступающими на вход кодера.

Разобьем информационный вектор I на блоки по K символов из $GF(q)$:

$$I = (i_0, i_1, i_2, \ldots, i_{K-1}) \cup (i_K, i_{K+1}, i_{K+2}, \ldots, i_{2K-1}) \cup (i_{2K}, i_{2K+1}, i_{2K+2}, \ldots, i_{3K-1}) \cup \ldots. \qquad (4.11)$$

Обозначим каждый блок из K символов через I_i:

$$I = I_0 \cup I_1 \cup I_2 \cup \ldots. \qquad (4.12)$$

В полиномиальном виде последнее выражение эквивалентно следующему:

$$I(x) = I_0(x) + x^K I_1(x) + x^{2K} I_2(x) + \ldots, \qquad (4.13)$$

где

$$I_i(x) = i_{i \cdot K} + i_{i \cdot K+1} x + i_{i \cdot K+2} x^2 + \ldots i_{(i+1) \cdot K-1} x^{K-1}.$$

Подставим (4.13) в (4.10), получим

$$C(x) = I(x) \cdot P(x) = \left(I_0(x) + x^K I_1(x) + x^{2K} I_2(x) + ...\right) \cdot P(x) =$$
$$I_0(x) \cdot P(x) + x^K I_1(x) \cdot P(x) + x^{2K} I_2(x) \cdot P(x) + ... = \sum_{i=0}^{\infty} x^{i \cdot K} I_i(x) \cdot P(x), \quad (4.14)$$

или в матричном виде

$$C = \sum_{i=0}^{\infty} \|I_i\|_K \cdot \|G\|_{K,N} \cdot \|0,I\|_{N,i \cdot K+N}, \quad (4.15)$$

где $\|0,I\|_{N,i \cdot K+N}$ – единичная матрица с добавленными слева $i \cdot K$ нулевыми столбцами.

Проанализируем полученное выражение (4.15). Каждое слагаемое содержит произведение порождающего многочлена циклического *(N, K, D)* кода на информационный многочлен $I_i(x)$ степени $deg\ I_i(x) \le K - 1$. Однако произведение $I_i(x) \cdot P(x)$ – суть кодовое слово циклического *(N, K, D)* кода, которое соответствует информационному вектору

$$I_i = (i_{i \cdot K}, i_{i \cdot K+1}, i_{i \cdot K+2}, \ldots, i_{(i+1) \cdot K-1}),$$

т.е.

$$I_i(x) \cdot P(x) = c_i(x), \quad (4.16)$$

где

$$c_i(x) = c_{i,0} + c_{i,1} x + c_{i,2} x^2 + \ldots c_{i,N-1} x^{N-1}.$$

Подставив (4.16) в (4.15), получим

$$C(x) = I(x) \cdot P(x) = c_0(x) + x^K c_1(x) + x^{2K} c_2(x) + ... = \sum_{i=0}^{\infty} x^{i \cdot K} c_i(x) \quad (4.17)$$

или в матричном виде

$$C = \sum_{i=0}^{\infty} \|c_i\|_N \cdot \|0,I\|_{N,i \cdot K+N}. \quad (4.18)$$

Таким образом, как следует из выражения (4.17), бесконечное кодовое слово нерекурсивного сверточного кода, алгебраически заданного через порождающий многочлен циклического кода, состоит из бесконечной суммы кодовых слов циклического кода, умноженных на соответствующий оператор задержки $x^{i \cdot K}$. Схематично структура бесконечного кодового слова алгебраического сверточного кода представлена на рис. 4.4.

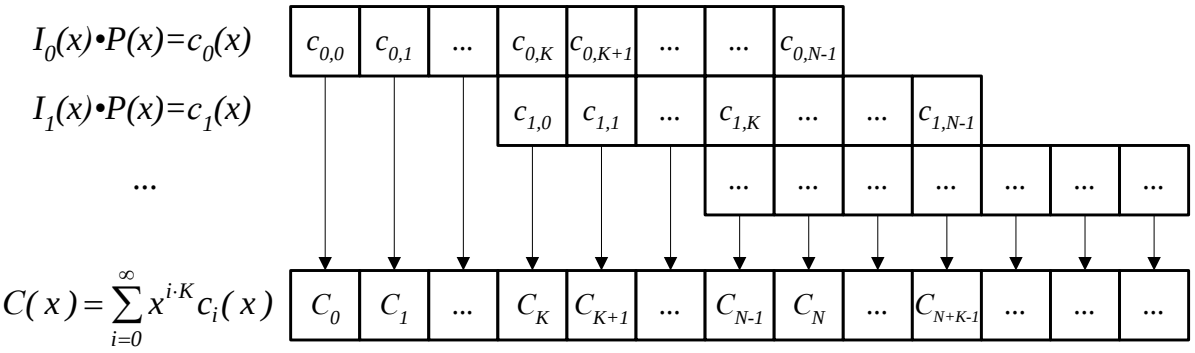


Рис. 4.4. Схематичная структура бесконечного кодового слова алгебраического нерекурсивного сверточного кода

Как видно из рис. 4.4, бесконечное кодовое слово сверточного кода формируется наложением бесконечного числа кодовых слов циклического кода и суммированием соответствующих элементов $c_{i,j}$.

Предположим теперь, что при передаче бесконечной кодовой последовательности вектор $C = (C_0, C_1, \ldots)$ исказился, т.е. на приемной стороне получено искаженное кодовое слово

$$C^*(x) = C(x) + E(x), \qquad (4.19)$$

где $E(x) = e_0 + e_1 x + e_2 x^2 + \ldots$ – бесконечный вектор ошибок.

По аналогии с информационным вектором разобьем вектор ошибок $E = (e_0, e_1, e_2, \ldots)$, составленный из коэффициентов многочлена ошибок $E(x)$, на блоки по K символов из $GF(q)$:

$$E = (e_0, e_1, e_2, \ldots, e_{K-1}) \cup (e_K, e_{K+1}, e_{K+2}, \ldots, e_{2K-1}) \cup \ldots.$$

Обозначим каждый блок из K символов через E_i:

$$E = E_0 \cup E_1 \cup E_2 \cup \ldots.$$

В полиномиальном виде последнее выражение эквивалентно следующему:

$$E(x) = E_0(x) + x^K E_1(x) + x^{2K} E_2(x) + \ldots, \qquad (4.20)$$

где $E_i(x) = e_{i\cdot K} + e_{i\cdot K+1}\, x + e_{i\cdot K+2}\, x^2 + \ldots\ e_{(i+1)\cdot K-1}\, x^{K-1}$.

Подставив (4.20) в (4.19), получим

$$C^*(x) = C(x) + E(x) = \left(I_0(x)\cdot P(x) + E_0(x)\right) + x^K\left(I_1(x)\cdot P(x) + E_1(x)\right) +$$
$$+ x^{2K}\left(I_2(x)\cdot P(x) + E_2(x)\right) + \ldots = \sum_{i=0}^{\infty} x^{i\cdot K}\left(I_i(x)\cdot P(x) + E_i(x)\right).$$

С учетом (4.17) последнее выражение перепишем в виде

$$C^*(x) = C(x) + E(x) = \left(c_0(x) + E_0(x)\right) + x^K\left(c_1(x) + E_1(x)\right) +$$
$$+ x^{2K}\left(c_2(x) + E_2(x)\right) + \ldots \sum_{i=0}^{\infty} x^{i\cdot K}\left(c_i(x) + E_i(x)\right), \qquad (4.21)$$

или в матричной форме

$$C^* = \sum_{i=0}^{\infty}\left(\|c_i\|_N + \|e_i, 0\|_N\right)\cdot\|0, I\|_{N, i\cdot K+N}, \qquad (4.22)$$

где $\|e_i, 0\|_N$ – вектор ошибок E_i длины K символов с добавленными справа $(N - K)$ нулями.

Проанализируем полученное выражение. Каждое слагаемое содержит сумму кодового слова $c_i(x)$ циклического (N, K, D) кода и многочлена ошибки $E_i(x)$. Размерность вектора E_i составляет K символов, т.е. сумма $c_i(x) + E_i(x)$ – суть кодовое слово циклического (N, K, D) кода, искаженное вектором ошибки E_i. Следовательно, можно утверждать, что

$$c_i^*(x) = c_i(x) + E_i(x). \qquad (4.23)$$

Тогда, с учетом (4.23), выражение (4.22) перепишем в виде

$$C^*(x) = C(x) + E(x) = c_0^*(x) + x^K c_1^*(x) + x^{2K} c_2^*(x) + \ldots = \sum_{i=0}^{\infty} x^{i \cdot K} c_i^*(x), \qquad (4.24)$$

или соответственно в матричной форме

$$C^* = \sum_{i=0}^{\infty} \left\| c^*_i \right\|_N \cdot \left\| 0, I \right\|_{N, i \cdot K + N} . \qquad (4.25)$$

Таким образом, как следует из выражения (4.24), бесконечное кодовое слово алгебраического нерекурсивного сверточного кода, искаженное бесконечным вектором ошибок, состоит из бесконечной суммы кодовых слов циклического кода, искаженных вектором ошибок конечной размерности и умноженных на соответствующий оператор задержки $x^{i \cdot K}$.

Представим для наглядности схематично структуру искаженного ошибками бесконечного кодового слова алгебраического сверточного кода, как показано на рис. 4.5.

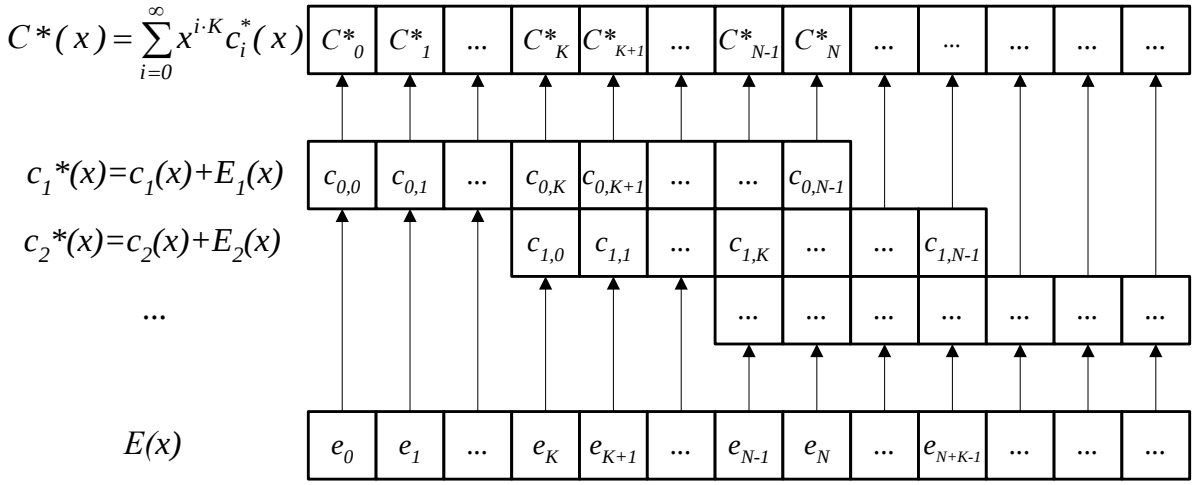


Рис. 4.5. Структура искаженного ошибками бесконечного кодового слова алгебраического нерекурсивного сверточного кода

Как видно из рис. 4.5, искаженное ошибками бесконечное кодовое слово сверточного кода формируется наложением бесконечного числа искаженных кодовых слов циклического кода и суммированием соответствующих элементов $c^*_{i,j}$.

Введем понятие „синдромный многочлен алгебраического сверточного кода”

$$S(x) = s_0 + s_1 x + s_2 x^2 + \ldots \qquad (4.26)$$

как бесконечную сумму синдромных многочленов циклического кода, умноженных на соответствующий оператор задержки $x^{i \cdot K}$, т.е. как бесконечную сумму остатков от деления кодовых слов циклического кода на порождающий многочлен $P(x)$

$$S(x) = \sum_{i=0}^{\infty} x^{i \cdot (N-K)} R_{P(x)}\left[c_i^*(x)\right]. \qquad (4.27)$$

В кольце многочленов $GF(q^m)[x]/(x^N - 1)$ существует единственный приведенный ненулевой многочлен $h(x)$

$$h(x) = \gamma_K x^K + \gamma_{K-1} x^{K-1} + \ldots + \gamma_1 x + \gamma_0$$

наименьшей степени K, который обозначается проверочным многочленом и также однозначно задает (N, K, D) циклический код над $GF(q^m)$. Соответствующая проверочная матрица (N, K, D) циклического кода может быть записана в виде

$$\|H\|_{N-K,N} = \begin{vmatrix} \gamma_K & \ldots & \gamma_2 & \gamma_1 & \gamma_0 & \ldots & 0 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & \gamma_K & \gamma_{K-1} & \gamma_{K-2} & \ldots & \gamma_0 & 0 & 0 \\ 0 & \ldots & 0 & \gamma_K & \gamma_{K-1} & \ldots & \gamma_1 & \gamma_0 & 0 \\ 0 & \ldots & 0 & 0 & \gamma_K & \ldots & \gamma_2 & \gamma_1 & \gamma_0 \end{vmatrix},$$

или в полиномиально-матричном обозначении

$$H(x) = \begin{vmatrix} h(x) \\ x \cdot h(x) \\ x^2 \cdot h(x) \\ \ldots \\ x^{r-1} \cdot h(x) \end{vmatrix},$$

где нумерация коэффициентов многочлена идет в порядке, обратном принятому в $G(x)$.

Воспользовавшись мультипликативно обратным многочлену $P(x)$ в кольце $GF(q^m)[x]/(x^N - 1)$ элементом – проверочным

многочленом *h(x)* циклического *(N, K, D)*–кода, выразим выражение (4.27) в виде

$$S(x)=\sum_{i=0}^{\infty}x^{i\cdot(N-K)}R_{(x^N-1)}\left[c_i^*(x)\cdot h(x)\right], \tag{4.28}$$

что в матричном виде эквивалентно следующему:

$$S=\sum_{i=0}^{\infty}\left\|c*_i\right\|_N\cdot\left\|H\right\|_{N-K,N}^{T}\cdot\left\|0,I\right\|_{N-K,i\cdot(N-K)+N-K}. \tag{4.29}$$

С учетом (4.24) выражение (4.29) запишем в виде

$$\begin{gathered}S(x)=\sum_{i=0}^{\infty}x^{i\cdot(N-K)}R_{P(x)}\left[c_i(x)+E_i(x)\right]=\\ =\sum_{i=0}^{\infty}x^{i\cdot(N-K)}\left(R_{P(x)}\left[c_i(x)\right]+R_{P(x)}\left[E_i(x)\right]\right)=\sum_{i=0}^{\infty}x^{i\cdot(N-K)}R_{P(x)}\left[E_i(x)\right].\end{gathered} \tag{4.30}$$

Перепишем через проверочный многочлен

$$S(x)=\sum_{i=0}^{\infty}x^{i\cdot(N-K)}R_{(x^N-1)}\left[E_i(x)\cdot h(x)\right], \tag{4.31}$$

что в матричной форме примет следующий вид:

$$S=\sum_{i=0}^{\infty}\left\|e_i,0\right\|_N\cdot\left\|H\right\|_{N-K,N}^{T}\cdot\left\|0,I\right\|_{N-K,i\cdot(N-K)+N-K}. \tag{4.32}$$

Таким образом, как следует из выражения (4.30), бесконечный синдром принятого с ошибками кодового слова алгебраического нерекурсивного сверточного кода состоит из бесконечной суммы синдромов принятых кодовых слов циклического кода, умноженных на соответствующий оператор задержки $x^{i\cdot(N-K)}$.

Следовательно, получаем

$$S(x)=\sum_{i=0}^{\infty}x^{i\cdot(N-K)}S_i(x), \tag{4.33}$$

или

$$S = \sum_{i=0}^{\infty} \|S_i\|_{N-K} \cdot \|0, I\|_{N-K, i\cdot(N-K)+N-K}\,, \qquad (4.34)$$

где $S_i(x) = s_{i\cdot K} + s_{i\cdot K+1}\,x + s_{i\cdot K+2}\,x^2 + \ldots\, s_{(i+1)\cdot K-1}\,x^{K-1}$ – синдромный многочлен циклического *(N, K, D)* кода,

$S_i = (s_{i\cdot K}, s_{i\cdot K+1}, s_{i\cdot K+2}, \ldots\ s_{(i+1)\cdot K-1})$ – соответствующий синдромный вектор.

Значение синдромного многочлена (вектора) зависит только от величины ошибок и не зависит от выбранного кодового слова. Представим схематично структуру и правило формирования синдромного многочлена, как показано на рис. 4.6.

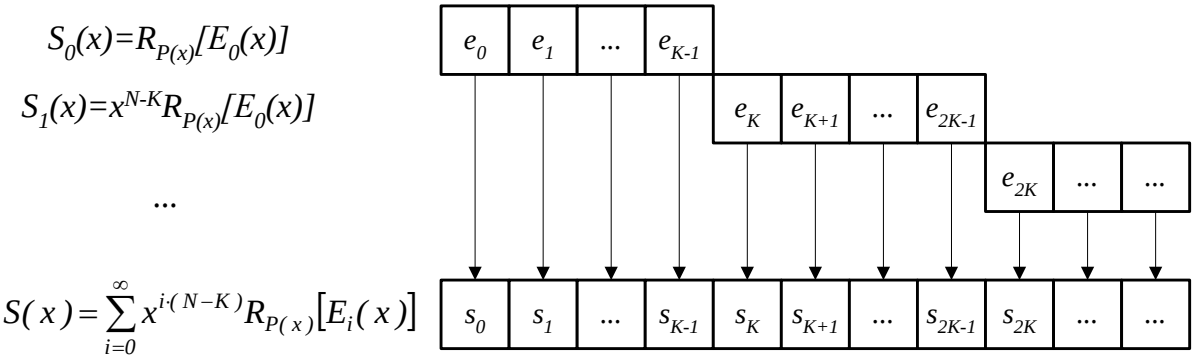


Рис. 4.6. Схематичная структура бесконечного синдромного многочлена алгебраического нерекурсивного сверточного кода

Как видно из рис. 4.6, бесконечный синдром формируется бесконечным суммированием соответствующих синдромов циклического кода $S_i(x)$. Причем синдромы $S_i(x)$ суммируются без наложений, т.е. каждый блок из $(N-K)$ синдромных символов зависит исключительно от блока из K ошибочных символов.

Этот факт позволяет реализовать алгебраическое правило декодирования алгебраически заданного сверточного кода.

Действительно, декодирование бесконечного кодового слова сверточного кода распадается на бесконечную последовательность декодирований кодовых слов циклического *(N, K, D)* кода. Причем каждый синдромный вектор S_i соответствует ошибке, произошедшей на блоке из K символов, и в случае неправильного декодирования ошибка распространяется

только в пределах блока данных из K символов. Следовательно, независимость блоков синдромных символов позволяет избежать распространения ошибок, которое присуще некоторым известным способам декодирования сверточных кодов.

Таким образом, в результате проведенных рассуждений удалось свести декодирование бесконечного кодового слова к бесконечной серии декодирований циклического блокового кода.

Рассмотрим теперь декодирование одного блока символов бесконечного сверточного кода, а затем обобщим его на случай бесконечных серий.

Проанализируем выражение (4.32). Оно содержит бесконечную сумму произведений непересекающихся ненулевых векторов ошибок на проверочную $\|H\|_{N-K,N}$ матрицу циклического (N, K, D) кода, заданного через порождающий многочлен $P(x)$. Очевидно, что матрица $\|H\|^T_{N-K,N}$ может быть записана в виде (4.5), но для декодирования циклических кодов используется другая ее форма, которая отражает структуру колец многочленов и непосредственно свойства самого многочлена $P(x)$.

Обозначим через X_l – l – ый корень порождающего многочлена $P(x)$, причем $X_l = \alpha^{Jl}$ для некоторого J_l, $X_l \in GF(q^m)$. Если $X_0, X_1, \ldots, X_{r-1}$ – все корни многочлена $P(x)$, т.е. $P(x) = (x + X_0)\cdot(x + X_1) \cdot \ldots \cdot (x + X_{r-1})$, то справедливо равенство

$$c(X_i) = c_0 + c_1 X_i + c_2 X_i^2 + \ldots + c_{N-1} X_i^{N-1} = 0,$$

где $c(x)$ кодовый многочлен циклического (N, K, D) кода.

Перепишем последнее выражение в виде матричного произведения

$$c(X_i) = (c_0, c_1, c_2, \ldots, c_{N-1}) \cdot (1, X_i, X_i^2, \ldots, X_i^{N-1})^T = 0,$$

где c кодовое слово циклического (N, K, D) кода как набор коэффициентов многочлена $c(x)$.

Обобщим последнее равенство для всех корней $P(x)$, получим

$$(c_0, c_1, c_2, ..., c_{N-1}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & ... & X_0^{N-1} \\ 1 & X_1 & X_1^2 & ... & X_1^{N-1} \\ 1 & X_2 & X_2^2 & ... & X_2^{N-1} \\ ... & ... & ... & ... & ... \\ 1 & X_{r-1} & X_{r-1}^2 & ... & X_{r-1}^{N-1} \end{vmatrix}^T = 0.$$

Это выражение соответствует условию взаимной ортогональности произвольного кодового слова $c = (c_0, c_1, c_2, \ldots, c_{N-1})$ и матрицы в правой части произведения. Следовательно, положим

$$\|H\|_{N-K,N} = \begin{vmatrix} 1 & X_0 & X_0^2 & ... & X_0^{N-1} \\ 1 & X_1 & X_1^2 & ... & X_1^{N-1} \\ 1 & X_2 & X_2^2 & ... & X_2^{N-1} \\ ... & ... & ... & ... & ... \\ 1 & X_{r-1} & X_{r-1}^2 & ... & X_{r-1}^{N-1} \end{vmatrix}. \qquad (4.35)$$

Предположим теперь, что кодовое слово c исказилось при передаче. Пусть число ошибок на блоке из N символов не превышает исправляющей способности $t = (D - 1)/2$ циклического (N, K, D) кода. Обозначим $e(x)$ – многочлен ошибок, так, что

$$e(x) = e_0 + e_1 x + e_2 x^2 + \ldots + e_{N-1} x^{N-1}$$

с $\leq t$ ненулевыми коэффициентами.

Пусть $c^*(x) = c^*_0 + c^*_1 x + c^*_2 x^2 + \ldots + c^*_{N-1} x^{N-1}$ – кодовое слово с ошибками, т.е.

$$c^*(x) = c(x) + e(x) = (c_0 + e_0) + (c_1 + e_1)x + (c_2 + e_2)x^2 + \ldots + (c_{N-1} + e_{N-1})x^{N-1}.$$

Значение вектора синдромов вычислим из выражения

$$(s_0, s_1, ..., s_{r-1}) = (e_0, e_1, e_2, ..., e_{N-1}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & ... & X_0^{N-1} \\ 1 & X_1 & X_1^2 & ... & X_1^{N-1} \\ 1 & X_2 & X_2^2 & ... & X_2^{N-1} \\ ... & ... & ... & ... & ... \\ 1 & X_{r-1} & X_{r-1}^2 & ... & X_{r-1}^{N-1} \end{vmatrix}^T,$$

что эквивалентно следующей системе уравнений:

$$s_0 = e_0 + e_1 X_0 + e_2 X_0^2 + \ldots + e_{N-1} X_0^{N-1} = \sum_{i=0}^{N-1} e_i X_0^i ;$$

$$s_1 = e_0 + e_1 X_1 + e_2 X_1^2 + \ldots + e_{N-1} X_1^{N-1} = \sum_{i=0}^{N-1} e_i X_1^i ; \qquad (4.36)$$

$$\ldots$$

$$s_{r-1} = e_0 + e_1 X_{r-1} + e_2 X_{r-1}^2 + \ldots + e_{N-1} X_{r-1}^{N-1} = \sum_{i=0}^{N-1} e_i X_{r-1}^i .$$

Задача декодирования блока данных из N символов состоит в нахождении всех e_i, $i = 0, \ldots, N-1$ по известным элементам синдромной последовательности $(s_0, s_1, \ldots, s_{r-1})$.

Система уравнений (4.36) нелинейна, прямых методов ее решения неизвестно. Для нахождения вектора ошибок $(e_0, e_1, e_2, \ldots, e_{N-1})$ воспользуемся искусственным приемом. Введем многочлен локаторов ошибок $\Lambda(x)$, корнями которого являются ненулевые элементы вектора ошибок, т.е.

$$\Lambda(x) = \prod_j (x + X_j), \qquad (4.37)$$

где j – индекс ненулевых элементов вектора ошибок;

X_j – локатор ошибки, произошедшей в j–ом символе кодового слова.

Раскроем скобки в выражении (4.37), получим

$$\Lambda(x) = x^w + \lambda_{w-1} x^{w-1} + \ldots + \lambda_1 x + \lambda_0, \qquad (4.38)$$

где степень w многочлена $\Lambda(x)$ задает число произошедших ошибок на блоке из N символов, $w \le t$, т.е. число ненулевых элементов вектора ошибок $(e_0, e_1, e_2, \ldots, e_{N-1})$.

Набор $(\lambda_0, \lambda_1, \ldots, \lambda_{w-1})$ коэффициентов многочлена (4.38) однозначно задает его корни, которые однозначно указывают (локализуют) расположение произошедших ошибок. Умножим многочлен (4.38) на $e_i X^i$ и вычислим его значение в X_j, получим

$$e_i X_j^{w+i} + e_i \lambda_{w-1} X_j^{w+i-1} + ... + e_i \lambda_1 X_j^{i+1} + e_i \lambda_0 X_j^{i} = 0,$$

где $X_j \in GF(q^m)$, т.е. $X_j = \alpha^{Jj}$ для некоторого J_j.

Следовательно,

$$X_j^{a+b} = \alpha^{a+b+Jj} = X^b{}_{j+a},$$

т.е. справедливо выражение

$$e_i X_{j+w}^{i} + e_i \lambda_{w-1} X_{j+w-1}^{i} + ... + e_i \lambda_1 X_{j+1}^{i} + e_i \lambda_0 X_j^{i} = 0.$$

Последнее равенство выполняется для любого j и при каждом i. Просуммируем по всем $i = 0...N - 1$, получим

$$\sum_{i=0}^{N-1} \left(e_i X_{j+w}^{i} + e_i \lambda_{w-1} X_{j+w-1}^{i} + ... + e_i \lambda_1 X_{j+1}^{i} + e_i \lambda_0 X_j^{i} \right) = 0.$$

Изменим порядок суммирования, вынесем коэффициенты многочлена локаторов ошибок за знак суммирования, получим

$$\sum_{i=0}^{N-1} e_i X_{j+w}^{i} + \lambda_{w-1} \cdot \sum_{i=0}^{N-1} e_i X_{j+w-1}^{i} + ... + \lambda_1 \cdot \sum_{i=0}^{N-1} e_i X_{j+1}^{i} + \lambda_0 \cdot \sum_{i=0}^{N-1} e_i X_j^{i} = 0.$$

Значение каждого слагаемого в последнем выражении соответствует произведению коэффициентов многочлена локаторов ошибок на соответствующие синдромы в выражении (4.38), так что запишем

$$s_{j+w} + \lambda_{w-1} \cdot s_{j+w-1} + \ldots + \lambda_1 \cdot s_{j+1} + \lambda_0 \cdot s_j = 0. \qquad (4.39)$$

Перепишем выражение (4.39) для каждого $j = 0...w$, получим систему линейных уравнений

$$\begin{aligned}
&s_w + \lambda_{w-1} \cdot s_{w-1} + \ldots + \lambda_1 \cdot s_1 + \lambda_0 \cdot s_0 = 0, \\
&s_{w+1} + \lambda_{w-1} \cdot s_w + \ldots + \lambda_1 \cdot s_2 + \lambda_0 \cdot s_1 = 0, \qquad (4.40) \\
&\ldots \\
&s_{2 \cdot w} + \lambda_{w-1} \cdot s_{2 \cdot w-1} + \ldots + \lambda_1 \cdot s_{w+1} + \lambda_0 \cdot s_w = 0.
\end{aligned}$$

Система из w линейных уравнений (4.40) с w неизвестными разрешима, сложность ее решения растет полиномиально в зависимости от числа неизвестных. Так, например, для решения системы (4.40) методом Гаусса необходимо выполнить n^2 сложений и умножений над элементами $GF(q^m)$.Формально, сложность алгоритма равна $O(n^2)$.

Решение системы уравнений (4.40) дает значения коэффициентов многочлена локаторов ошибок (4.38). Корнями многочлена (4.38) являются локаторы – элементы $GF(q^m)$, которые однозначно указывают расположение ошибок. Следовательно, для локализации ошибок необходимо найти корни уравнения (4.38).

Наиболее простая процедура поиска корней многочлена локаторов ошибок состоит в подстановке всех элементов поля $GF(q^m)$ и выборе тех элементов, которые обращают в нуль многочлен (4.38). В литературе такой прием получил название процедуры Ченя.

После локализации ошибок необходимо вычислить величины ошибок в j–ом символе, т.е. вычислить вектор ошибок $(e_0, e_1, e_2, \ldots, e_{N-1})$ и восстановить кодовое слово: $c = c^* - e$.

Для нахождения значений ошибок воспользуемся выражением (4.37). Подставим значения найденных локаторов X_j и неизвестные значения e_j в систему уравнений. Остальные e_i при $i \neq j$ равны нулю. Следовательно, система уравнений (4.36) запишется в виде

$$s_0 = \sum_{i \in J} e_i X_0^i, \quad s_1 = \sum_{i \in J} e_i X_1^i, \ \ldots, \ s_{r-1} = \sum_{i \in J} e_i X_{r-1}^i, \tag{4.41}$$

где J – множество индексов ненулевых элементов вектора ошибок, т.е. набор номеров локаторов ошибок, причем $|J| = w \leq t$.

Система (4.41) из r линейных уравнений содержит $|J| = w \leq t$ неизвестных значений ошибок e_i, причем $t < r$. Следовательно, система (4.41) разрешима, ее решение дает неизвестные ненулевые значения ошибок вектора $(e_0, e_1, e_2, \ldots, e_{N-1})$. Для восстановления кодового слова длины N кодовых символов достаточно снять действие найденного вектора ошибок:

$$c = c^* - e.$$

Таким образом, для реализации предложенного подхода алгебраического декодирования бесконечных кодовых слов алгебраического сверточного кода, заданного через порождающий многочлен циклического кода, необходимо и достаточно вычислить бесконечную сумму синдромов соответствующих кодовых слов циклического кода, т.е. вычислить все значения $S_i = (s_{i\cdot K}, s_{i\cdot K+1}, s_{i\cdot K+2}, \ldots s_{(i+1)\cdot K-1})$ в выражении (4.33). Для вычисления синдромной последовательности в алгебраической теории блоковых кодов используют умножение кодового слова на проверочную матрицу и/или, что эквивалентно, формируют синдромный многочлен $S_i(x)$ через операции в кольце многочленов $GF(q)[x]/(x^n - 1)$. Эквивалентной операцией для непрерывных кодов является определение произведения кодового слова на полубесконечную проверочную матрицу сверточного кода, заданную через корни порождающего многочлена. Конструктивных способов построения полубесконечной проверочной матрицы несистематического сверточного кода с сохранением алгебраических свойств неизвестно. Следовательно, для реализации предложенного выше подхода алгебраического декодирования сверточных кодов необходимо теоретически обосновать и разработать процедуры формирования бесконечной серии синдромных последовательностей $S_i = (s_{i\cdot K}, s_{i\cdot K+1}, s_{i\cdot K+2}, \ldots s_{(i+1)\cdot K-1})$.

4.3. Разработка способа формирования бесконечной серии конечных синдромов для алгебраического декодирования сверточных кодов

Рассмотрим структуру бесконечного кодового слова алгебраического сверточного кода, схематично представленную на рис. 4.4. Если алгебраический сверточный код задан через порождающий многочлен (N, K, D) циклического кода, то бесконечное кодовое слово формируется суммированием бесконечного числа кодовых слов циклического кода, сдвинутых на K символов вправо. Следовательно, i–ый блок из N кодовых символов бесконечного кодового слова состоит из суммы i–ого

кодового слова циклического кода и соответствующих частей $i+j$–ых кодовых слов. Схематично структура произвольного блока из N символов бесконечного кодового слова алгебраического нерекурсивного сверточного кода представлена на рис. 4.7.

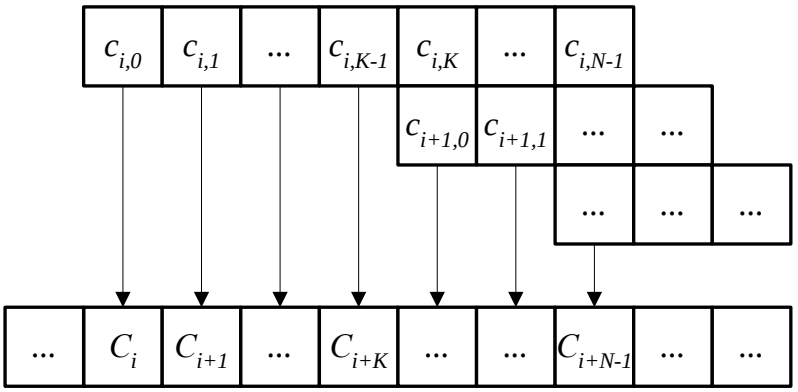


Рис. 4.7. Структура произвольного блока из N символов бесконечного кодового слова алгебраического нерекурсивного сверточного кода

Формально можем записать:

$$\|C\|_N = \|c_i\|_N + \|c_{i+1}\|_{N-K} + \|c_{i+2}\|_{N-2\cdot K} + \|c_{i+3}\|_{N-3\cdot K} + \ldots, \qquad (4.32)$$

где $\|C\|_N$ – блок блока из N символов бесконечного кодового слова алгебраического сверточного кода;

$\|c_i\|_N$ – i–ое кодовое слово циклического кода;

$\|c_{i+j}\|_{N-j\cdot K}$ – соответствующие подблоки $i+j$–ых кодовых слов циклического кода.

Предположим, что на рассматриваемом блоке из N символов произошло не более $t = (D-1)/2$ ошибок. Сформируем из блока $\|C\|_N$ кодовый многочлен и вычислим остаток от деления его на порождающий многочлен циклического кода. Последняя операция эквивалентна умножению на проверочный многочлен и/или нахождению блока $\|C\|_N$ и проверочной матрицы $\|H\|_{N-K,N}$ циклического кода.

В результате получим синдромный вектор S*:

$$S^*(x) = R_{g(x)}\left[c_i(x) + e_i(x) + Z(x)\right] = \\ = R_{g(x)}\left[c_i(x)\right] + R_{g(x)}\left[e_i(x)\right] + R_{g(x)}\left[Z(x)\right]. \quad (4.33)$$

где $e_i(x)$ – многочлен ошибок, коэффициентами которого являются элементы вектора ошибок длины N символов, т.е. вектор ошибки на i–ом кодовом слове циклического кода;

$Z(x)$ – сумма многочленов, коэффициентами которых являются элементы векторов $\left\|c_{i+j}\right\|_{N-j\cdot K}$ в выражении (4.32), при $j \neq 0$.

Очевидно, что первое слагаемое в выражении (4.33) суть остаток от деления кодового слова циклического кода на соответствующий порождающий многочлен. Следовательно, $R_{g(x)}\left[c_i(x)\right] = 0$.

Второе слагаемое в выражении (4.33) соответствует остатку от деления многочлена ошибок кодового слова циклического кода на его порождающий многочлен. То есть в веденных ранее обозначениях $R_{g(x)}\left[e_i(x)\right] = S_i(x)$.

Третье слагаемое соответствует остатку от деления суммы многочленов из выражения (4.32) на порождающий многочлен циклического кода. Отметим, что все многочлены, коэффициентами которых являются элементы векторов $\left\|c_{i+j}\right\|_{N-j\cdot K}$ в выражении (4.32), при $j \neq 0$, имеют степень $\leq N - K$, а степень порождающего многочлена $\deg g(x) = N - K + 1$. Следовательно, запишем $R_{g(x)}\left[Z(x)\right] = Z(x)$. Тогда выражение (4.33) запишется в виде

$$S^*(x) = S_i(x) + Z(x). \quad (4.34)$$

Очевидно, что при $Z(x) = 0$ выполняется равенство $S^*(x) = S_i(x)$. Практически это означает, что при выполнении равенства нулю суммы векторов $\left\|c_{i+j}\right\|_{N-j\cdot K}$ в выражении (4.32), $j \neq 0$ значения синдромов S^* для блока из N кодовых символов совпадут с соответствующими синдромами $S_i(x)$ i–ых кодовых слов циклического кода.

Таким образом, для формирования бесконечной серии конечных синдромов для алгебраического декодирования сверточных кодов достаточно выполнить условие $Z(x) = 0$.

Для выполнения сформулированного условия рассмотрим правило формирования многочлена $Z(x)$. Как показано выше, многочлен $Z(x)$ формируется суммированием многочленов, коэффициентами которых являются элементы кодовых i–ых слов, сдвинутых на $j \cdot K$ символов вправо. Предположим, что бесконечное кодовое слово нерекурсивного сверточного кода, алгебраически заданного через порождающий многочлен циклического кода (см. рис. 4.4.), состоит из бесконечной суммы кодовых слов циклического кода, умноженных на оператор задержки $x^{i \cdot N}$. Тогда многочлен $Z(x)$ будет равен сумме многочленов, коэффициентами которых являются элементы кодовых i–ых слов, сдвинутых на $j \cdot N$ символов вправо. Но по определению, вектор S^* – синдромная последовательность, соответствующая блоку из N кодовых символов бесконечного кодового слова. Практически это означает, что третье слагаемое в выражении (4.33) равно нулю, т.е. $Z(x) = 0$ соответственно. Подставив $Z(x) = 0$ в (4.34), получим

$$S^*(x) = S_i(x).$$

Последнее выражение позволяет формировать бесконечную серию конечных синдромов бесконечного кодового слова сверточного кода. Практически это обеспечивает реализацию разработанного выше алгебраического алгоритма декодирования сверточных кодов.

Таким образом, в результате проведенных исследований разработан способ формирования бесконечной серии конечных синдромов для алгебраического декодирования сверточных кодов.

Следует отметить, что при этом наблюдается некоторое ухудшение конструктивных свойств сверточного кода. Для реализации предлагаемого способа при формировании кодового слова алгебраического сверточного кода оператор задержки $x^{i \cdot K}$ следует заменить на $x^{i \cdot N}$. Практически это означает, что после

подачи на вход кодера K информационных символов необходимо подать дополнительно $(N – K)$ нулевых символов. В этом случае на приемной стороне выполнится условие $Z(x) = 0$ и, соответственно, равенство $S^*(x) = S_i(x)$. В терминах кодирования подача на вход кодера $(N – K)$ нулевых символов соответствует снижению скорости кодирования в $(N – (N – K))/N = K/N$ раз, т.е. снижение скорости пропорционально скорости циклического (N, K, D) кода.

Последнее обстоятельство несколько снижает помехоустойчивость алгебраических сверточных кодов (с алгебраическим способом декодирования). Однако при соответствующем выборе параметров циклического (N, K, D) кода это ухудшение можно минимизировать. Действительно, если при построении сверточного кода использовать циклические (N, K, D) коды с $R = K/N \to 1$, то для реализации алгебраического декодирования необходимо внести ничтожно малую долю нулевых символов и, таким образом, снижение помехоустойчивости будет минимальным.

Приведем *пример*. Зафиксируем конечное поле $GF(2^3)$ и порождающий многочлен (7, 3, 5) кода РС, например $g(x) = (x + \alpha^1)\cdot(x + \alpha^2)\cdot(x + \alpha^3)\cdot(x + \alpha^4)$, где α – примитивный элемент поля $GF(2^3)$. Тогда, по теоремам 1–3 (см. раздел 2), имеем алгебраически заданные сверточные коды с параметрами

1. $k^0 = 1; n^0 = 3; \nu = 4; k = 5; n = 15; R = 1 / 3; d_\infty \geq 5;$

2. $k^0 = 2; n^0 = 3; \nu = 8; k = 10; n = 15; R = 2 / 3; d_\infty \geq 5.$

В этом случае для реализации алгебраического алгоритма декодирования на передающей стороне после подачи на вход кодера трех информационных символов следует подать четыре нуля. В результате скорость сверточного кода будет равна, соответственно, $R = 1 / 7$ и $R = 2 / 7$, что, очевидно, очень сильно ухудшает параметры сверточного кода, что, соответственно, существенно снизит его помехоустойчивость.

Рассмотрим теперь (63, 55, 5) код БЧХ над $GF(2^3)$ с порождающим многочленом, например, вида

$$g(x)=(x+\alpha^1)\cdot(x+\alpha^2)\cdot(x+\alpha^3)\cdot(x+\alpha^4)\cdot(x+\alpha^8)\cdot(x+\alpha^{16})\cdot(x+\alpha^{24})\cdot(x+\alpha^{32}),$$

где α – примитивный элемент поля $GF((2^3)^2)$.

Тогда, по теоремам 1–3 (раздел 2), имеем алгебраически заданные сверточные коды с параметрами

1. $k^0 = 1; n^0 = 3; \nu = 8; k = 9; n = 27; R = 1/3; d_\infty \geq 5;$

2. $k^0 = 2; n^0 = 3; \nu = 8; k = 18; n = 27; R = 2/3; d_\infty \geq 5.$

Теперь для реализации алгебраического алгоритма на приемной стороне декодирования сверточного кода со сходными параметрами на передающей стороне после подачи на вход кодера 55 информационных символов следует подать 8 нулей. В результате скорость сверточного кода будет равна, соответственно, $R=55/189 \approx 1/3$ и $R = 110/189 \approx 2/3$, что, очевидно, практически оставляет без изменения параметры сверточного кода и, соответственно, не приводит к значительному ухудшению его помехоустойчивости.

Другой подход алгебраического декодирования сверточных кодов с сохранением высоких конструктивных показателей может состоять в использовании циклических кодов, заданных над большим полем. Приведем, например, следующее сравнение. Зафиксируем конечное поле $GF(2^6)$ и порождающий многочлен (63, 59, 5) кода РС, например $g(x) = (x+\alpha^1)\cdot(x+\alpha^2)\cdot(x+\alpha^3)\cdot(x+\alpha^4)$,

где α – примитивный элемент поля $GF(2^6)$.

Тогда, по теоремам 1–3 (см. раздел 2), имеем алгебраически заданные сверточные коды с параметрами

1. $k^0 = 1; n^0 = 6; \nu = 4; k = 5; n = 30; R = 1/6; d_\infty \geq 5;$

2. $k^0 = 2; n^0 = 6; \nu = 4; k = 10; n = 30; R = 1/3; d_\infty \geq 5;$

3. $k^0 = 3; n^0 = 6; \nu = 4; k = 15; n = 30; R = 1/2; d_\infty \geq 5;$

4. $k^0 = 4; n^0 = 6; \nu = 4; k = 20; n = 30; R = 2/3; d_\infty \geq 5;$

5. $k^0 = 5; n^0 = 6; \nu = 4; k = 25; n = 30; R = 5/6; d_\infty \geq 5.$

Второй и четвертый коды в приведенном списке по конструктивным параметрам очень близки к рассмотренным выше, однако имеют меньшую длину кодового ограничения. Для реализации алгебраического алгоритма их декодирования на передающей стороне после подачи на вход кодера 59 информационных символов следует подать дополнительно четыре нуля. В результате скорость сверточного кода будет равна, соответственно, $R = 59 / 189 \approx 1/3$ и $R = 118 / 189 \approx 2/3$, что, очевидно, лучший результат, по сравнению с использованием (63, 55, 5) кода БЧХ над $GF(2^3)$.

Вопросы практической реализации предложенного алгебраического метода декодирования применительно к конкретным практическим приложениям подробно исследуются в следующем разделе.

Кроме того, как показано ниже, применение предлагаемого подхода в комбинации с известными методами позволяет существенно повысить эффективность декодирования сверточных кодов.

4.4. Разработка комбинированного метода декодирования алгебраических сверточных кодов

Рассмотрим последовательное декодирование (например, алгоритмом Фано) сверточного кода, алгебраически заданного через циклический (N, K, D) код. Предположим, что для устранения эффекта переполнения буфера в алгоритме Фано используется искусственное внесение в передаваемые данные набора из $\nu = k^0 r$ нулей (см. подраздел 4.1). Предположим также, что эта процедура выполняется с периодичностью $M \cdot K$ информационных символов, где M – произвольное целое, отличное от нуля.

Как показано выше, бесконечное кодовое слово сверточного кода распадается на бесконечную сумму кодовых слов циклического кода. Структуру кодового слова с учетом периодичного внесения ν нулей представим на рис. 4.8.

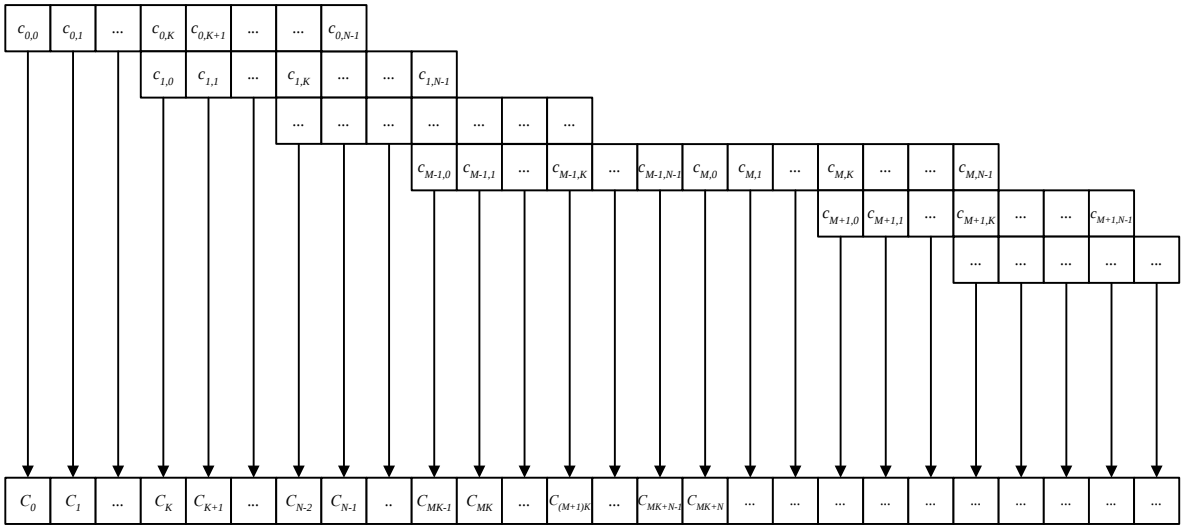


Рис. 4.8. Структура кодового слова алгебраического сверточного кода с периодично вносимыми *v* нулями (период – *M·K* информационных символов)

Как видно из рис. 4.8, каждые *(M–1)K + N* кодовых символов представляют собой сумму соответствующих *M·K* кодовых слов циклического *(N, K, D)* кода, т.е.

$$\|C\|_{MK+N} = \sum_{j=0}^{M-1} \|c\|_j \cdot \|0,I\|_{N,j\cdot K+N}, \qquad (4.35)$$

где $\|c\|_j$ – *j*–ое кодовое слово циклического кода,

$$\|c\|_j = (c_{j,0}, c_{j,1}, ..., c_{j,N-1});$$

$\|0,I\|_{N,i\cdot K+N}$ – единичная матрица с добавленными слева $i\cdot K$ нулевыми столбцами.

Обозначим, как и ранее, через X_l – l – ый корень порождающего многочлена *P(x)*, причем $X_l = \alpha^{Jl}$ для некоторого J_l, $X_l \in GF(q^m)$.

Если $X_0, X_1, ..., X_{r-1}$ – все корни многочлена *P(x)*, т.е. $P(x) = (x + X_0)\cdot(x + X_1) \cdot ... \cdot (x + X_{r-1})$, то справедливо равенство

$$c(X_i) = c_{j,0} + c_{j,1} X_i + c_{j,2} X_i^{\,2} + ... + c_{j,N-1} X_i^{\,N-1} = 0,$$

где $c(x)$ - кодовый многочлен циклического (N, K, D) кода.

Обобщив полученное выражение до суммы (4.35), получим

$$C_0 + C_1 X_i + C_2 X_i^2 + \ldots + C_{MK+N-1} X_i^{MK+N-1} =$$
$$= (c_{0,0} + c_{0,1} X_i + c_{0,2} X_i^2 + \ldots + c_{0,N-1} X_i^{N-1}) +$$
$$+ X_i^K (c_{1,0} + c_{1,1} X_i + c_{1,2} X_i^2 + \ldots + c_{1,N-1} X_i^{N-1}) + \ldots +$$
$$+ X_i^{MK} (c_{M-1,0} + c_{M-1,1} X_i + c_{M-1,2} X_i^2 + \ldots + c_{M-1,N-1} X_i^{N-1}) = 0.$$

После раскрытия скобок получим

$$C_0 + C_1 X_i + C_2 X_i^2 + \ldots + C_{MK+N-1} X_i^{MK+N-1} =$$
$$= (c_{0,0} + c_{0,1} X_i + c_{0,2} X_i^2 + \ldots + c_{0,N-1} X_i^{N-1}) +$$
$$+ (c_{1,0} X_i^K + c_{1,1} X_i^{K+1} + c_{1,2} X_i^{K+2} + \ldots + c_{1,N-1} X_i^{K+N-1}) + \ldots +$$
$$+ (c_{M-1,0} X_i^{MK} + c_{M-1,1} X_i^{MK+1} + c_{M-1,2} X_i^{MK+2} + \ldots +$$
$$+ c_{M-1,N-1} X_i^{MK+N-1}) = 0.$$

Если $X_i \in GF(Q)$, $Q = q^m$, $X_i \neq 0$, то справедливо равенство

$$X_i^{Q+a} = X_i^{a+1}.$$

Если при этом выполняется равенство $N = q^m - 1$, то имеем

$$C_0 + C_1 X_i + C_2 X_i^2 + \ldots + C_{MK+N-1} X_i^{MK+N-1} =$$
$$= C_0 + C_1 X_i + C_2 X_i^2 + \ldots + C_{N-1} X_i^{N-1} + \ldots +$$
$$+ C_{MK+N-1} X_i^{MK-1} = (c_{0,0} + c_{0,1} X_i + c_{0,2} X_i^2 + \ldots + c_{0,N-1} X_i^{N-1}) +$$
$$+ (c_{1,0} X_i^K + c_{1,1} X_i^{K+1} + c_{1,2} X_i^{K+2} + \ldots + c_{1,N-K-1} X_i^0 + \ldots +$$
$$+ c_{1,N-1} X_i^{K-1}) + \ldots + (c_{M-1,0} X_i^{MK^*} + c_{M-1,1} X_i^{(MK+1)^*} +$$
$$+ c_{M-1,2} X_i^{(MK+2)^*} + \ldots + c_{M-1,N-1} X_i^{(MK-1)^*}) = 0,$$

где $(\xi)^* = (\xi) mod(q^m - 1)$.

Перепишем последнее выражение в виде матричного произведения:

$$C(X_i) = (C_0, C_1, C_2, \ldots, C_{MK+N-1}) \cdot (1, X_i, X_i^2, \ldots, X_i^{MK+N-1})^T =$$
$$= (C_0, C_1, C_2, \ldots, C_{N-1}, \ldots, C_{MK+N-1}) \cdot (1, X_i, X_i^2, \ldots, X_i^{N-1}, \ldots, X_i^{MK-1})^T =$$
$$= (c_{0,0}, c_{0,1}, c_{0,2}, \ldots, c_{0,N-1}) \cdot (1, X_i, X_i^2, \ldots, X_i^{N-1})^T +$$
$$+ (c_{1,0}, c_{1,1}, c_{1,2}, \ldots, c_{1,N-K-1}, \ldots, c_{1,N-1}) \cdot$$
$$\cdot (X_i^K, X_i^{K+1}, X_i^{K+2}, \ldots, X_i^0, \ldots, X_i^{K-1})^T + \ldots +$$
$$+ (c_{M-1,0}, c_{M-1,1}, c_{M-1,2}, \ldots, c_{M-1,N-1}) \cdot$$
$$\cdot (X_i^{MK*}, X_i^{(MK+1)*}, X_i^{(MK+2)*}, \ldots, X_i^{(MK-1)*})^T = 0.$$

Следует отметить, что в каждом произведении последнего выражения есть сомножитель $(1, X_i, X_i^2, \ldots, X_i^{N-1})^T$, умноженный последовательно на $X_i^{\ j}$, или, что эквивалентно, циклически сдвинутый на K символов вправо. Следовательно, справедливо равенство

$$C(X_i) = (c_{0,0}, c_{0,1}, c_{0,2}, \ldots, c_{0,N-1}) \cdot (1, X_i, X_i^2, \ldots, X_i^{N-1})^T +$$
$$+ (c_{1,N-K-1}, c_{1,N-K}, \ldots, c_{1,N-1}, c_{1,0}, c_{1,1}, c_{1,2}, \ldots, c_{1,N-K-2}) \cdot$$
$$\cdot (1, X_i, X_i^2, \ldots, X_i^{N-1})^T + \ldots +$$
$$+ (c_{M-1,-MK*}, c_{M-1,(-MK-1)*}, c_{M-1,(-MK-2)*}, \ldots, c_{M-1,(-MK+1)*}) \cdot$$
$$\cdot (1, X_i, X_i^2, \ldots, X_i^{N-1})^T = 0.$$

Обобщим последнее равенство для всех корней $P(x)$, получим

$$(c_{0,0}, c_{0,1}, c_{0,2}, \ldots, c_{0,N-1}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \ldots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \ldots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \ldots & X_2^{N-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & X_{r-1} & X_{r-1}^2 & \ldots & X_{r-1}^{N-1} \end{vmatrix}^T +$$

$$+ (c_{1,N-K-1}, c_{1,N-K}, \ldots, c_{1,N-1}, c_{1,0}, \ldots, c_{1,N-K-2}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \ldots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \ldots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \ldots & X_2^{N-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & X_{r-1} & X_{r-1}^2 & \ldots & X_{r-1}^{N-1} \end{vmatrix}^T + \ldots$$

$$+ (c_{M-1,MK*}, c_{M-1,(-MK-1)*}, \ldots, c_{M-1,(-MK+1)*}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \ldots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \ldots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \ldots & X_2^{N-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & X_{r-1} & X_{r-1}^2 & \ldots & X_{r-1}^{N-1} \end{vmatrix}^T = 0.$$

Приведем подобные, получим

$$(c_{0,0} + c_{1,N-K-1} + \ldots + c_{M-1,MK^*},$$
$$c_{0,1} + c_{1,N-K} + \ldots + c_{M-1,(-MK-1)^*}, \ldots, \tag{4.36}$$
$$c_{0,N-1} + c_{M-1,(-MK-1)^*} + \ldots + c_{M-1,(-MK+1)^*}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \ldots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \ldots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \ldots & X_2^{N-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & X_{r-1} & X_{r-1}^2 & \ldots & X_{r-1}^{N-1} \end{vmatrix}^T = 0.$$

Последнее выражение соответствует условию взаимной ортогональности вектора $(c_{0,0} + c_{1,N-K-1} + \ldots + c_{M-1,-MK^*}, c_{0,1} + c_{1,N-K} + \ldots + c_{M-1,(-MK-1)^*}, \ldots, c_{0,N-1} + c_{1,N-K-2} + \ldots + c_{M-1,(-MK+1)^*})$ и матрицы в правой части произведения.

Как показано выше, матрица в правой части произведения – проверочная матрица (N, K, D) циклического кода.

Для рассмотрения влияния ошибок предположим, что кодовое слово исказилось при передаче. Пусть число ошибок на блоке из $(M-1)K + N$ кодовых символов не превышает исправляющей способности $t = (D-1)/2$ циклического (N, K, D) кода.

Тогда кодовое слово с ошибками C^* на длине $(M-1)K + N$ кодовых символов запишется в виде

$$\|C\|^*_{MK+N} = \|C\|_{MK+N} + \|e\|_{MK+N}.$$

Подставив последнее выражение в (4.35), получим

$$\|C\|_{MK+N} = \left(\sum_{j=0}^{M-1} \|c\|_j \cdot \|0, I\|_{N, j \cdot K+N} \right) + \|e\|_{MK+N} = \sum_{j=0}^{M-1} \left(\|c\|_j \cdot \|0, I\|_{N, j \cdot K+N} + \|e_i, 0\|_N^j \right),$$

где $\|e_i, 0\|_N^j$ – вектор ошибок E_i длины K символов с добавленными справа $(N-K)$ нулями, он соответствует j–му кодовому слову $\|c\|_j$ и согласно (4.18) равен

$$E_i(x) = e_{i \cdot K} + e_{i \cdot K+1} x + e_{i \cdot K+2} x^2 + \ldots e_{(i+1) \cdot K-1} x^{K-1}.$$

Вычислим значения кодового многочлена с ошибками во всех корнях порождающего многочлена *P(x)*.

После соответствующей подстановки получим

$$\sum_{j=0}^{M-1}\left(\left\|c\right\|_j \cdot \left\|0,I\right\|_{N,j\cdot K+N} + \left\|e_i,0\right\|_N^j\right)\cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix} =$$

$$= \sum_{j=0}^{M-1}\left(\left\|c\right\|_j \cdot \left\|0,I\right\|_{N,j\cdot K+N}\right)\cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix} +$$

$$+ \sum_{j=0}^{M-1}\left(\left\|e_i,0\right\|_N^j\right)\cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix} =$$

$$= (c_{0,0} + c_{1,N-K-1} + \dots + c_{M-1,MK^*},$$

$$c_{0,1} + c_{1,N-K} + \dots + c_{M-1,(-MK-1)^*}, \dots,$$

$$c_{0,N-1} + c_{M-1,(-MK-1)^*} + \dots + c_{M-1,(-MK+1)^*})\cdot$$

$$\cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix}^T +$$

$$+ \sum_{j=0}^{M-1}\left(\left\|e_i,0\right\|_N^j\right)\cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix}.$$

Как следует из (4.36), первое слагаемое в правой части последнего выражения равно нулю. Следовательно

$$\|S\|_{(N-K)} = \|C\|^{*}_{MK+N} \cdot \|H\|_{N-K,N} = \sum_{j=0}^{M-1} \left(\|e_i, 0\|_N^j \right) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix},$$

т.е. синдром $\|S\|_{(N-K)}$ зависит только от вектора ошибок $\|e\|_{MK+N}$ и не зависит от кодового слова $\|C\|_{MK+N}$. Перепишем полученное выражение в виде

$$\begin{aligned} \|S\|_{(N-K)} = (& e_0 + e_K + e_{2K} + \dots + e_{(M-1)K}, \\ & e_1 + e_{K+1} + e_{2K+1} + \dots + e_{(M-1)K+1}, \dots, \\ & e_{K-1} + e_{2K-1} + e_{3K-1} + \dots + e_{MK}, e_{MK+1}, \dots, e_{MK+N}) \cdot \\ & \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \dots & X_0^{N-1} \\ 1 & X_1 & X_1^2 & \dots & X_1^{N-1} \\ 1 & X_2 & X_2^2 & \dots & X_2^{N-1} \\ \dots & \dots & \dots & \dots & \dots \\ 1 & X_{r-1} & X_{r-1}^2 & \dots & X_{r-1}^{N-1} \end{vmatrix}. \end{aligned} \qquad (4.37)$$

Выражение (4.37) по сути, соответствует ограничению синдромного вектора (4.24) на длину кодового слова *(M–1)K + N* кодовых символов и, очевидно, может быть получено другим способом (другой последовательностью соответствующих преобразований). Таким образом, в результате проведенных исследований получено обобщенное представление бесконечного кодового слова через бесконечную сумму последовательных наборов из *M* кодовых слов *(N, K, D)* циклического кода. Это дает мощный механизм комбинированного декодирования алгебраически заданных сверточных кодов.

Действительно, синдромная последовательность в выражении (4.37) зависит от вектора ошибок, компоненты которого сгруппированы в периодическую структуру. Практически это означает, что соответствующий синдром задает рекуррентное правило формирования систем линейных уравнений (4.30) – (4.31), решения которых локализуют ошибку с точностью до периодичной структуры (коэффициенты вектора ошибок сгруппированы с периодом *K*). При соответствующем выборе

параметров циклического *(N, K, D)* кода можно задать период группирования ошибок, превосходящий средний размер пакета ошибок. Это позволит с помощью предложенных алгебраических процедур свести задачу декодирования коррелированных ошибок к декодированию ошибок, близких к независимому распределению. Декодирование независимых ошибок (нахождение значения ошибок, локализованых с точностью до периодичной структуры) может быть возложено на другой метод декодирования сверточных кодов, например на последовательный алгоритм. Действительно, если для устранения эффекта переполнения буфера в алгоритме последовательного декодирования (например, алгоритма Фано) используется искусственное внесение в передаваемые данные набора из *v* нулей с периодичностью *M·K* информационных символов, то работа этого алгоритма внутри блока из *M·K* символов может быть существенно упрощена. Поскольку алгебраический алгоритм локализовал (с точностью до периода) ошибку, то последовательный поиск по всей кодовой решетке можно заменить на поиск только в тех узлах, которые входят в соответствующую локализацию. Алгоритм такого комбинированного декодирования алгебраически заданного сверточного кода представим в виде последовательности следующих шагов:

Шаг 1. Прием $(M-1)K + N$ кодовых символов из $GF(q^m)$ (или, что эквивалентно, $(M \cdot K + N)m$ кодовых символов из $GF(q)$).

Шаг 2. Вычисление синдрома по выражению (4.37).

Шаг 3. Решение систем линейных уравнений (4.30) – (4.31).

Шаг 4. Локализация ошибок с точностью до периода *K* символов.

Шаг 5. Последовательный поиск по кодовой решетке в узлах, соответствующих компонентам сгруппированной ошибки.

Шаг 6. Исправление сгруппированной ошибки.

Шаг 7. Прием следующих $(M-1)K + N$ кодовых символов из $GF(q^m)$ (или, что эквивалентно, $(M \cdot K + N)m$ кодовых символов из $GF(q)$). Переход к шагу 2 и т.д.

Поиск по кодовой решетке (шаг 5) может выполняться и сразу, после приема первого кодового символа (как это делается при последовательном декодировании). Это может быть оправдано при малом числе ошибок. Если же число ошибок велико, то сложность последовательного декодирования быстро возрастает и, очевидно, следует ожидать предварительной локализации ошибок (шаг 4).

На рис. 4.9 приведена в общем виде схема алгоритма комбинированного декодирования алгебраически заданного сверточного кода.

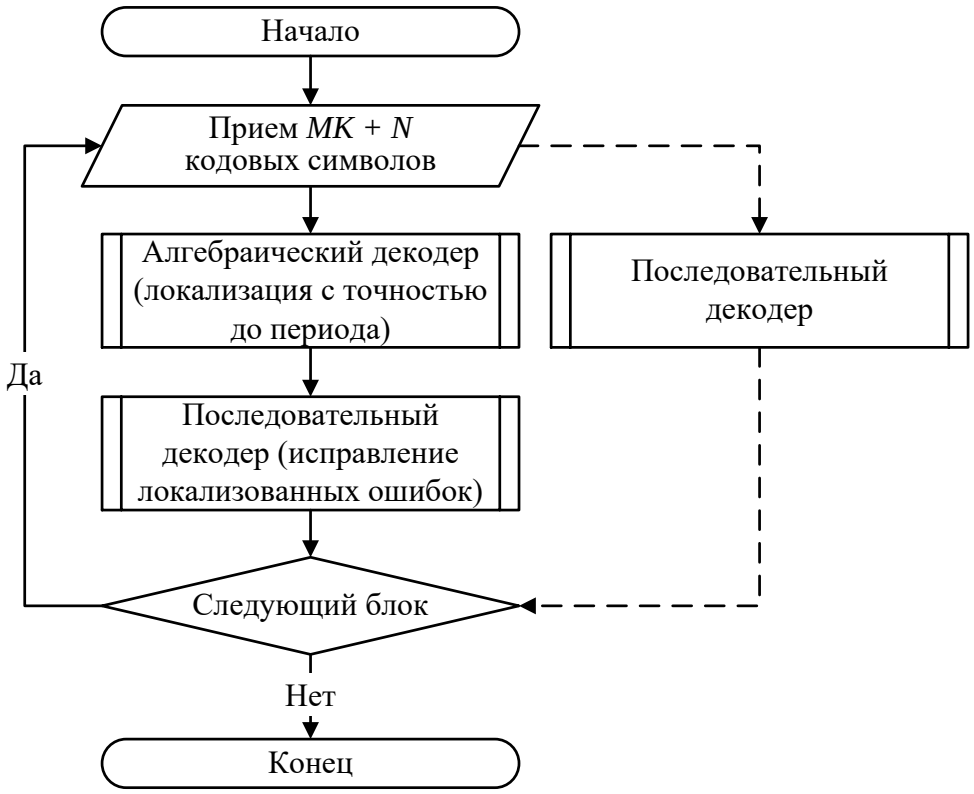


Рис. 4.9. Схема алгоритма комбинированного декодирования алгебраически заданного сверточного кода

Рассмотрим *пример* построения алгоритма комбинированного декодирования алгебраически заданного сверточного кода. Зафиксируем *(7, 3, 5)* код РС над $GF(2^3)$ с порождающим многочленом:

$$P(x) = (x + \alpha^1)\cdot(x + \alpha^2)\cdot(x + \alpha^3)\cdot(x + \alpha^4) = x^4 + x^3 + \alpha^1 x^2 + \alpha^6 x + \alpha^5,$$

где α – примитивный элемент поля $GF(2^3)$.

Пусть задан алгебраический сверточный код с параметрами $k^0 = 1$, $n^0 = 3$, $\nu = 4$, $k = 5$, $n = 15$, $R = 1/3$, $d_\infty \geq 5$. Соответствующие порождающие многочлены сверточного кода равны

$$P_1(x) = x^4 + x^3 + x + 1;$$

$$P_2(x) = x^2 + 1;$$

$$P_1(x) = x + 1.$$

Схема кодера приведена на рис. 4.10. Соответствующая кодовая решетка приведена на рис. 4.12.

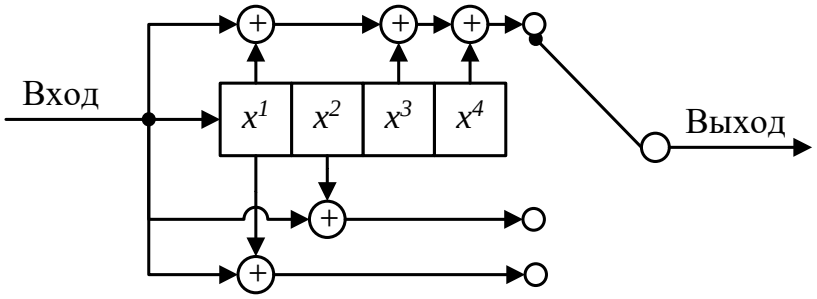


Рис. 4.10. Схема кодера алгебраического сверточного (15, 5) кода

Предположим, что используется комбинированный метод декодирования с $M = 2$. Тогда структуру кодового слова представим в виде рис.4.11.

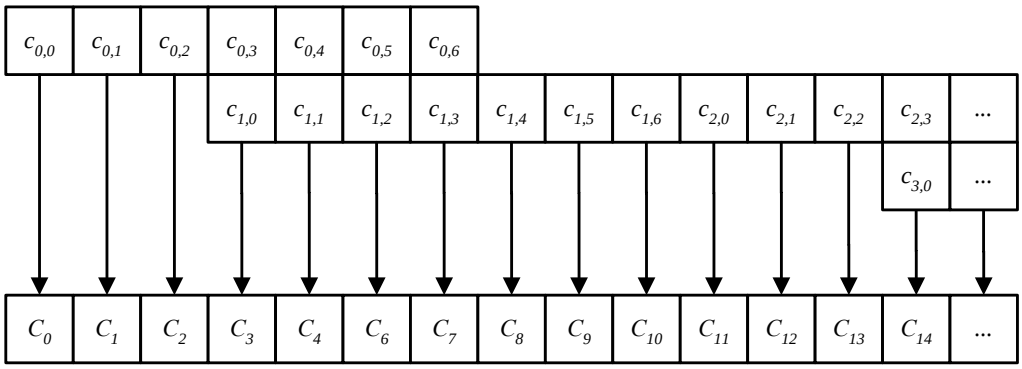


Рис.4.11. Структура кодового слова сверточного (15, 5) кода при комбинированном алгоритме декодирования *(M = 2)*

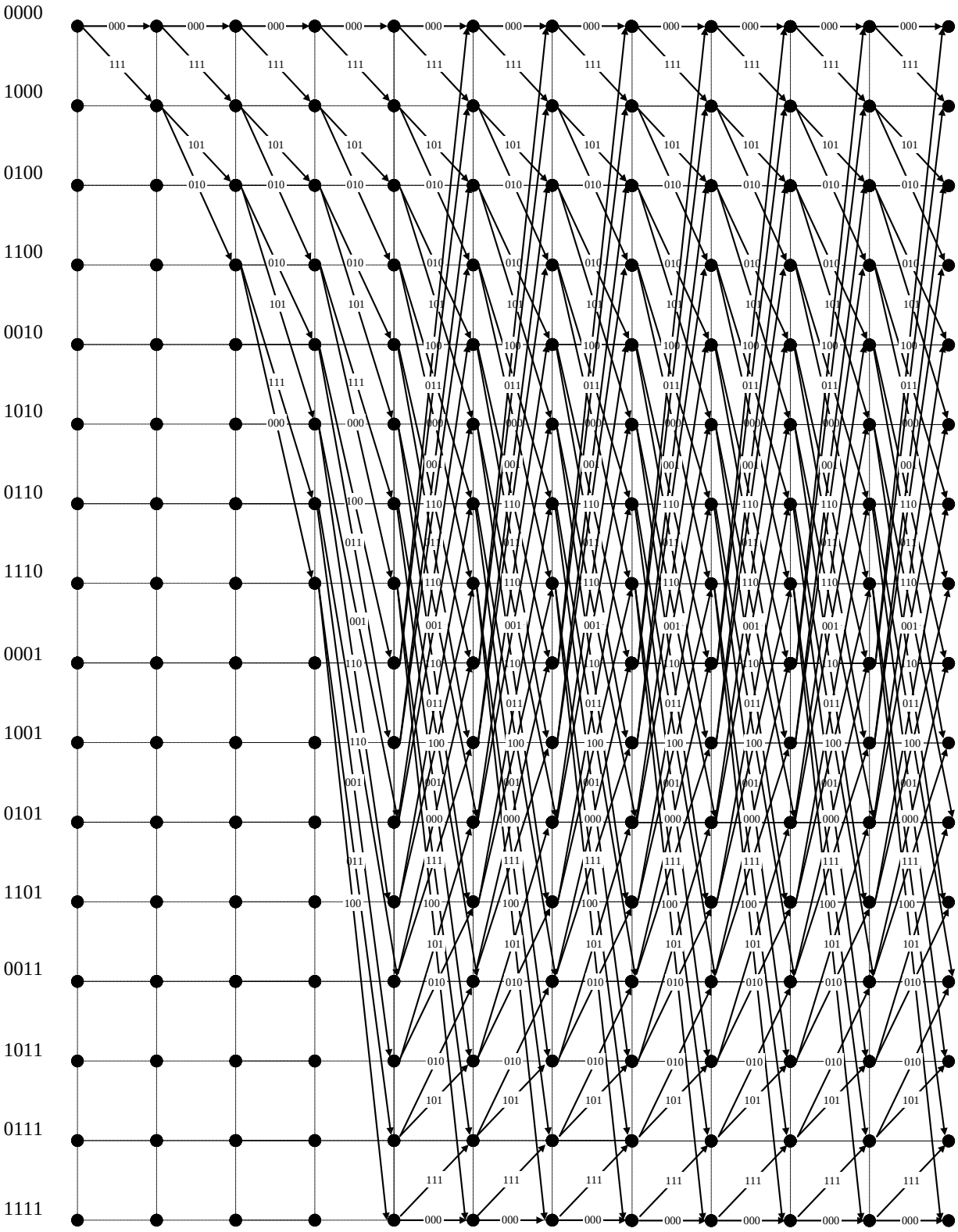


Рис. 4.12. Кодовая решетка алгебраического сверточного (15, 5) кода

Воспользуемся алгоритмом комбинированного декодирования. Рассмотрим блок данных из $(M–1)K + N = 11$ кодовых символов из $GF(q^m)$, т.е. фрагмент бесконечного кодового слова из 11 символов: $//C//_{10} = (C_0, C_1, \ldots, C_{10})$.

Пусть на вход кодера подается информационная последовательность $I = (1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, \ldots)$. Тогда с выхода кодера в канал поступает кодовое слово $C = (111, 101, 101, 001, 110, 011, 001, 101, 001, 110, 011, \ldots)$.

Предположим также, что при передаче по каналу связи на $//C//_{10}$ произошло 2 ошибки, например, $//e//_{10} = (0, 1, \ldots, 0, 1, 0)$, т.е. ошибки произошли в C_1 и C_9 (см. табл. 4.1).

Таблица 4.1

Кодовое слово, вектор ошибки, искаженное кодовое слово сверточного (15, 5) кода и фрагменты последовательного декодирования

	0	1	2	3	4	5	6	7	8	9	10	…
C	*111*	*101*	*101*	*001*	*110*	*011*	*001*	*101*	*001*	*110*	*011*	*…*
e		*100*								*100*		
*C**	*111*	*001*	*101*	*001*	*110*	*011*	*001*	*101*	*001*	*010*	*011*	*…*
t(l)	*1*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*8*	*9*	
t(l)–		*0*								*7*		

Вычислим синдром:

$$\|S\|_{(N-K)} = (e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}) \cdot \begin{vmatrix} 1 & X_0 & X_0^2 & \ldots & X_0^6 & .. & X_0^{10} \\ 1 & X_1 & X_1^2 & \ldots & X_1^6 & .. & X_1^{10} \\ 1 & X_2 & X_2^2 & \ldots & X_2^6 & .. & X_2^{10} \\ \ldots & \ldots & \ldots & \ldots & \ldots & .. & \ldots \\ 1 & X_{r-1} & X_{r-1}^2 & \ldots & X_{r-1}^6 & .. & X_{\kappa-1}^{10} \end{vmatrix}.$$

Подставим вместо X_i корни порождающего многочлена $P(x) = (x + \alpha^1)\cdot(x + \alpha^2)\cdot(x + \alpha^3)\cdot(x + \alpha^4)$.

Для всех $\alpha^I \neq 0$ выполняется условие $(\alpha^j)^{Q+a} = (\alpha^j)^{a+1}$,

следовательно, запишем

$$\|S\|_{(N-K)} = (e_0 + e_7, e_1 + e_8, e_2 + e_9, e_3 + e_{10}, e_4, e_5, e_6) \cdot$$

$$\cdot \begin{vmatrix} 1 & \alpha^1 & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \alpha^1 & \alpha^3 & \alpha^5 \\ 1 & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^5 & \alpha^1 & \alpha^4 \\ 1 & \alpha^4 & \alpha^1 & \alpha^5 & \alpha^2 & \alpha^6 & \alpha^3 \end{vmatrix} = \begin{pmatrix} \alpha^1 & + & \alpha^2 \\ \alpha^2 & + & \alpha^4 \\ \alpha^3 & + & \alpha^6 \\ \alpha^4 & + & \alpha^1 \end{pmatrix} = \begin{pmatrix} \alpha^4 \\ \alpha^1 \\ \alpha^4 \\ \alpha^2 \end{pmatrix}.$$

Найденный синдромный вектор задает систему линейных уравнений

$$\alpha^4 + \lambda_1 \cdot \alpha^1 + \lambda_0 \cdot \alpha^4 = 0;$$
$$\alpha^2 + \lambda_1 \cdot \alpha^4 + \lambda_0 \cdot \alpha^1 = 0.$$

Решив систему уравнений, получим $\lambda_0 = \alpha^3$, $\lambda_1 = \alpha^4$, что дает возможность сформировать многочлен локаторов ошибок:

$$\Lambda(x) = x^2 + \alpha^4 x + \alpha^3.$$

Воспользуемся процедурой Ченя, найдем корни $\Lambda(x)$:

$$\Lambda(x) = (x + \alpha^1) \cdot (x + \alpha^2),$$

откуда следует, что ошибки локализуются следующим образом: $(e_1 + e_8)$ и $(e_2 + e_9)$. Найдем значение ошибок. С учетом локализации сформируем систему линейных уравнений

$$(e_1 + e_8) \cdot \alpha^1 + (e_2 + e_9) \cdot \alpha^3 = \alpha^4;$$
$$(e_1 + e_8) \cdot \alpha^2 + (e_2 + e_9) \cdot \alpha^4 = \alpha^1.$$

Решение системы уравнений дает следующее:

$$(e_1 + e_8) = \alpha^0 = (100);$$
$$(e_2 + e_9) = \alpha^0 = (100).$$

Полученное решение означает следующее. Произошло две ошибки:

- одна ошибка (100) в символе C_1 или C_8,;
- другая ошибка (100) произошла в символе C_2 или C_9.

Таким образом, разработанная алгебраическая процедура локализации ошибок позволила установить размещение ошибок с точностью до периода в семь символов. Практически это означает, что при выполнении последовательного поиска (шаг 5 в алгоритме комбинированного декодировании) необходимо осуществить поиск по решетке только в символах *(C_1* или *C_8)* и *(C_2* или *C_9)*. Строго говоря, на этом этапе может использоваться любой алгоритм декодирования сверточных кодов, который позволяет принять решение о текущем символе. Решение принимается с учетом того, что все символы, за исключением *(C_1* или *C_8)* и *(C_2* или *C_9)*, приняты без ошибок.

Рассмотрим теперь работу комбинированного алгоритма декодирования на шаге 5 – последовательный поиск по кодовой решетке в узлах, соответствующих компонентам сгруппированной ошибки. Поскольку известно, что ошибок в символах C_0, $C_3 – C_7$, C_{10} не произошло, поиск по решетке в этих местах проводить не нужно, что существенно ускоряет работу последовательного декодера. Перейдем сразу к символу C_1.

На рис. 4.13 приведена иллюстрация работы алгоритма Фано при комбинированном декодировании. Предположим, что исходными данными алгоритма Фано являются следующие: $P_0 = 0{,}3$, $P^* = 1/3$.

Тогда правило принятия решения запишем в виде

$$t(l) = P^* n^0 l - d(l) = l - d(l),$$

т.е. движение по решетке считается правильным, если величина *t(l)* не уменьшается. В противном случае результат поиска считается неверным и нужно изменить путь.

В табл. 4.1 в строке «*t(l)*» отмечаются значения *t(l)* для истинного пути (см. рис. 4.13). В строке «*t(l)*–» отмечены значения *t(l)* для ошибочно выбранного пути (тонкие стрелки на рис. 4.13).

Действительно, при работе алгоритма Фано вычисленное в символе C_1 значение *t(l)* для истинного пути равно следующему:

$$t(l) = 2 - d(101, 001) = 2 - 1 = 1,$$

т.е. *t(l)* не уменьшается.

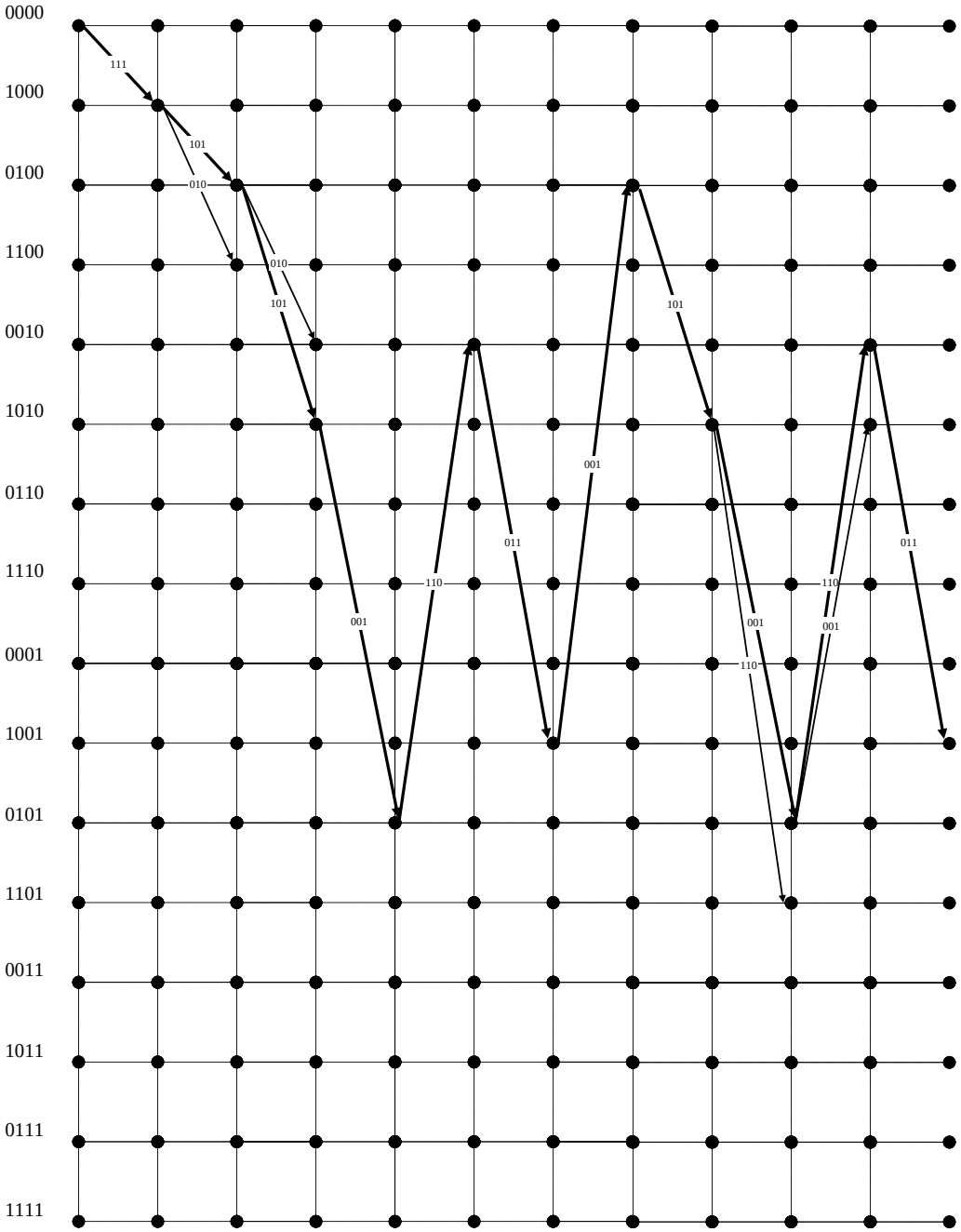


Рис. 4.13. Иллюстрация работы алгоритма Фано при комбинированном декодировании (шаг 5)

Для другого (ошибочного) направления движения это значение равно

$$t(l) = 2 - d(010, 001) = 2 - 2 = 0$$

и *t(l)* уменьшается, что свидетельствует об ошибочности этого

пути.

Далее последовательный поиск по кодовой решетке осуществляется в символе C_2, что дает аналогичные результаты. В символах C_3–C_7 поиск производить не требуется, поскольку заранее известно, что там ошибок нет. Результаты поиска по кодовой решетке в символах C_8 и C_9 с помощью алгоритма Фано представлены в табл. 4.1 и отражены на рис. 4.13.

Следует отметить, что исправление ошибок в символах C_2 и C_9 можно осуществить несколько быстрее, а весь процесс декодирования еще больше упростить. Действительно, как только величина $d(l)$ становится отличной от нуля, очевидно, что этот кадр с ошибкой. Но значения ошибки уже вычислены алгебраической процедурой: $(e_1 + e_8) = \alpha^0 = (100)$, $(e_2 + e_9) = \alpha^0 = (100)$. Если последовательный поиск привел к $d(l)>0$ в символе C_1, значит $e_1 = \alpha^0 = (100)$, а $e_8 = 0 = (000)$ и можно сразу исправить ошибку в символе C_1 и отказаться от ее поиска в символе C_8. Подобную процедуру можно выполнить и в символах C_2 и C_9. Таким образом, последовательный поиск необходим только для уточнения расположения ошибок среди символов, сгруппированных алгебраической процедурой.

Применение разработанного комбинированного метода позволяет существенно ускорить процедуру декодирования алгебраического сверточного кода, что позволяет утверждать целесообразность его практического применения.

Выводы

1. Дальнейшее развитие методов декодирования алгебраических сверточных кодов позволяет обобщить задачу декодирования линейного блокового кода на случай бесконечной длины кодового слова непрерывных кодов в отличие от известных применением алгебраических процедур локализации и исправления ошибок циклических кодов.

2. В результате проведенных исследований установлено, что:

– бесконечное кодовое слово нерекурсивного сверточного кода, алгебраически заданного через порождающий многочлен циклического кода, состоит из бесконечной суммы кодовых слов

циклического кода;

– бесконечное кодовое слово алгебраического нерекурсивного сверточного кода, искаженное бесконечным вектором ошибок, состоит из бесконечной суммы кодовых слов циклического кода, искаженных вектором ошибок конечной размерности;

– бесконечный синдром принятого с ошибками кодового слова алгебраического нерекурсивного сверточного кода состоит из бесконечной суммы синдромов принятых кодовых слов циклического кода.

Таким образом, проведенные исследования позволяют свести декодирование бесконечного кодового слова непрерывного кода к бесконечной серии декодирований циклического блокового кода. Для реализации предложенного подхода необходимо и достаточно вычислить бесконечную сумму синдромов соответствующих кодовых слов циклического кода.

3. В результате проведенных исследований разработан способ формирования бесконечной серии конечных синдромов для алгебраического декодирования сверточных кодов. Реализация предлагаемого способа при формировании кодового слова алгебраического сверточного кода требует после подачи на вход кодера K информационных символов подать дополнительно $(N – K)$ нулевых символов, что соответствует снижению скорости кодирования в $(N – (N – K))/N = K/N$ раз, т.е. снижение скорости пропорционально скорости циклического (N, K, D) кода.

4. Полученое обобщенное представление бесконечного кодового слова через бесконечную сумму последовательных наборов из M кодовых слов (N, K, D) циклического кода дает мощный механизм комбинированного декодирования алгебраически заданных сверточных кодов (с использованием предложенных алгебраических процедур и известных алгоритмов, например, последовательного алгоритма Фано). Применение предложенных алгебраических процедур декодирования позволяет локализовать ошибки в кодовом слове сверточного кода с точностью до некоторого периода, выбираемого заранее. Это значительно упрощает работу второго алгоритма при ускорении последовательного поиска по кодовой

решетке.

РАЗДЕЛ 5
РАЗРАБОТКА И ИССЛЕДОВАНИЕ МЕТОДОВ ТУРБОКОДИРОВАНИЯ НА ОСНОВЕ АЛГЕБРАИЧЕСКИХ РЕКУРСИВНЫХ СВЕРТОЧНЫХ КОДОВ

Исследуются существующие методы построения турбокодов и процедуры их декодирования. Предлагается метод построения турбокодов с использованием алгебраических рекурсивных сверточных кодов, заданных через порождающий и/или проверочный многочлены недвоичного циклического кода. Разрабатываются практические алгоритмы построения турбокодов с требуемыми параметрами.

5.1. Исследование методов построения турбокодов и процедур их декодирования

Энергетическая эффективность телекоммуникационных систем зависит, в первую очередь, от энергетической эффективности применяемых методов помехоустойчивого кодирования, формирования и обработки сигнально–кодовых конструкций. Под энергетической эффективностью здесь и далее будем понимать минимально допустимое значение отношения энергии сигнала к спектральной плотности мощности шума E/N_o, требуемое для обеспечения заданной достоверности приема сообщения.

Проведенный в анализ показал, что применение турбокодов позволяет получить высокую энергетическую эффективность помехоустойчивого кодирования.

Использование турбокодов обеспечивает передачу сообщений с вероятностью ошибки $P_{ош} = 10^{-5}$ при величине E/N_o, превышающей лишь на 0,5 дБ минимально необходимую (граничную) величину для заданной скорости передачи информации, что является лучшим на сегодняшний день показателем.

Основным принципом построения турбокодов является параллельное соединение двух рекурсивных систематических сверточных кодеров (Recursive Systematic Convolutional Codes — RSC). Структурная схема турбокодера, в общем виде, представлена на рис. 5.1.

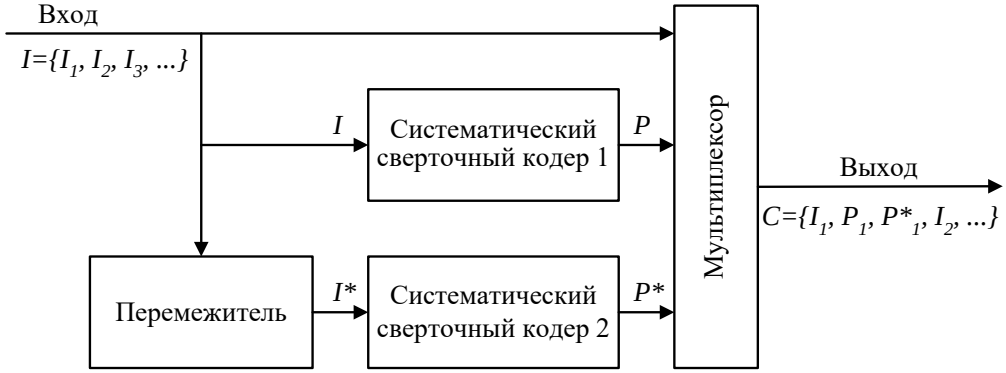


Рис. 5.1. Структурная схема турбокодера

Работа кодера осуществляется следующим образом. На вход подается информационная последовательность $I = \{I_1, I_2, I_3, ...\}$, которая одновременно поступает непосредственно на вход мультиплексора, на вход первого систематического сверточного кодера и на вход перемежителя. В перемежителе информационная последовательность перемешивается и подается на вход второго систематического сверточного кодера. Оба сверточных кодера формируют по введенной информационной последовательности проверочные части: P и P^*, соответственно. Сформированные проверочные данные поступают на вход мультиплексора. В мультиплексоре формируется кодовое слово C путем поочередного считывания поступивших на его вход информационных и проверочных символов: $C = \{I_1, P_1, P^*_1, I_2, P_2, P^*_2, ...\}$. Если скорость используемых сверточных кодов $R_{CK} = 1/2$, то скорость турбокода составляет $R_{TK} = 1/3$ (см. рис. 5.1). В общем случае, если $R_{CK} = 1/m$, то скорость турбокода определяется следующей леммой.

Лемма 5.1. Скорость турбокода, построенного на рекурсивных сверточных кодах с $R_{CK} = 1/m$ в систематическом виде, задается

выражением

$$R_{TK} = \frac{1}{2 \cdot m - 1}. \qquad (5.1)$$

Доказательство. Турбокодер, по определению, это параллельное каскадное соединение двух рекурсивных кодеров в систематическом виде. Если скорость сверточных кодов $R_{CK} = 1/m$, то на каждый информационный символ оба кодера формируют m кодовых символов. Если сверточный код задан в систематическом виде, то m кодовых символов содержат один информационный и $(m - 1)$ проверочных символов. В мультиплексоре поочередно считываются один информационный символ и с выхода каждого сверточного кодера по $(m - 1)$ проверочных символов. Всего для каждого информационного символа, поданного на вход турбокодера, на выходе мультиплексора будет получено $1 + (m - 1) + (m - 1) = 2 \cdot m - 1$ символов. Отношение $\frac{1}{2 \cdot m - 1}$ задает скорость турбокодирования.

А для турбокодов, построенных на рекурсивных сверточных кодах в несистематическом виде, справедлива лемма 5.2.

Лемма 5.2. Скорость турбокода, построенного на рекурсивных сверточных кодах с $R_{CK} = 1/m$ в несистематическом виде, задается выражением

$$R_{TK} = \frac{1}{2 \cdot m}. \qquad (5.2)$$

Доказательство аналогично доказательству леммы 5.1 и очевидно. Если каждый сверточный кодер задан в несистематическом виде, то на вход мультиплексора подается по m символов, без разделения их на информационные и проверочные. В результате для каждого информационного символа, поданного на вход турбокодера, будет сформировано $2 \cdot m$ кодовых (выходных) символов.

Таким образом, в рассмотренной схеме (рис. 5.1) формирование кодового слова осуществляется считыванием информационных и проверочных символов с двух параллельно соединенных сверточных кодеров. В канал связи поступает два кодовых слова сверточных кодов, соединенных в одно кодовое слово турбокода. Зная правило работы перемежителя, всегда можно по известному информационному вектору $I = \{I_1, I_2, I_3, ...\}$ сформировать перемешанную последовательность $I^* = \{I^*_1, I^*_2, I^*_3, ...\}$. Вектор $I = \{I_1, I_2, I_3, ...\}$ в явном виде записан в кодовом слове $C = \{I_1, P_1, P^*_1, I_2, P_2, P^*_2, ...\}$, из которого также можно выделить проверочные части $P = \{P_1, P_2, P_3, ...\}$ и $P^* = \{P^*_1, P^*_2, P^*_3, ...\}$. Вектор $I = \{I_1, I_2, I_3, ...\}$ и вектор $P = \{P_1, P_2, P_3, ...\}$ в совокупности образуют кодовое слово первого систематического сверточного кода. Вектор $I^* = \{I^*_1, I^*_2, I^*_3, ...\}$ и вектор $P^* = \{P^*_1, P^*_2, P^*_3, ...\}$ в совокупности образуют кодовое слово второго систематического сверточного кода.

В ряде случаев рассматриваются также перфорированные (выколотые) турбокоды. Кодовое слово выколотого кода формируется путем выкалывания некоторых проверочных символов, в результате чего повышается скорость R_{TK} турбокода и несколько снижается его корректирующая способность.

Как уже отмечалось, параллельное соединение двух сверточных кодов и одновременная передача в канал связи их кодовых слов, соединенных в одно слово турбокода, позволяет существенно повысить энергетическую эффективность помехоустойчивого кодирования. Действительно, если энергетическая эффективность каждого сверточного кода в отдельности определяется его исправляющей способностью, т.е. величиной $t = \lfloor (d_\infty - 1)/2 \rfloor$, то энергетическая эффективность турбокода определяется некоторой средней величиной $t_{ср} = \lfloor (d_{ср} - 1)/2 \rfloor$, т.е. выражается через среднее значение $d_{ср}$ расстояний между кодовыми блоками. Причем, в большинстве случаев, $d_{ср} > d_\infty$ и, соответственно, $t_{ср} > t_\infty$. Графически эту особенность турбокодов представим на рис. 5.2.

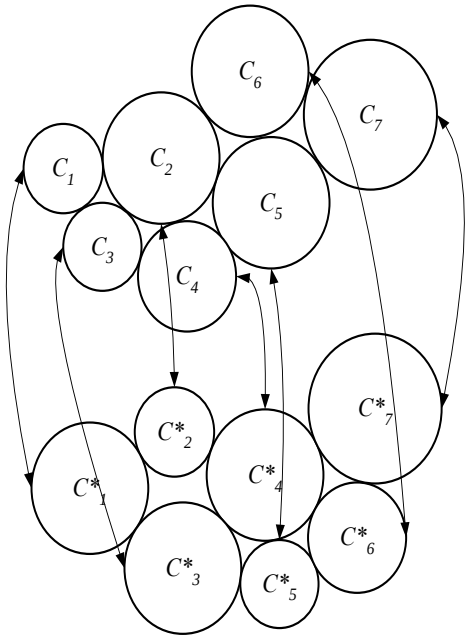


Рис. 5.2. Сферы упаковки турбокода

На рис. 5.2 представлены сферы упаковки турбокода в виде соответствующих сфер упаковки двух рекурсивных сверточных кодов. Символами C_1, C_2, ... обозначены кодовые слова первого сверточного кода. Радиус каждой сферы задает вес ошибки в соответствующем кодовом слове, которую потенциально можно исправить первым сверточным кодом. Минимальный радиус сфер, образованных вокруг всех кодовых слов C_1, C_2 ..., задает его исправляющую способность.

Символами C^*_1, C^*_2 ..., обозначены кодовые слова второго сверточного кода. Минимальный радиус сфер, образованных вокруг кодовых слов C^*_1, C^*_2 ..., задает исправляющую способность второго кода.

Стрелками обозначено объединение кодовых слов двух сверточных кодов в одно кодовое слово турбокода. Среднее значение между радиусами соответствующих кодовых слов задает вес ошибки, которую потенциально можно исправить турбокодом в этом слове. Минимальное значение из всех средних радиусов задает исправляющую способность турбокода. Этот показатель зависит от спектральных распределений сверточных кодов, т.е. от распределения радиусов сфер упаковки кода по различным кодовым словам. В то же время, вычисление спектра кода является одной из сложнейших задач теории

помехоустойчивого кодирования, которая решена на сегодняшний день лишь для узкого класса кодов.

Однако, применение рекурсивного сверточного кодера (кодер с обратной связью), имеющего неограниченную реакцию при воздействии на его вход единичного символа, позволяет получить наиболее благоприятную форму спектрального распределения, с точки зрения его влияния на вероятность ошибочного декодирования.

Для эффективной реализации турбокодов необходимо обеспечить параллельное декодирование первого и второго сверточного кода и, таким образом, реализовать их среднюю исправляющую способность. При этом решение об ошибке в каждом символе принятого слова должно приниматься взвешенно, с учетом соответствующего решения первого и второго декодера.

Структурная схема турбодекодера представлена на рис. 5.3. Турбодекодер работает следующим образом.

Поступившее кодовое слово $C = \{I_1, P_1, P^*_1, I_2, P_2, P^*_2, ...\}$ разделяется в демультиплексоре на информационную часть $I = \{I_1, I_2, I_3, ...\}$ и две проверочные части: $P = \{P_1, P_2, P_3, ...\}$ и $P^* = \{P^*_1, P^*_2, P^*_3, ...\}$. На первой итерации на вход первого декодера поступают "мягкие" оценки (решения) символов информационной I и первой проверочной частей P кодового слова первого сверточного кода. На выходе первого декодера формируется "мягкая" оценка (решение) информационных символов I_+, которая затем используется в качестве априорной информации для второго декодера.

Второй декодер производит "мягкую" оценку информационных символов с выхода перемежителя (I^*_+) на основе проверочной части P^* кодового слова второго сверточного кода.

На второй и последующих итерациях декодирования эта оценка обновляется и используется как "мягкая" априорная информация о переданном символе для первого декодера. Таким образом, на вход каждого из двух элементарных декодеров первого и второго сверточного кода поступают "мягкие" решения. В результате декодирования также формируется "мягкое" решение. В зарубежной литературе такая процедура декодирования получила название Soft Input Soft Output (SISO). Окончание декодирования происходит либо после выполнения

заданного количества итерационных циклов, либо после того, как величина поправки результата декодирования достигнет установленного порога.

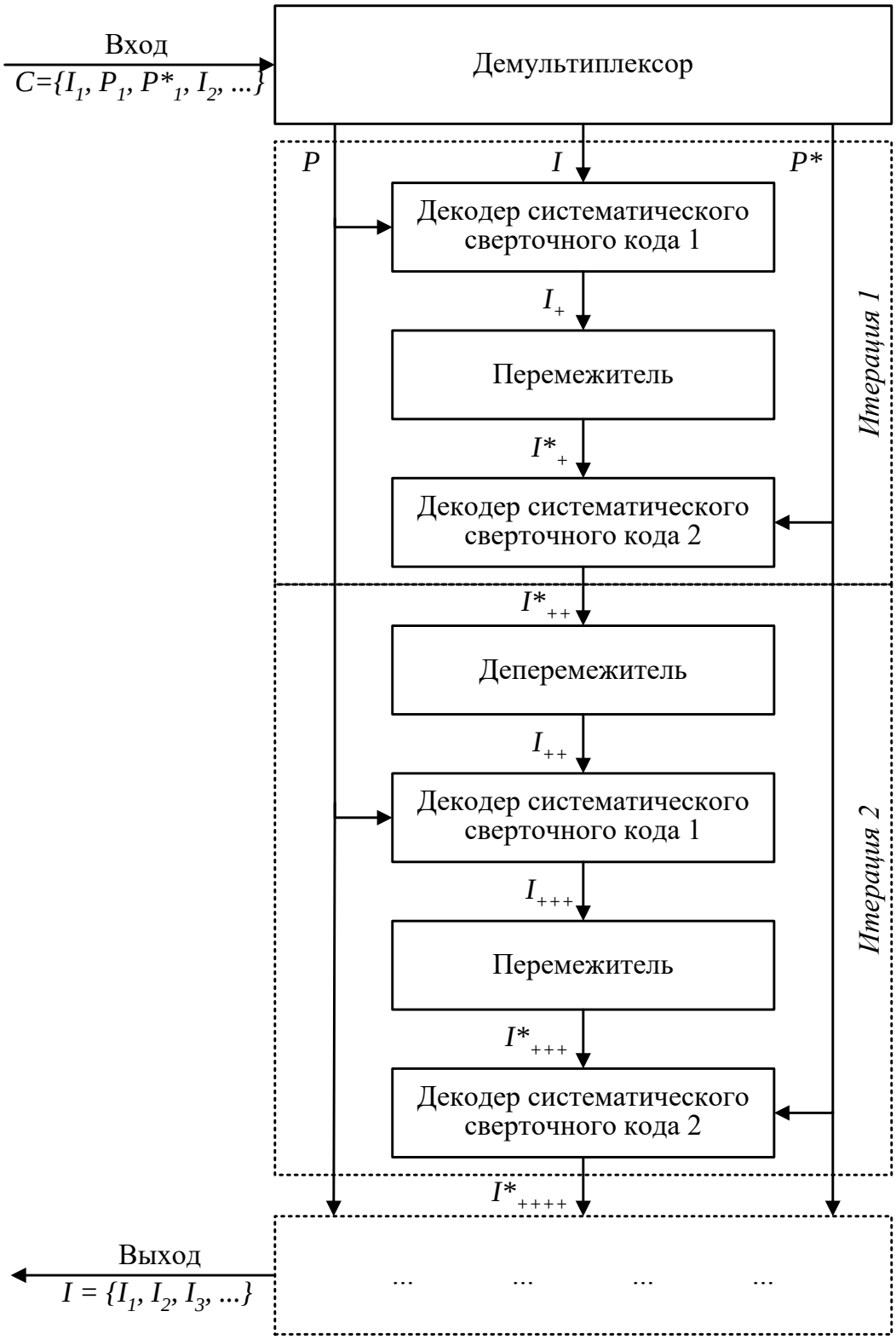


Рис. 5.3. Структурная схема турбодекодера

Анализ результатов экспериментальных исследований энергетической эффективности турбокодов показал, что структура перемежителя слабо влияет на его эффективность.

В то же время она пропорционально увеличивается с ростом длины кодового ограничения v используемых сверточных кодов.

Однако отсутствие регулярных алгебраических алгоритмов построения сверточных кодов с хорошими конструктивными (n, k, d_∞) параметрами и большим v сдерживает дальнейшее развитие методов турбокодирования.

Актуальной научно-технической задачей видится разработка и исследование турбокодов, построенных с использованием предложенных алгебраических рекурсивных сверточных кодов в систематическом и несистематическом виде.

5.2. Разработка турбокодов с использованием алгебраических рекурсивных сверточных кодов в несистематическом виде

Воспользуемся алгебраическим методом построения рекурсивных сверточных кодов для определения турбокодов с заранее заданными конструктивными характеристиками.

Для этого используем теоремы 3.4–3.5 (раздел 3), дающие мощный механизм построения алгебраических рекурсивных сверточных кодов в несистематическом виде, их параметры алгебраически связаны с параметрами недвоичных циклических кодов.

Рассмотрен рекурсивный сверточный кодер, построенный в виде недвоичного регистра сдвига с обратными связями (см. рис. 3.3). Такой кодер реализует пакетную обработку данных по m символов из $GF(q)$ или, что эквивалентно, по одному символу из $GF(q^m)$. Соответствующая схема турбокодера, построенного на алгебраических рекурсивных сверточных кодах, с обработкой элементов из $GF(q^m)$ представлена на рис. 5.4.

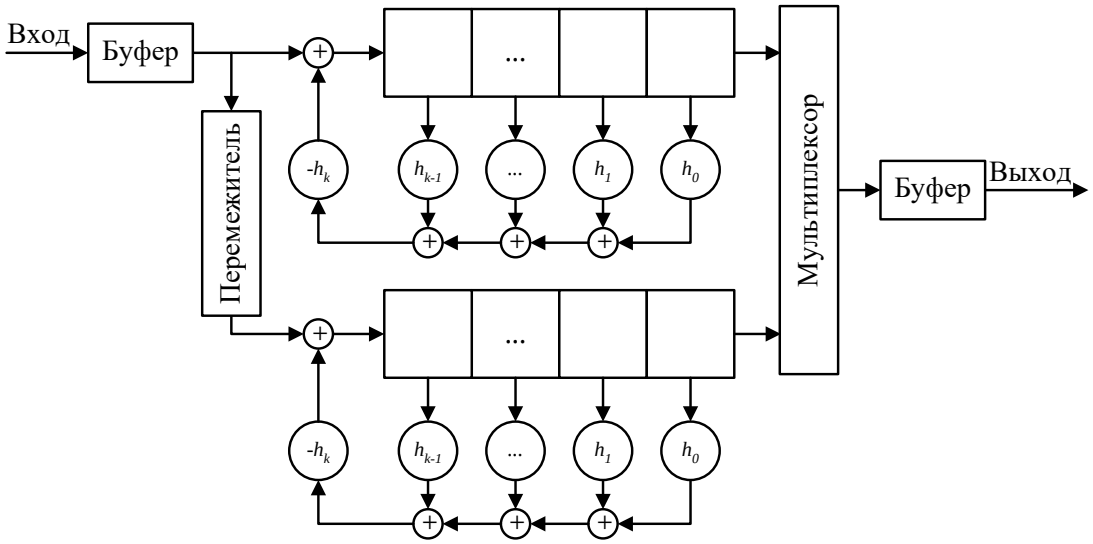


Рис. 5.4.Схема турбокодера на алгебраических рекурсивных сверточных кодах в несистематическом виде с обработкой элементов из $GF(q^m)$

Устройство, схема которого представлена на рис. 5.4, работает следующим образом. На вход кодера поступает информационная последовательность с символами из $GF(q)$. Во входном буфере символы из $GF(q)$ преобразуются в символы из $H \subseteq GF(q^m)$, и, согласно теореме 3.5, сопоставляются символам из $GF(q^m)$. Полученные символы из $GF(q^m)$ поступают на вход первого алгебраического рекурсивного сверточного кодера в несистематическом виде и, через перемежитель, на вход второго кодера. Выходные символы из $GF(q^m)$ поступают на мультиплексор, где формируется кодовое слово турбокода с элементами из $GF(q^m)$. Выходной буфер преобразует символы из $GF(q^m)$ в кодовые символы из $GF(q)$.

Справедлива следующая теорема.

Теорема 5.1. Турбокодер, построенный на алгебраических рекурсивных сверточных кодах в несистематическом виде, имеет скорость кодирования

$$R = \frac{k^0}{2 \cdot m}. \qquad (5.3)$$

Доказательство. Каждый сверточный кодер в схеме турбокодера (рис. 3.4) построен в виде цепи регистров с обратными связями, где отводы в цепи обратной связи задаются коэффициентами проверочного многочлена $h(x)$ недвоичного рекурсивного циклического (N, K, D) кода в несистематическом виде над $GF(q^m)$. Для каждых K введенных информационных символов из $GF(q^m)$ кодовое слово на выходе каждого сверточного кодера суть кодовое слово циклического (N, K, D) кода над $GF(q^m)$.

Если на вход каждого кодера подавать непрерывную последовательность, то, согласно теоремам 3.4–3.5, получим правило сверточного кодирования с параметрами

$$\nu = K \cdot k^0,\ n^0 = m,\ k = (K + 1)\cdot k^0,\ n = (K + 1)\cdot n^0,\ R = k^0/m,\ d_\infty \geq D.$$

Следовательно, для каждых k^0 входных символов из $GF(q)$ на выходе каждого сверточного кодера будет сформировано $n^0 = m$ символов из $GF(q)$ или, что эквивалентно, по одному символу из $GF(q^m)$. В мультиплексоре эти символы поочередно считываются, а затем в выходном буфере преобразуются в последовательность из $2\cdot m$ символов из $GF(q)$. Следовательно, для каждых k^0 входных символов из $GF(q)$ турбокодером будет сформировано $2\cdot m$ символов из $GF(q)$, что и задает скорость турбокода по выражению (5.3).

Следствие. Если $k^0 = 1$, то, в соответствии с теоремой 2.4, имеем рекурсивный сверточный код в несистематическом виде с параметрами

$$\nu = K,\ n^0 = m,\ k = K + 1,\ n = (K + 1)\cdot n^0,\ R = 1/m,\ d_\infty \geq D.$$

Соответствующий турбокодер имеет скорость кодирования $R_{TK} = 1/(2\cdot m)$, что соответствует лемме 5.2.

Другую схему турбокодера построим на алгебраических сверточных кодах в несистематическом виде с обработкой символов из $GF(q)$. Для этого воспользуемся кодерами, представленными на рис. 3.4. Соответствующая схема турбокодера в общем виде приведена на рис. 5.5.

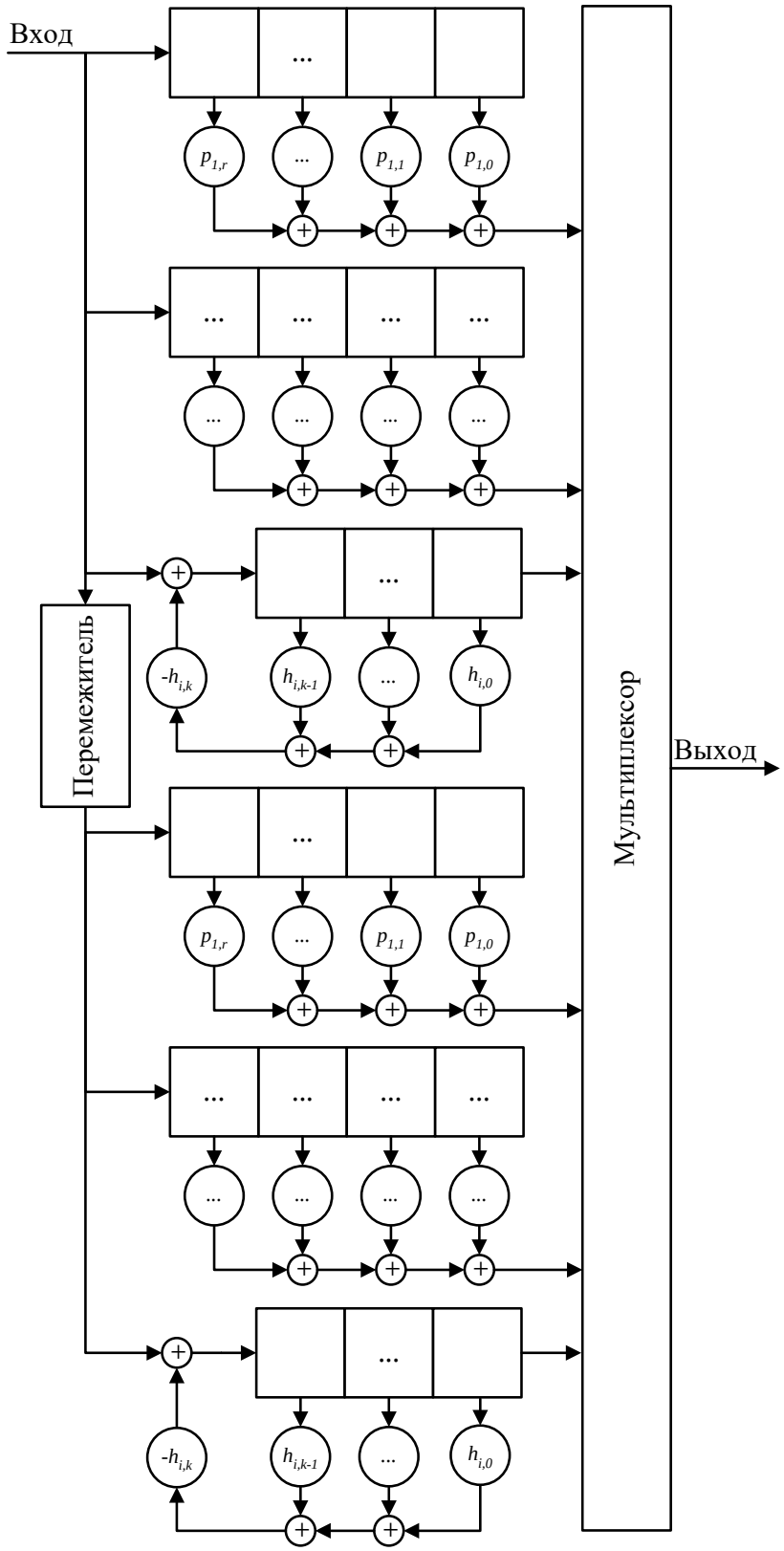


Рис. 5.5. Схема турбокодера на алгебраических рекурсивных сверточных кодах в несистематическом виде с обработкой элементов из *GF(q)*

Следует отметить особенности работы устройств, схемы которых приведены на рис. 5.4 и 5.5. Они отличаются способом обработки входных данных. Так, кодер, приведенный на рис. 5.5, реализует посимвольную обработку элементов из *GF(q)*. Кодер же, приведенный на рис. 5.4, реализует посимвольную обработку элементов из $GF(q^m)$. Если выполняются условия теорем 3.4–3.5, то последовательности на выходе соответствующих устройств совпадут.

Рассмотрим *пример* построения турбокодера на алгебраических рекурсивных сверточных кодах.

Зададим РС код (7, 3, 5) над $GF(2^3)$ и построеные на его основе алгебраические рекурсивные сверточные коды из примеров 3.1, 3.2.

На рис. 5.6 приведена соответствующая схема турбокодера с обработкой элементов из $GF(2^3)$, а на рис. 5.7 – с обработкой двоичных элементов.

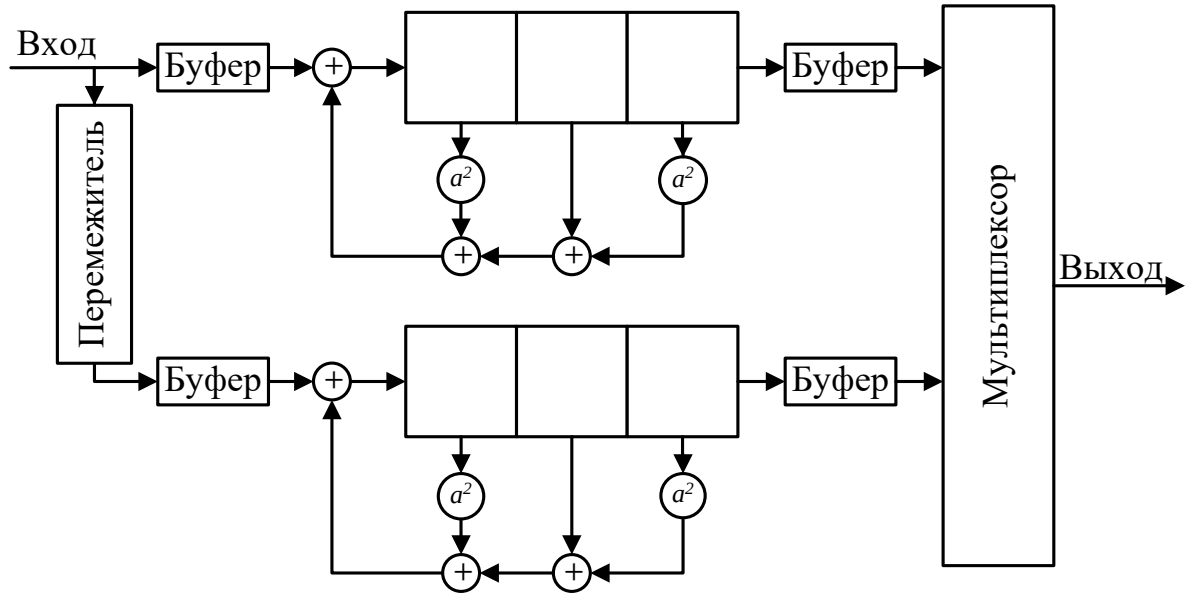


Рис. 5.6. Схема турбокодера на алгебраических рекурсивных сверточных кодах с обработкой символов из $GF(2^3)$

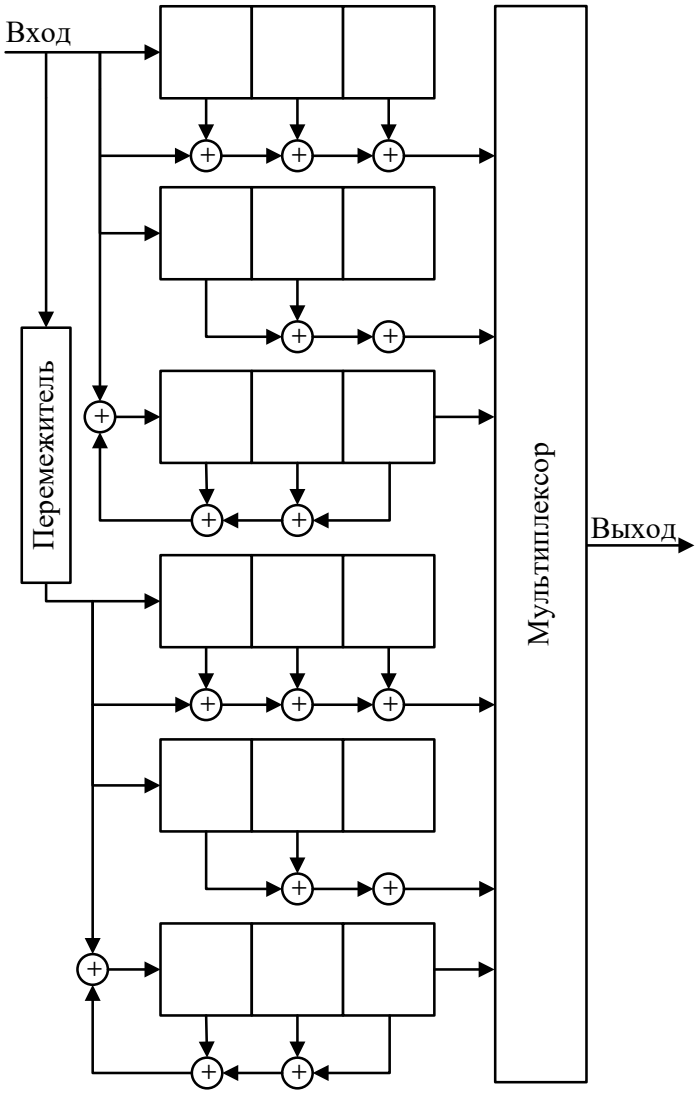


Рис. 5.7. Пример турбокодера на алгебраических рекурсивных сверточных кодах с обработкой двоичных символов

5.3. Разработка турбокодов с использованием алгебраических рекурсивных сверточных кодов в систематическом виде

Для построения турбокодов на алгебраических рекурсивных сверточных кодах в систематическом виде воспользуемся также теоремами 3.4–3.5, дающими мощный механизм построения алгебраических рекурсивных сверточных кодов в систематическом виде, их параметры алгебраически связаны с параметрами недвоичных циклических кодов.

Рассмотрим рекурсивный сверточный кодер, построенный в виде недвоичного регистра сдвига с обратными связями (см. рис. 3.18). Такой кодер реализует пакетную обработку данных по m символов из $GF(q)$ или, что эквивалентно, по одному символу из $GF(q^m)$.

Схема турбокодера, построенного на алгебраических рекурсивных сверточных кодах с обработкой элементов из $GF(q^m)$, представлена на рис. 5.8.

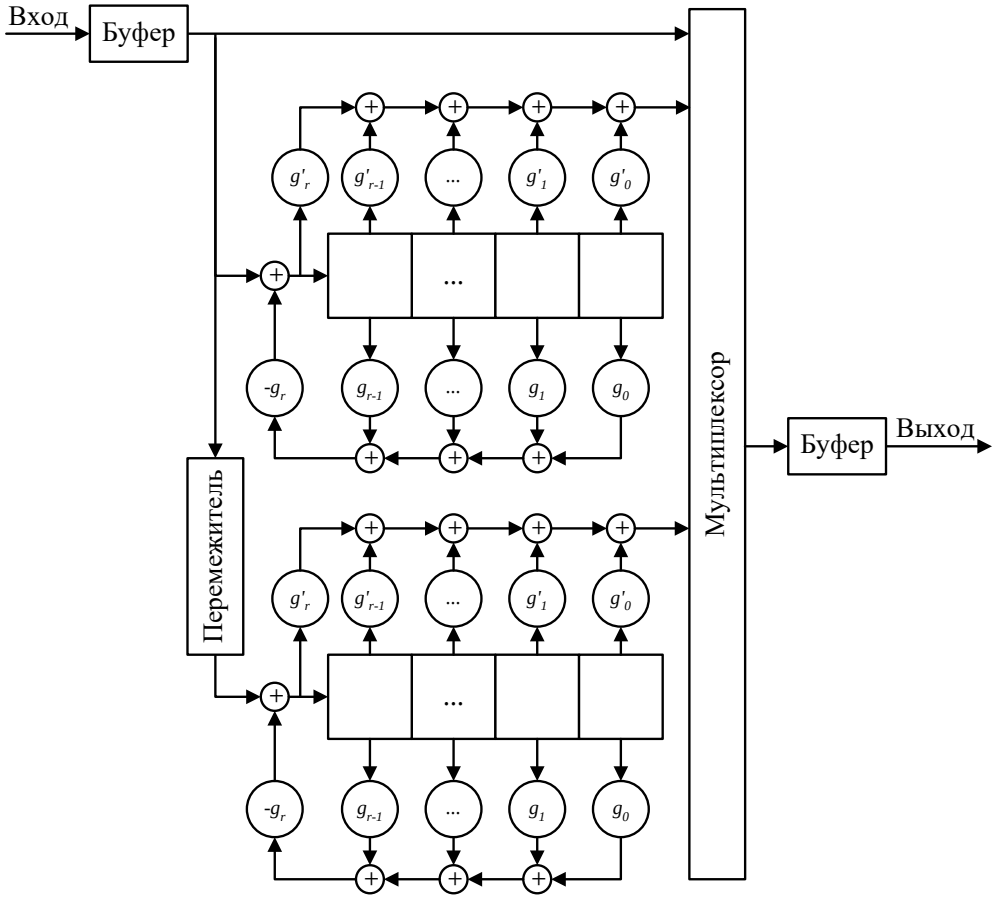


Рис. 5.8. Схема турбокодера с использованием алгебраических рекурсивных сверточных кодов в систематическом виде с обработкой элементов из $GF(q^m)$

Устройство работает следующим образом. На вход кодера поступает информационная последовательность с символами из $GF(q)$. Во входном буфере символы из $GF(q)$ преобразуются в символы из $H \subseteq GF(q^m)$ и, согласно теореме 3.5, сопоставляются символам из $GF(q^m)$. Полученные символы из $GF(q^m)$ поступают

на вход первого алгебраического рекурсивного сверточного кодера в систематическом виде и через перемежитель подаются на вход второго кодера. Информационные символы из *GF(q)* и проверочные символы с первого и второго сверточных кодеров из $GF(q^m)$ поступают на мультиплексор, где формируется кодовое слово турбокода с элементами из $GF(q^m)$. Выходной буфер преобразует символы из $GF(q^m)$ в кодовые символы из *GF(q)*.

Для определения параметров построенного таким образом турбокода сформулируем и докажем следующую теорему.

Теорема 5.2. Турбокодер, построенный на алгебраических рекурсивных сверточных кодах в несистематическом виде, имеет скорость кодирования

$$R_{TK} = \frac{k^0}{2 \cdot n^0 - k^0} = \frac{K \cdot \log_q |H|}{2 \cdot (N-K) \cdot m + K \cdot \log_q |H|}. \qquad (3.4)$$

Доказательство. Каждый сверточный кодер в схеме турбокодера на рис. 5.18 построен в виде цепи регистров с обратными связями, где отводы в цепи обратной связи задаются коэффициентами порождающего многочлена *g(x)* недвоичного рекурсивного циклического *(N, K, D)* кода в систематическом виде над $GF(q^m)$. Для каждых $N - K$ введенных информационных символов из $GF(q^m)$ кодовое слово на выходе каждого сверточного кодера является кодовым словом циклического *(N, K, D)* кода над $GF(q^m)$. Если на вход каждого кодера подавать непрерывную последовательность, то, согласно теоремам 3.4–3.5, полученное отображение представляет правило сверточного кодирования с параметрами

$$\nu = (N-K) \cdot K \cdot \lfloor \log_q H \rfloor; \; k^0 = K \cdot \log_q \lfloor H \rfloor; \; n^0 = ((N-K) \cdot m + K \cdot \log_q \lfloor H \rfloor);$$

$$k = (N-K+1) \cdot K \cdot \log_q \lfloor H \rfloor; \; n = (N-K+1) \cdot ((N-K) \cdot m + K \cdot \log_q \lfloor H \rfloor);$$

$$R = K \cdot \log_q \lfloor H \rfloor / ((N-K) \cdot m + K \cdot \log_q \lfloor H \rfloor), \; d_\infty \geq D.$$

Следовательно, для каждых $k^0 = K \cdot \log_q |H|$ входных символов из $GF(q)$ на выходе каждого сверточного кодера будет сформировано $n^0=((N–K)\cdot m+K\cdot \log_q |H|)$ символов из $GF(q)$. Сверточный код задан в систематическом виде, следовательно, блок из $((N–K)\cdot m+K\cdot \log_q |H|)$ кодовых символов содержит $K\cdot \log_q |H|$ информационных и $(N–K)\cdot m$ проверочных символов. В мультиплексор поочередно поступают информационные символы, с выхода каждого сверточного кодера - проверочные символы. В выходном буфере считанные символы из $GF(q)$ отображаются в поле из $GF(q^m)$.

Следовательно, для каждых $k^0=K\cdot \log_q |H|$ входных символов из $GF(q)$ турбокодером будет сформировано $(2\cdot n^0–k^0) = (2\cdot(N–K)\cdot m+K\cdot \log_q |H|)$ символов из $GF(q)$, что и задает скорость турбокода

$$R_{TK} = k^0/(2\cdot n^0–k^0) = K\cdot \log_q |H|/(2\cdot(N–K)\cdot m+K\cdot \log_q |H|).$$

Следствие 1. Если $k^0 = K$, $n^0 = N$, то, по теореме 3.4, имеем рекурсивный сверточный код в систематическом виде с параметрами $\nu =(N–K)\cdot K$; $k^0=K$; $n^0=N$; $k=(N–K+1)\cdot K$; $n=(N–K+1)\cdot N$; $R=K/N$, $d_\infty \geq D$.

Соответствующий турбокодер имеет скорость кодирования $R_{TK} = k^0/(2\cdot n^0–k^0) = K/(2\cdot N–K)$, т.е. скорость турбокода будет определяться исключительно скоростью циклического (N, K, D) кода.

Следствие 2. Если $k^0 = K = 1$, $n^0 = N$, то, согласно теореме 2.4, имеем рекурсивный сверточный код в систематическом виде с параметрами $\nu =N–1$; $k^0=1$; $n^0=N$; $k=N$; $n=N^2$; $R=1/N$;, $d_\infty \geq D$. Соответствующий турбокодер имеет скорость кодирования $R_{TK} = k^0/(2\cdot n^0–k^0) = 1/(2\cdot N–1)$, что соответствует обобщению результата леммы 5.1.

Другую схему турбокодера на алгебраических сверточных кодах в систематическом виде построим с обработкой символов из $GF(q)$. Для этого рассмотрим соответствующие кодеры сверточных кодов, представленные на рис. 3.12. Схема турбокодера в общем виде приведена на рис. 5.9.

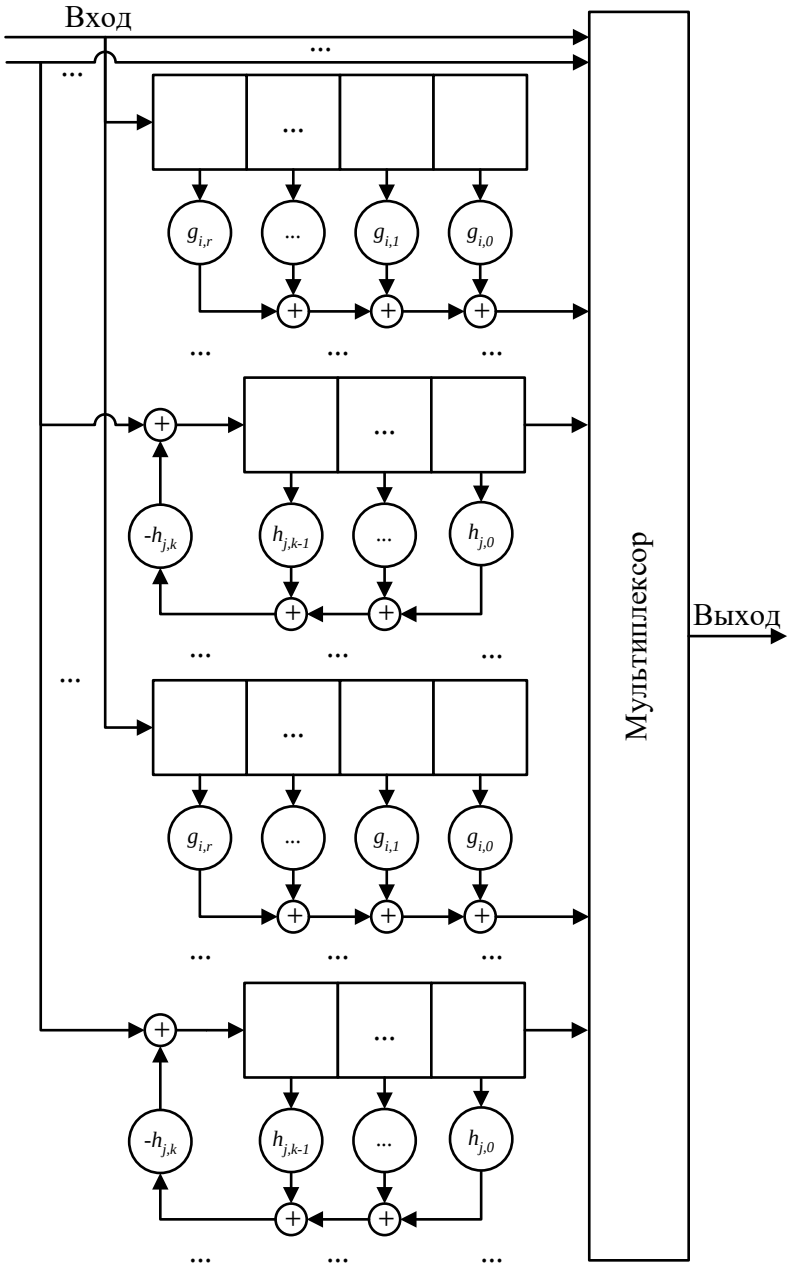


Рис. 5.9. Схема турбокодера с использованием алгебраических рекурсивных сверточных кодов в систематическом виде с обработкой элементов из *GF(q)*

Рассмотрим *пример* построения турбокодера с использованием алгебраических рекурсивных сверточных кодов в систематическом виде.

Зафиксируем конечное поле $GF(2^2)$, построенное по кольцу многочленов $\{0 = \alpha^{-\infty}, 1 = \alpha^0, x = \alpha^1, x + 1 = \alpha^2\}$ с операциями, по модулю $G(x) = x^2 + x + 1$. Зададим (3, 2, 2) код РС с

порождающим многочленом $g(x) = x + \alpha^2$ и алгебраический рекурсивный сверточный код в систематическом виде с параметрами $\nu = 2$; $k^0 = 1$; $n^0 = 2$; $k=3$; $n = 6$; $R = 1 / 2$; $d_\infty \geq 2$ (см. пример 2.3). Построим турбокодер с обработкой двоичных символов, схема кодера приведена на рис. 5.10.

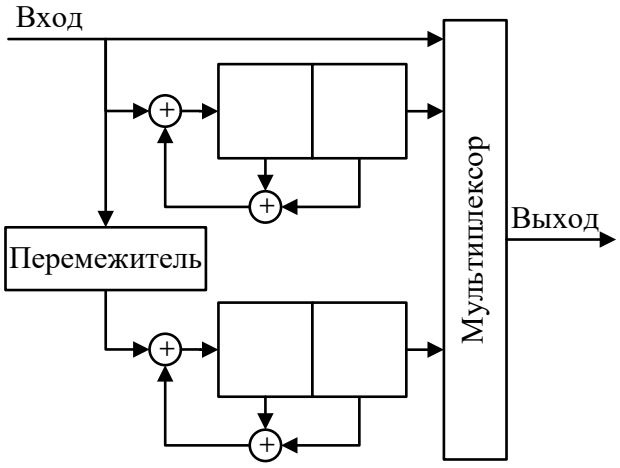


Рис. 5.10. Схема турбокодера с использованием алгебраического рекурсивного сверточного кода в систематическом виде с обработкой двоичных элементов (вариант)

Воспользуемся теоремой 5.2. Скорость алгебраически заданного турбокода определяется выражением (5.4) и в данном случае она равна

$$R_{TK} = \frac{k^0}{2 \cdot n^0 - k^0} = \frac{1}{3},$$

что соответствует также результату леммы 5.1.

Приведенный пример наглядно демонстрирует возможности предложенного подхода турбокодирования с использованием алгебраических рекурсивных сверточных кодов.

Для использования предложенного подхода разработаем практический алгоритм построения турбокодов с требуемыми параметрами.

5.4. Разработка алгоритма построения турбокодов с использованием алгебраических рекурсивных сверточных кодов

Предположим, что в параллельной каскадной схеме с требуемой скоростью R_{TK} необходимо использовать сверточные коды с фиксированными k^0 и n^0 параметрами. Тогда для формирования порождающих и/или проверочных многочленов сверточного кода необходимо выбрать соответствующий недвоичный циклический код и вариант его использования. Подробно эти вопросы рассмотрены в подразделе 3.4 (см. рис. 3.14).

Общая схема предлагаемого алгоритма построения турбокодов на алгебраических рекурсивных сверточных кодах приведена на рис. 5.11. Алгоритм состоит из последовательности следующих шагов:

ШАГ 1. Ввод требуемой скорости турбокода R_{TK}, требуемых k^0 и n^0 параметров соответствующих сверточных кодов, ввод мощности алфавита кодовых символов q.

ШАГ 2. Расчет скорости $R_{ск}$ рекурсивных сверточных кодов.

ШАГ 3. Выбор способа обработки кодовых символов.

ШАГ 4. Выбор варианта построения рекурсивных сверточных кодов, расчет параметров соответствующего циклического кода над $GF(q^m)$, формирование порождающих многочленов и построение схемы кодера сверточного кода над $GF(q)$ (*по отдельному алгоритму, см. раздел 3*).

ШАГ 5. Построение параллельной каскадной схемы с алгебраическими рекурсивными сверточными кодами.

Рассмотрим последовательно выполнение каждого шага предложенного алгоритма отдельно.

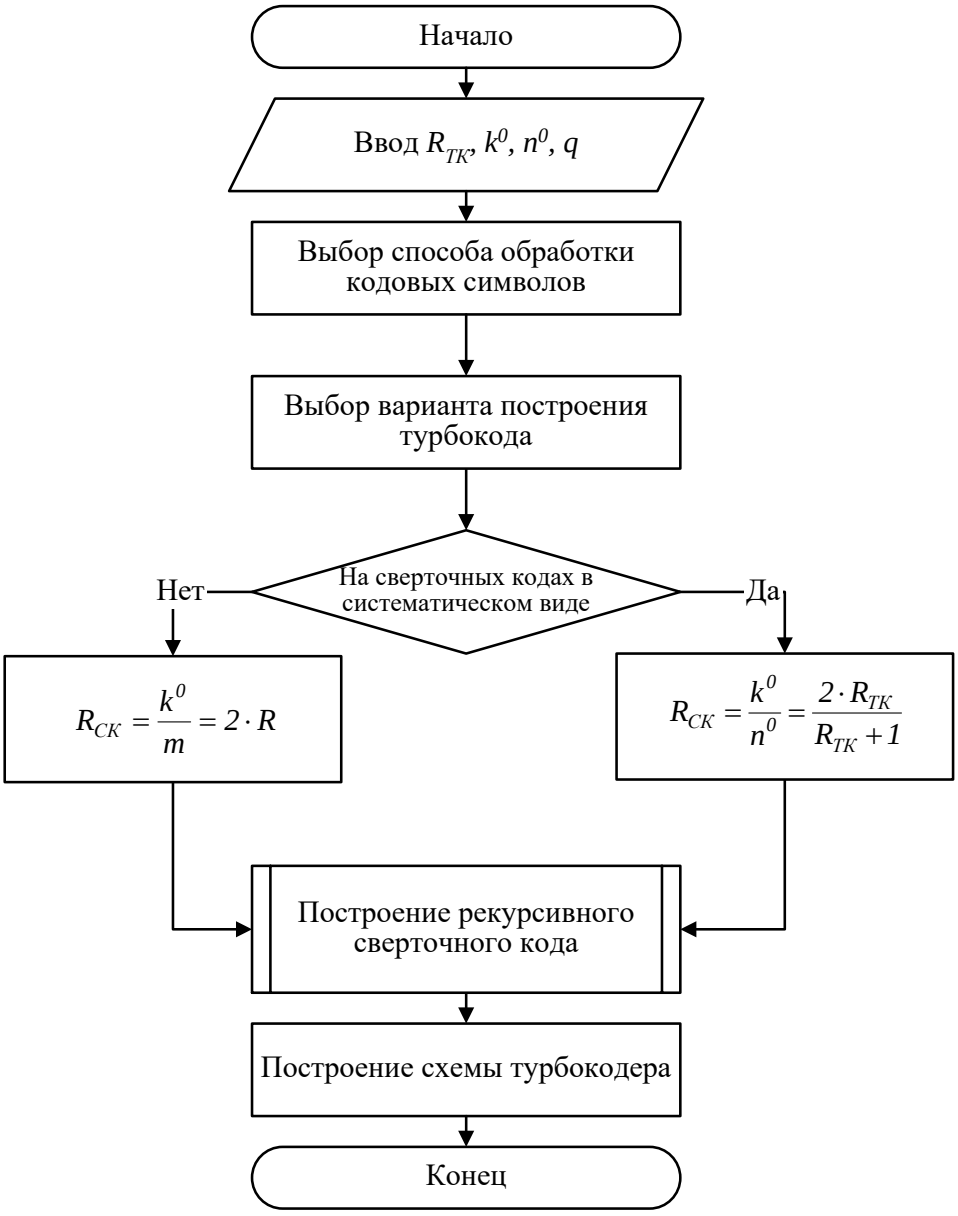


Рис. 5.21. Общая схема алгоритма построения турбокодов с использованием алгебраических рекурсивных сверточных кодов

Рассмотрим параллельные каскадные схемы турбокодирования для алгебраических рекурсивных сверточных кодов в несистематическом виде. Предположим, что необходимо построить турбокод со скоростью R_{TK}. Тогда скорость R_{CK} соответствующих сверточных кодов можно получить из выражения (5.3) теремы 5.1:

$$R_{CK} = \frac{k^0}{n^0} = 2 \cdot R_{TK}\,. \qquad (5.5)$$

Затем выполняется процедура алгебраического построения рекурсивных сверточных кодов (см. раздел 2, рис. 2.12). Расчет кодовых параметров соответствующего циклического кода выполняется по выражениям (3.11) и/или (3.12).

Рассмотрим параллельные каскадные схемы турбокодирования на алгебраических рекурсивных сверточных кодах в систематическом виде.

Теперь скорость $R_{ск}$ соответствующих сверточных кодов можно получить из выражения (5.4) теремы 5.2:

$$R_{TK} \cdot \left(2 \cdot n^0 - k^0\right) = k^0 .$$

Раскроем скобки и приведем подобные, получим:

$$2 \cdot n^0 \cdot R_{TK} - k^0 \cdot R_{TK} - k^0 = 2 \cdot n^0 \cdot R_{TK} - k^0 \cdot \left(R_{TK} + 1\right) = 0 .$$

Разделим обе части уравнения на n^0, получим

$$2 \cdot R_{TK} - \frac{k^0}{n^0} \cdot \left(R_{TK} + 1\right) = 0 ,$$

откуда имеем

$$R_{CK} = \frac{k^0}{n^0} = \frac{2 \cdot R_{TK}}{R_{TK} + 1} . \qquad (5.6)$$

После расчета скорости R_{CK} сверточных кодов выполняется процедура алгебраического построения рекурсивных сверточных кодов. Расчет кодовых параметров соответствующего циклического кода выполняется по выражениям (3.13) и/или (3.14).

На пятом шаге разработанного алгоритма строится параллельная каскадная схема с полученными алгебраическими рекурсивными сверточными кодами. При этом учитывается вид сверточного кодера (в систематическом и несистематическом виде) и способ обработки кодовых символов (посимвольно, элементами из *GF(q)*, или посимвольно, элементами из $GF(q^m)$). Особенности построения рекурсивных сверточных кодов определяются теоремами 3.4–3.7, а параллельных каскадных схем – теоремами 5.1–5.2.

Выводы

1. При дальнейшем развитии методов построения параллельных каскадных схем сверточного кодирования получены новые схемы, отличающиеся от известных использованием алгебраически заданных рекурсивных сверточных кодов в систематическом и несистематическом виде, что позволяет упростить описание турбокодов и алгебраически задавать их параметры.

2. Теоретически обоснованы процедуры алгебраического построения параллельных каскадных схем с использованием алгебраически заданных рекурсивных сверточных кодов. Доказанные теоремы 5.1.–5.2 позволяют аналитически связать параметры турбокодов с параметрами алгебраических сверточных кодов в систематическом и несистематическом виде.

3. Разработан алгоритм построения турбокодов, который использует алгебраически заданные сверточные коды и позволяет за конечное число шагов формировать параллельные каскадные схемы с требуемыми характеристиками. Разработаны схемы турбокодеров с использованием алгебраически заданных рекурсивных сверточных кодов, позволяющие реализовать непрерывную обработку входных данных посимвольно – элементами из $GF(q)$ или элементами из $GF(q^m)$.

4. Исследованы особенности построения параллельных каскадных схем, выработаны рекомендации по использованию недвоичных циклических кодов и алгебраически заданных рекурсивных сверточных кодов при построении схем турбокодирования.

СПИСОК ЛИТЕРАТУРЫ

1. Берлекэмп Э.Р. Алгебраическая теория кодирования / Пер. с англ.– М.: Мир, 1971.– 477 с.
2. Бернард Скляр. Цифровая связь. Теоретические основы и практическое применение. – М.: Вильямс, 2003. – 1104 с.
3. Блейхут Р. Теория и практика кодов, контролирующих ошибки / Пер. с англ. – М.: Мир, 1986. – 576 с.

4. Вентцель Е.С., Овчаров Л.А. Теория вероятностей и ее инженерные приложения. – М.: Высшая школа, 2000. – 480 с.
5. Витерби А. Границы ошибок для сверточных кодов и асимптотически оптимальный алгоритм декодирования // Некоторые вопросы теории кодирования. – М.: Мир, 1970. – С. 142–165.
6. Витерби А.Д., Омура Дж. К. Принципы цифровой связи и кодирования / Пер. с англ.; Под ред. К.Ш. Зигангирова. – М.: Радио и связь, 1982. – 535 с.
7. Габидулин Э.М., Афанасьев В.Б. Кодирование в радиоэлектронике. – М.: Радио и связь, 1986. – 176 с.
8. Галлагер Р. Простой вывод теоремы кодирования и некоторые применения // Кибернетический сборник. Новая серия. – М.: Мир. – 1966. – Вып. 3. – С. 50–90.
9. Гантмахер Ф.Р. Теория матриц. – М.: Наука, 1988. – 552 с.
10. Гмурман В.Е. Теория вероятностей и математическая статистика. – М., 2002. – 480 с.
11. Дженнингс Ф. Практическая передача данных: модемы, сети и протоколы. – М.: Мир, 1989. – 272 с.
12. Додд А.З. Мир телекоммуникаций. Обзор технологий и отрасли / Пер. с англ. – М., 2002. – 400 с.
13. Долгов В.И. Основы статистической теории приема дискретных сигналов. – Харьков: ХВВКИУРВ, 1989. – 448 с.
14. Жураковський Ю.П., Полторак В.П. Теорія інформації та кодування. – К.: Вища школа, 2001. – 255 с.
15. Злотник Б. М. Помехоустойчивые коды в системах связи. – М.: Радио и связь, 1989. – 232 с.
16. Ильин В.А., Позняк Э.Г. Линейная алгебра. – М.: Наука. ФИЗМАТЛИТ, 1999. – 296 с.
17. Карпов Ю.Г. Теория автоматов. – С.Пб.: Питер, 2002. – 224 с.
18. Кларк Дж.–мл., Кейн Дж. Кодирование с исправлением ошибок в системах цифровой связи / Пер. с англ.; Под ред. Б.С. Цыбакова. – М.: Радио и связь, 1987. – 392 с.
19. Ключко В.И. Защита информации от ошибок в АСУ. – МО СССР, 1980. – 256 с.
20. Коржик В.И., Финк Л.М. Помехоустойчивое кодирование дискретных сообщений в каналах со случайной структурой. – М.: Связь, 1975. – 272 с.

21. Коржик В.И., Финк Л.М., Шелкунов К.Н. Расчет помехоустойчивости систем передачи дискретных сообщений: Справочник. – М.: Радио и связь, 1981. – 232 с.
22. Коричнев Л.П., Королев В.Д. Статистический контроль каналов связи. – М.: Радио и связь, 1989. – 240 с.
23. Коррекция ошибок в оптических накопителях информации // А.П. Типикин, В.В. Петров, А.Г. Бабанин; Отв. ред. А.Г. Додонов; АН УССР. Ин–т проблем регистрации информации. – К.: Наукова думка, 1990. – 172 с.
24. Краснобаев В.А., Приходько С.И., Снисаренко А.Г. Помехоустойчивое кодирование в АСУ. - Харьков: ХВВКИУРВ, 1990. – 155 с.
25. Кремер Н.Ш. Теория вероятностей и математическая статистика. – М., 2000. – 543 с.
26. Кузнецов А.А., Гусев С.А., Жученко А.С., Палажченко С.И. Применение алгебраических сверточных кодов для построения турбокодов // Системи обробки інформації. – Харків: ХВУ. –2004. – Вип. 10(38). – С. 98–102.
27. Кузнецов А.А., Приходько С.И., Гусев С.А., Кужель И.Е. Алгебраический метод сверточного кодирования // Комп'ютерні системи та інформаційні технології. – Харьков: ХАИ. – 2005.– №1. – С.46–52.
28. Лагутенко О.И. Современные модемы. – М.: Эко–Трендз, 2002. – 343 с.
29. Левин Л.С., Плоткин М.А. Цифровые системы передачи информации. – М.: Радио и связь, 1982. – 215 с.
30. Лидл Р., Нидеррайтер Г. Конечные поля / Пер. с англ.: В 2т. – М.: Мир, 1988.– Т. 1. – 430 с.
31. Лидл Р., Нидеррайтер Г. Конечные поля / Пер. с англ.: В 2т. – М.: Мир, 1988.– Т. 2. – 392 с.
32. Мак–Вильямс Ф.Дж., Слоэн Н.Дж.А. Теория кодов, исправляющих ошибки.– М.: Связь, 1979. – 744 с.
33. Мутер В.М. Основы помехоустойчивой телепередачи информации. – Л.: Энергоатомиздат. Ленингр. отд–ние, 1990. – 288 с.
34. Передача дискретных сообщений / В.П. Шувалов, Н.В. Захарченко, В.О. Шварцман, С.Д. Свет, Г.И. Скворцов, В.В. Лебедянцев. – М.: Радио и связь, 1990. – 464 с.

35. Питерсон У., Уэлдон Э. Коды, исправляющие ошибки / Пер. с англ.; Под ред. Р.Л. Добрушина и С.И. Самойленко. – М.: Мир, 1976. – 576 с.
36. Помехоустойчивость и эффективность систем передачи информации / А.Г. Зюко, А.И. Фалько, И.П. Панфилов, В.Л. Банкет, П.В. Иващенко; Под. ред. А.Г. Зюко. – М.: Радио и связь, 1985. – 272 с.
37. Поставной В.И. Теория передачи сигналов. – МО СССР, 1985. – 264 с.
38. Приходько С.И. Алгебраические процедуры декодирования сверточных кодов // Современные методы кодирования в электронных системах: Материалы международной НТК 23–24 апреля 2002 г. – Сумы: СМКЭС, 2002. – С.11–12.
39. Приходько С.И. Алгебраические сверточные коды // Інформаційно–керуючі системи на залізничному транспорті.– Харьков: ХарДАЗТ. - 1999. - №2. - С. 62–64.
40. Приходько С.И., Кузнецов А.А., Гусев С.А. Алгебраический метод сверточного кодирования // Современные методы кодирования в электронных системах: Материалы международной НТК 26–27 октября 2004 г. – Сумы: СМКЭС, 2004. – С.11–12.
41. Приходько С.И., Кузнецов А.А., Гусев С.А., Кужель И.Е. Алгебраическое построение несистематических сверточных кодов // Системи обробки інформації. – Харків: ХВУ. –2004 – Вип. 8(36). – С. 170–175.
42. Сорока Л.С., Приходько С.И. Основы построения АСУ. – Харьков: ХВВКИУ, 1988. – 132 с.
43. Судоплатов С.В., Овчинникова Е.В. Элементы дискретной математики. – М., 2002. – 280 с.
44. Уолрэнд Дж. Телекоммуникационные и компьютерные сети. – М.: Постмаркет, 2001. – 480 с.
45. Фигурин В.А., Оболонин В.В. Теория вероятностей и математическая статистика. – Мн., 2000. – 208 с.
46. Финк Л.М. Теория передачи дискретных сообщений. – М.: Советское радио, 1970. –728 с.
47. Хопкрофт Джон. Введение в теорию автоматов, языков и вычислений. – М.: Вильямс, 2002. – 528 с.

48. Цифровые телекоммуникационные сети / Г.В. Горелов, Н.А. Казанский, В.А. Кудряшов, О.Н. Ромашкова. – Харьков: Регион–информ: ХФИ "Транспорт Украины", 2000. – 213 с.
49. Шеннон К. Работы по теории информации и кибернетике. – М.: ИЛ, 1963. – 829 с.
50. Шеннон К. Связь при наличии шума // Теория информации и ее приложения. Сборник переводов. – М.: ФИЗМАТГИЗ, 1959. – С. 82–112.
51. Элементы теории передачи дискретной информации / Л.П. Пуртов, А.С. Замрий, А.И. Захаров, В.М. Охорзин; Под ред. Л.П. Пуртова. – М.: Связь, 1972. – 232 с.
52. Якубайтис В.А. Архитектура вычислительных сетей. – М.: Статистика, 1980. – 279 с.
53. Andersen J.D. Selection of component codes for turbo coding based on convergence properties //Annales des Telecommunications. Special issue on turbo codes, march – april 1999. – 1999. – Vol. 54, No 3–4. http://www.tele.dtu.dk/~jda/
54. Berrou C., Glavieux A. Near Optimum Error Correcting Coding and Decoding: Turbo–Codes // IEEE Trans. On Comm., Vol. 44, No. 10, October 1996.
55. Berrou C., Glavieux A, Thitimajshima P. Near Shannon Limit Error–Correcting Coding and Decoding: Turbo–Codes // Proceedings of ICC'93, Geneva, Switzerland, pp. 1064–1070, May, 1993.
56. Brengarth N., Novello R., Pham N., Piloni V., Tousch J. DVB–RCS turbo code on a commercial OPB statellite payload: Skyplex // 2nd Int'l Symp. on Turbo Codes", Brest, France, Sept. 2000.
57. CCSDS 101.0–B–4: Telemetry Channel Coding. Blue Book. Issue 4. May 1999 (http://www.ccsds.org).
58. Douillard C., Jezequel M., Berrou C., Brengarth N., Tousch J., Pham N. The turbo code standard for DVB–RCS // 2nd International Symposium on turbo codes, Brest Sept 2000.
59. ETSI EN 301 790 V1.2.1 (2000–07) Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems (DVB– RCS) (www.etsi.org).
60. http://www.turboconcept.com

Н.И. Данько, С.П. Евсеев, А.А. Кузнецов,
П.Ф. Поляков, С.И. Приходько

АЛГЕБРАИЧЕСКИЕ СВЕРТОЧНЫЕ КОДЫ

Учебное пособие

Ответственный за випуск Приходько С.И.

Редактор Решетилова В.В.

Подписано к печати 17.04.06 г.

Формат бумаги 60х84 1/16 . Бумага писчая.

Усл.-печ. л 14,75 Уч.-изд. л. 15,0.

Заказ № Тираж 150 Цена

Издательство УкрГАЖТа, свидетельство ДК № 112 от 06.07.2000 г.
Типография УкрГАЖТа,
61050, Харьков - 50, пл. Фейербаха, 7